The Pennsylvania State University
The Graduate School

BLOCKCHAIN ENABLED VACCINE SUPPLY CHAIN PROVENANCE

A Thesis in
Industrial Engineering
by
Mit Piyush Goda

© 2021 Mit Piyush Goda

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2021
The thesis of Mit Piyush Goda is to be reviewed and approved by the following:

Soundar R.T. Kumara
Allen E. Pearce/Allen M. Pearce Professor of Industrial Engineering
Thesis Adviser

Saurabh Basu
Assistant Professor of Industrial and Manufacturing Engineering

Robert Voigt
Professor of Industrial and Manufacturing Engineering
Professor and Graduate Program Coordinator
ABSTRACT

This thesis aims to understand and implement a proof-of-concept study on how blockchain can be used to provide traceability in a Vaccine supply chain. Provenance in a vaccine supply chain provides an opportunity to counter the serious problems of counterfeiting and substandard products. Blockchain is a distributed ledger that allows untrusted multiple stakeholders in a shared network to exchange transactions or confidential data. It provides consensus, security, provenance, and trust in the supply chain network consisting of multiple manufacturers, distributors, clinics and patients. A detailed literature review of existing vaccine supply chain standards, traceability technologies and blockchain use cases in multiple fields is also done for this thesis. The outbreak of covid-19 pandemic and an emergency use case for deployment of vaccines motivated the research in the thesis and also formed the basis of a blockchain network that was deployed privately to understand the applications and extent of provenance that can be achieved. The blockchain network was deployed on a test Ethereum network using Ganache and a wallet of MetaMask to replicate a real-life scenario. Future research can study the integration of IoT with blockchain in detail and the opportunity of global traceability and reduction in costs by deploying Distributed Applications (DApps) for the Supply Chain.
# TABLE OF CONTENTS

LIST OF FIGURES ...................................................................................................................................... vi
LIST OF TABLES ......................................................................................................................................... viii
ACKNOWLEDGEMENTS ............................................................................................................................ ix

Chapter 1  Introduction ............................................................................................................................... 1
  1.1 Motivation ........................................................................................................................................... 1
  1.2 Problem definition ............................................................................................................................. 1
  1.3 Brief description of the methodology ............................................................................................... 2
  1.4 Results ............................................................................................................................................... 3
  1.5 Organization of Thesis ...................................................................................................................... 3

Chapter 2  Background ............................................................................................................................... 1
  2.1 Vaccine Supply Chain ....................................................................................................................... 1
    2.1.1 Vaccine Manufacturing and Distribution .................................................................................... 3
    2.1.2 Vaccine Approval and Lot release ............................................................................................. 6
    2.1.3 Vaccine Supply Chain in COVID-19 ......................................................................................... 8
  2.2 Blockchain ......................................................................................................................................... 10
    2.2.1 Blockchain Components ........................................................................................................... 11
      2.2.1.1 Hash function ...................................................................................................................... 11
      2.2.1.2 Asymmetric Cryptography ................................................................................................. 12
      2.2.1.3 Digital Signature ................................................................................................................. 12
      2.2.1.5 Consensus mechanisms ..................................................................................................... 15
    2.2.2 Blockchain Implementation ......................................................................................................... 17
  2.3 Smart Contracts .................................................................................................................................. 18
    2.3.1 Ethereum .................................................................................................................................... 18
    2.3.2 Smart Contracts on Ethereum ................................................................................................... 19

Chapter 3 Background Literature ............................................................................................................. 21
  3.1 Vaccine Supply Chain (VSC) .............................................................................................................. 21
  3.2 Last-mile in VSC ............................................................................................................................... 23
  3.3 Benefits and challenges of traceability ......................................................................................... 25
  3.4 Technology in VSC .......................................................................................................................... 27
  3.5 Literature on Blockchain .................................................................................................................. 29

Chapter 4 Methodology and Analysis ..................................................................................................... 32
  4.1 Blockchain based traceability ........................................................................................................... 33
4.1.1 System Setup.................................................................................................................. 34
  4.1.1.1 Ethereum Virtual Machine ................................................................................... 35
  4.1.1.2 Remix ...................................................................................................................... 36
  4.1.1.3 Ganache ................................................................................................................ 36
  4.1.1.4 Metamask................................................................................................................ 38

4.1.2 Smart Contract.............................................................................................................. 41

Chapter 5 Conclusions and Future Work............................................................................... 47

Bibliography ......................................................................................................................... 51

Appendix ................................................................................................................................. 58
LIST OF FIGURES

Figure 2-1: Operation Warp Speed Vaccine Delivery Milestones (U.S. Department of Defense. Figure available at https://www.defense.gov/Explore/Spotlight/Coronavirus/Operation-Warp-Speed/) 3

Figure 2-2: Operation Warp Speed Vaccine distribution graphic (U.S. Department of Defense. Figure available at https://www.defense.gov/Explore/Spotlight/Coronavirus/Operation-Warp-Speed/) 5

Figure 2-3: Records maintained for vaccines. (Yong, et al. 2020) 6

Figure 2-4: Phased approach of vaccine distribution (From the Factory to the Frontlines: The Operation Warp Speed Strategy for Distributing a Covid-19 Vaccine., 2020. https://www.hsdl.org/?view&did=844253) 7

Figure 2-5: Phased approach to Vaccine allocation for COVID 19 (National Academies of Sciences, Engineering and Medicine, 2020) 8

Figure 2-6: Structure of a Blockchain (Patel, et al., 2017) 11

Figure 2-7: Transaction secured by Digital Signature (Lemieux, 2017) 14

Figure 2-8: Merkle Tree 15

Figure 2-9: Example of Smart Contract input & output (Lin et al., 2020) 20

Figure 4-1: COVID-19 vaccine coverage index by Surgo Ventures 33

Figure 4-2: Blockchain based Vaccine traceability system 34

Figure 4-3: Ganache blockchain with accounts having public and private keys 37

Figure 4-4: Connecting Ganache network with Metamask wallet using URL 38

Figure 4-5: Import an account from Ganache network with seed phrase 39

Figure 4-7: Importing other Ganache account using private key (step 2) 40
Figure 4-6: Importing other Ganache accounts (step1) 40

Figure 4-8: Activity diagram for Vaccine Supply chain smart contract 41

Table 4-1. Private Blockchain network for Vaccine Supply Chain 42

Figure 4-9: Registering the entities of Supply Chain using account ID 43

Figure 4-10: Data recorded at vaccine manufacturing stage 44

Figure 4-11: Data recorded by Distributer at procurement stage 45

Figure 4-12: Data recorded by Clinics at Receival 46

Figure 5-1: Runtime error showing that the system expected Receival data record 47

Figure 5-2: Runtime error prohibiting an entity to execute an event when it is not the owner of the asset 48

Figure 5-3: Visibility in to manufacturing data captured by blockchain 49

Figure 5-4: End-to-end data captured by the blockchain 49
LIST OF TABLES

Table 1-1. Structure of Thesis ........................................................................................................3
Table 4-1. Private Blockchain network for Vaccine Supply Chain .............................................41
ACKNOWLEDGEMENTS

Firstly, I would like to sincerely thank my thesis advisor Dr. Soundar Kumara, for supporting and guiding me throughout my thesis program. He always directed me towards the right path and with his care and support, I have learned to be a better student and a better person.

I would also like to thank Dr. Saurabh Basu, for taking the time to read and review my thesis and provide valuable feedback.

Finally, I would like to express my profound gratitude to my parents Mr. Piyush Goda and Mrs. Nipa Goda, my sister Mrs. Ashni Goda Sanghavi for their unconditional love and support throughout my academic study; and my friends who supported and encouraged me in this journey.
Chapter 1
Introduction

1.1 Motivation

People get vaccinated with a belief that it will prevent any kind of flu or diseases in the future. The idea of vaccines is to immunize people. However, if not maintained at the prescribed temperature and if not administered within the given timeframe, it can have adverse effects on the health of the same people who took them hoping to avoid ill health. Hence, it is extremely important that the required temperature and time control is maintained at all the nodes in the end-to-end supply chain of the vaccine. Another important effort is to make sure that the product changes hands minimum number of times in the supply chain.

We consider the COVID-19 emergency deployment in this thesis. In this use case, it was also seen that the distribution of vaccines that would be required would be spread geographically across the globe. This also leads to stretched oversight capacities of the authorities and fragmentated supply chains. In such a situation, it becomes imperative to have an effective visibility over the supply chain to reduce the risk of falsified, counterfeited and substandard products to maintain public confidence, reduce vaccine hesitancy and achieve herd immunity quickly.

1.2 Problem definition

This thesis aims to solve the problem of provenance in vaccine supply chain which will ultimately help in achieving greater sense of accountability. It aims to address 3 main questions in
end-to-end vaccine supply chain through blockchain application. For a given vaccine administered to a patient these relate to manufacturing, logistics, and hospital as described below:

- Who is the Manufacturer? Which location/plant was the vaccine manufactured? At what temperature and time was it packed and shipped?

- Who was the distributor/logistics provider? When and at which distribution center was it stored at? At what temperature was it stored in cold storage and when was it dispatched?

- Which clinic/hospital was it shipped to and when was it received? At what temperature was it stored and when was it administered to a patient?

Having the responses about all these queries in a system will help achieve end-to-end visibility, greater accountability and reduce counterfeiting or any malpractices in manufacturing.

### 1.3 Brief description of the methodology

To answer the questions described above in the problem statement, a replication of real-life scenario was achieved virtually. Different blockchain accounts for players in the supply chain (Manufacturer, Distributer, Clinic and Patient) were obtained using Ganache (https://www.trufflesuite.com/ganache), their wallets were maintained using Metamask (https://metamask.io/) and the back end of the smart contract coded in Solidity (https://soliditylang.org/) as described in the next section was executed using Remix IDE (https://remix.ethereum.org/).

The procedure followed for the development of the smart contract involved defining the variables that changed throughout the process in end-to-end supply chain, nodes of the network,
the functions which enabled state changes and functions that enabled the output which gave data on provenance. This procedure is described in detail in chapter 4 of the thesis.

1.4 Results

The vaccine (asset) moves from the manufacturer to the patient sequentially according to the smart contract and doesn’t allow an unauthorized entity to execute a command and also doesn’t allow the data to be recorded out of order. The data recorded at each node can be called and viewed by the authorized personnel along with the timestamp and temperature. The system helps answer all the queries related to Manufacturers, Distributers and Clinic and ensures that accountability is maintained within the system, thus avoiding counterfeited and substandard products.

1.5 Organization of Thesis

Chapter 2 gives a background about Vaccine Supply Chains, existing practices in the industry and also explains the functioning of a Blockchain and its core principles in detail. It also gives a brief idea about the platform which is used to implement a blockchain network for vaccine supply chain. A literature survey about existing work in vaccine supply chains, traceability in various supply chains and applications and use cases of blockchain is reported in Chapter 3. Chapter 4 describes the methodology, tools and programming logic used to develop a blockchain platform for vaccine supply chain. It also discusses the analysis and results of the queries this thesis aims to answer. Finally, Chapter 5 concludes the thesis with a summary of possible future work.
Chapter 2

Background

2.1 Vaccine Supply Chain

Just like any other Supply Chain network, in a Vaccine Supply Chain network, different enterprises may be involved in Procurement, Manufacturing, Distribution and Transportation. A vaccine production company has a twofold responsibility in its operations. While maintaining high efficiency and lower costs; it also has a social responsibility which other companies usually don’t have (Yong et al., 2020). It’s production processes and operations are closely monitored by the State Food and Drug Administration (SFDA) to make sure that the Good Manufacturing Practice (GMP) guidelines are followed throughout the process. GMP guidelines highlight the good manufacturing practices that are prescribed by regulatory authorities including SFDA and WHO (Milstien et al., 2014). GMP guidelines are directed at not only the manufacturing companies but also the National Regulatory Authorities (NRA) so that the production, labelling, lot release, distribution and quality standards are regulated (World Health Organization, 1996).

According to Yong et al., 2020 a vaccine supply chain can be further simplified in to three chains to understand the manufacturing, testing and deployment of the vaccine. They term these chains as the GMP chain, the Release chain and the Inoculation chain. These three chains are supervised by three different institutions which are the vaccine enterprises, the lot release agency and the CDC respectively. Before being injected to the patients, the vaccine moves through these three chains sequentially.

- Authorization
- Prioritization
- Allocation
- Distribution

In the time leading up to the distribution of the vaccine, it is extremely important for all the departments in the federal and state government to coordinate and plan the equitable delivery of the vaccines to all the people until the last mile. Figure 2-1 illustrates the steps taken from approval to vaccine administration for the COVID-19 vaccine according to the Operation Warp Speed.
2.1.1 Vaccine Manufacturing and Distribution

In case a vaccine accident occurs, it is important to trace the cause of the accident all the way upstream to the enterprise providing the raw materials for the manufacturing of the vaccines. Since the chain is time and temperature sensitive, provenance is of utmost importance in a vaccine supply chain.

A Vaccine supply chain can get very granular depending on the compounds and chemicals used for manufacturing. U.S Department of Health and Human Services (https://www.hhs.gov/immunization/basics/types/index.html) suggest that there are different types of vaccines depending on various criteria like the immune system response, technology used to
manufacture the vaccine, etc. A non-exhaustive list of different types of vaccines as given on the HHS website (https://www.hhs.gov/immunization/basics/types/index.html) is listed below:

- Inactivated vaccines
- Live-attenuated vaccines
- Messenger RNA (mRNA) vaccines
- Subunit, recombinant, polysaccharide, and conjugate vaccines
- Toxoid vaccines
- Viral vector vaccines

Irrespective of the type of vaccine in question, the manufacturing of a vaccine involves procurement of multiple processed raw materials. After the raw material is processed to make a vaccine, it is stored in temperature-controlled warehouses and then distributed to clinics and hospitals, from where the patient/end user can access it. During emergency situations of demanding a fast and effective rollout of the vaccine, a centralized distribution approach can also be taken up. Operation Warp Speed is an example of such an approach where the distribution of COVID-19 vaccines on the basis of 4 tenets of “Visibility, Coverage, Uptake and Traceability” was undertaken (https://www.defense.gov/Explore/Spotlight/Coronavirus/Operation-Warp-Speed/). A centralized distribution strategy through the existing contractor Mckesson was chosen (U.S. Department of Health and Human Services, From the Factory to the Frontlines: The Operation Warp Speed Strategy for Distributing a Covid-19 Vaccine., 2020). The layout given in figure 2-2 shows the flow of vaccine through such a supply chain.
Designing a traceability system requires to initially understand the parameters that need to be tracked in order to maintain the quality of the vaccines and help restrict distribution of counterfeiting and substandard products. According to the GMP guidelines, the manufacturing companies need to record the information about the product, its raw materials and production up until distribution (World Health Organization, 1996). Yong, et al., (2020) define these as batch packing and batch production records. Figure 2-3 reproduces the figure from (Yong et al., 2020).
2.1.2 Vaccine Approval and Lot release

As discussed earlier, the vaccine production process is closely monitored. Along with that, even the vaccine development, approval and the process after approval is closely monitored. This can be attributed to the fact that vaccines are not only considered as drugs but also as a biological product by the Center for Biological Evaluation and Research (CBER) of the FDA (Marshall and Baylor, 2011). The approval process for a vaccine goes through a phased approach. Once the vaccine is approved by FDA, it can be open to distribution to the population according to the strategy decided by U. S. Department of Health and Human Services (HHS).
National Academies of Science, Engineering and Medicine is a group of highly qualified individuals representing a diverse population. This board provides a prioritization schema to the CDC Advisory Committee on Immunization Practices (ACIP) (Walter and Moody, 2021). Upon reviewing and approving the vaccine priority plan, the ACIP recommends the plan to the HHS which approves the plan for allocation.

Based on the ethical principles of “Maximum Benefit, Equal concern and Mitigation of Health inequities” and the Procedural principles of “Fairness, Transparency and Evidence-based”; the National Academies define the prioritization of the vaccine distribution. Initially a portion of the population is prioritized over the others because of a limited supply of vaccines and as the supply increases, the vaccine distribution is opened to other people in a phased approach (National Academies of Sciences, Engineering and Medicine, 2020). The graph shown in figure 2-4 shows why it is important to have a phased approach in distribution of vaccines.

![Figure 2-4: Phased approach of vaccine distribution](https://www.hsdl.org/?view&did=844253)
Within each phase of allocation, all groups have equal priority. Figure 2-5, given below shows the recommendation of National Academies of Science, Engineering and Medicine for the allocation of vaccine.

![Figure 2-5: Phased approach to Vaccine allocation for COVID-19 (National Academies of Sciences, Engineering and Medicine, 2020)](image)

To summarize, the traditional and prevalent systems of vaccine manufacturing, distribution and monitoring after the approval have three significant flaws. First, records are at a risk of counterfeiting, posing the risk of forgery; second, the system is too large to be maintained on a national scale; and third, vaccine information is vulnerable to tampering or deletion.

### 2.1.3 Vaccine Supply Chain in COVID-19

The supply chain of a vaccine during a pandemic is very different from the supply chain of other vaccines in any other scenario. Considering that the development of the COVID-19 vaccine was done in an unprecedented timeframe of under 2 years as compared to the traditional vaccine
development timeframe of 10 years, it becomes important to make sure that the manufacturing and equitable distribution of that vaccine to the people is done at scale with speed as well (Le et al., 2020).

To achieve this expedited production and distribution, the COVID-19 vaccine supply chain had to be a lot different than the other vaccine supply chains. The primary purpose in the pandemic situation was to reduce the transmission of disease and achieve herd immunity as soon as possible. The government directly procures the vaccine for distribution instead of going through the traditional channels involving distributors and retailers (Alam et al., 2021); as discussed earlier and as shown in figure 2-2. Israel was able to achieve the fastest rate vaccination at the end of 2020 because of its favorable geographical factors, centralized national system of government, it’s IT and logistics capabilities and experience in implementing large-scale national emergencies among others (Rosen et al., 2021).

In a region with larger land coverage, it becomes harder and more complex to distribute the vaccines; but coordination and planning can make the process of distribution seamless. Irwin (2021) reported that the production of a vaccine requires hundreds of individual components that are manufactured at various locations internationally. The procurement and production in case of such a requirement needs to be highly coordinated. Competing companies had to cooperate and work together to manufacture vaccine doses required to reach herd immunity (Irwin, 2021).

 Various studies have been done to identify the number of doses that need to be manufactured to reach herd immunity. Randolph and Barreiro (2020) established in their research that the herd immunity for COVID-19 will be 67%. Rele (2021) and Irwin (2021) heuristically calculated the demand of vaccine doses to be around 10-12 billion doses globally to reach herd immunity. However, the securing of 6 billion doses out of 8.6 billion doses in March 2021 by high and upper-middle income
countries will cause a complication in this calculation. It is important to make sure that the coordination and cooperation among the companies across the world is made easier and promoted with technology that will mitigate the problems that might arise later due to mistrust and lack of transparency.

The last stage of the supply chain is to make sure that the vaccine reaches the final patient. This is equivalent to the last mile delivery of products in some other industries. Effective communication about the details of the vaccine helps establish trust and encourages more people to get vaccinated early. Even if the vaccine production is very high, any lack of transparency around the vaccine may lead to vaccine hesitancy among the people (Thiagarajan, 2021).

2.2 Blockchain

Blockchain is a distributed ledger that allows untrusted multiple stakeholders in a shared network to exchange transactions or confidential data (Lin, et al., 2020) The transactions are stored in ledgers called as blocks and linked in the form of a chain. Since the data is linked in the form of a chain, tampering with the data in one block will not suffice. If someone wants to manipulate the data, it will have to be manipulated in the ledgers maintained by all the nodes and change all the blocks before the block where data is manipulated. The blocks are updated regularly. The authenticity of a transaction initiated by one node is validated by a majority of the other nodes through a consensus mechanism and then the time-stamped information is added in a new block which is then linked to the existing chain of blocks making it immutable. All of this takes place without the intervention of a central authority; hence it is a decentralized system. Figure 2-6 shows the structure of a basic block chain network as reproduced from Patel et al. (2017).
2.2.1 Blockchain Components

To understand how a blockchain network works, it is important to understand many essential techniques including hashing, cryptography, digital signature, consensus mechanisms, etc. which are used to create a blockchain network. In this section, a brief description of the essential components required to build a blockchain network is undertaken.

2.2.1.1 Hash function

Hash functions are one of the important concepts in modern day cryptography. Hash function is a deterministic function that can be used to map arbitrarily sized data to hash values of fixed size. No two hash functions give the same hash value and it is very difficult to restore the original data from its hash value. Message digest 5 (MD5) and SHA256 are the two most popular hash functions used in blockchain, depending on the complexity of the data (Rachmawati et al., 2020)
Hash functions are very helpful to verify the integrity of data in case of transmission between 2 nodes. An encrypted hash function can be sent with the data to the receiver node. The receiver can authenticate the integrity of the data by calculating the hash value of the data received using the same hash function. If the hash value that is generated and the hash value that is received are identical, then it can be verified that the data was received without any corruption.

2.2.1.2 Asymmetric Cryptography

In blockchain implementation, symmetric cryptography and asymmetric cryptography are the two techniques that are used to transmit encrypted data in the network. In symmetric cryptography, the same key is used to encrypt and decrypt data. However, Asymmetric cryptography is used in conjunction with a hash function to enforce digital signature strategy in distributed systems to implement verifiable transactions (Kumar et al., 2011) Asymmetric encryption is a process in which two keys, a public and a private key, are used to construct a key pair. Each user in the network has a key pair where the private key of a user is to be known only by the respective user and the public key of all users is known by every user in the network. The public key of a user can be derived from the private key, but the inverse is not true. To transmit data confidentially to only the intended user, the public key of that user can be used to encrypt the data. Thus, the receiver of data can use his/her private key to ensure that confidentiality was maintained (Bednarek, 2019)

2.2.1.3 Digital Signature

A digital signature is required to identify if a certain individual initiated the transaction in question (Lozupone, 2018). Just like a real signature is used to test the authenticity of a document,
a legitimate digital signature certifies that a transaction was initiated by a known source. Once the transaction is conducted, it is encrypted using the sender’s digital signature. The asymmetric cryptography technique guarantees verification of this sign because only the sender has his own private key. Furthermore, if the sender doesn’t want anyone to know that a transaction is sent to the receiver apart from the receiver itself, the public key of the receiver can be used to direct the transaction. The receiving party verifies the signature by using the public key that corresponds to the private key that signed the original letter, which is then submitted along with the message. If the signatures fit, it's fair to say the data signed with the private key hasn't been tampered with (Bednarek, 2019). Since it tracks ownership of messages used to commit transactions to the blockchain, digital signature is one of the most important features to maintain accountability in blockchain implementations. Figure 2-7 shows an example of a transaction with digital signature. (Lemieux, 2017)
Merkle trees are a key component of how blockchains function. While there is no restriction on having a blockchain network without Merkle trees; as the number of transactions increase, it becomes computationally very expensive for computer systems with lesser power and storage to efficiently retrieve older transactions and also to add new transactions (Patel et al., 2017). As more and more transactions are linked to the chain in the form of blocks, the memory consumption to store all the transactions keep increasing. Merkle trees break this large amount of data into subsets and hash it. This procedure is repeated until only the root node remains.
The most common type of Merkle tree is the binary Merkle tree which consists of leaf hash nodes, intermediate hash nodes and root hash node. Figure 2-8 shows a basic example of a Merkle tree with four transactions $T_A$, $T_B$, $T_C$, $T_D$.

2.2.1.5 Consensus mechanisms

Blockchain proposes to solve the Byzantine Generals Problem (Lamport, 1983) by using consensus mechanisms which protect the data by allotting random candidates from all nodes the responsibility of updating data blocks. There are many consensus mechanisms out of which the popular ones are Proof of Work (PoW), Proof of Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), and Proof of Elapse Time (PoET).

- **Proof of Work**: Out of the total number of digital cryptocurrencies, around 90% use Proof of Work as their consensus mechanism (Gervais et al., 2016). In PoW, all the participant nodes compete to solve a CPU-based computational puzzle. The node which solves the puzzle first gets to create the block of new transactions and append to the chain of existing blocks upon approval from all the other member nodes as well. This process is called
mining. However, this process is very inefficient in the consumption of energy because for
the energy utilized by all the nodes to solve the puzzle, only one winner gets the returns
(Zhao et al., 2019).

- **Proof of Stake:** As the rate of mining of cryptocurrencies back by PoW increases, the
  energy consumption by the miners also increases. King and Nadal (2012) formally
  proposed the concept of proof of stake to counter that. This mechanism provides an
  alternative to the very high energy consumption in PoW with a solution that is based on
  selecting the member node based on the idea of stake. The node with higher amount of
  stake i.e., one who has higher amount of currency has a higher probability of proposing the
  new block in the chain.

- **Practical Byzantine Fault Tolerance:** Castro and Liskov (1999) proposed an algorithm that
  was practical to work in asynchronous environments and also provided a solution to
  Byzantine faults which are described as a failure of communication. Practical Byzantine
  Fault Tolerance is an extension of this algorithm in blockchain which help in allowing to
  reach a consensus. According to this algorithm, consensus can be reached if the number of
  faulty nodes is less than one third of the total number of nodes. Nodes broadcast message
  amongst each other to validate the block proposed by one of the nodes. As compared to
  PoW where the node with higher computational power proposes the block and PoS where
  the node with higher stake proposes the block, in PBFT the block which is agreed upon by
  3f+1 node where f is the maximum number of faulty nodes gets approved to be added to
  the chain (Khullar, 2019).

- **Proof of Elapsed Time:** Intel came up with Proof of Elapsed Time which was based on its
  Software Guard Extensions (SGX) to decide the node that will propose the new block in
the chain (Ahmad et al., 2021). The node to propose the new block is decided based on the amount of waiting time given to them by the enclave which is trusted execution environment. The waiting time for each user is decided using a random distribution with a predefined mean. This also helps to reduce the very high amount of energy used in the PoW mechanism (Zhao et al., 2019).

2.2.2 Blockchain Implementation

There are several stages involved while recording a transaction to maintain the integrity of the data. Some of the nuances involved in this process change depending on the blockchain network used, consensus mechanism, etc. Given below are the stages involved in a general blockchain protocol:

- **Stage 1**: The sender sends a message which is hashed and sent with a digital signature to either the whole network or a particular user according to the use case.

- **Stage 2**: The transaction is verified by using the public key of the sender and checking the adherence to all the rules defined for the data in the network. This verification process is done by all the nodes in the network.

- **Stage 3**: After a predefined amount of time, all the transactions occurred in that time are collected and hashed together. This collection of the transactions is called as a block. Based on the decided consensus mechanism, some of which are discussed in the previous section, nodes work to append the block to the blockchain.

- **Stage 4**: Every consensus mechanism also has a process to verify the transactions. The purpose of this process is to make fraudulent transactions impossible and make sure that
all the transactions are valid. All the consensus mechanisms have their pros and cons and need to be chosen according to the application of the blockchain.

- **Stage 5**: Upon validation of the transactions by majority of the nodes in the network, the block is added to the blockchain. To maintain the integrity of the data, the chain structure also helps in a way that to add a block to the chain, the hash value of previous block is required. So, to tamper with a saved transaction, all the previous blocks in all the ledgers of all the nodes will have to be updated.

Using this process, all the transactions in a distributed peer-to-peer network can be stored in a decentralized manner sequentially. It provides a transparent system for all the stakeholders in the network while maintaining the integrity of the data.

### 2.3 Smart Contracts

#### 2.3.1 Ethereum

Ethereum is an open blockchain network that allows anyone to create and use decentralized applications (DApps) based on Smart contracts. The platform was created by Vitalik Buterin in 2013 to give developers a Turing-complete virtual machine that allowed them to create applications with the security of blockchain and its consensus mechanism without the need to create new blockchains every time they want to create an app (Buterin, 2016).

The applications created on Ethereum have the functionality of economic transactions as well. The transactions of finances can be done using ether which is the internal crypto-token used in Ethereum networks (Patel et al., 2017). During the development of DApps on Ethereum, the accounts including Externally owned Accounts (EOA) and Smart contracts are the states. In case
of an EOA, the state stores Ether as the account’s balance and in case of contract, the state stores the code and contract storage as a key-attribute database (Buterin, 2016).

2.3.2 Smart Contracts on Ethereum

Buterin (2016) defines a Smart contract as "a computer program that directly controls digital assets". Since a smart contract is stored in a state in Ethereum, it has its own memory along with the functions and transactions that it can perform like any other external user. As they have an address, they can also store digital assets. This means that for an asset in the possession of a contract can only be transferred according to the code of the contract. Thus, a smart contract can be said as a computerized business logic that helps in executing transactions where there is an existing trustworthiness. As the contract program is self-executed and immutable, two nodes can partake in a transaction without having to depend on trust as a factor (Patel et al., 2017). Once an asset is placed in the possession of a contract, it is automatically transferred to its final state according to the terms of the contract. Figure 2-9 shows an example of Smart contract (Lin et al., 2020) where it calls the ordering services to check the endorsements and connect the validated transactions to blockchain blocks after collecting all of the endorsements. No one will tamper with the data on the blockchain since the documents are timeless. DApps may also use the smart contract to query the status of accounts or purchases.
If the final state of a contract is another contract, then the next contract is initiated, and the asset is transferred in the possession of that contract. This can happen sequentially giving the external users freedom to execute multiple state changes at the same time without supervision (Hirai, 2017). Although supervision is not necessary, the asset can still be traced as the contract executes giving a user visibility into the asset throughout the process. Ethereum Virtual Machine (EVM) stores all the contracts in bytecode format and also executes the smart contracts. Ether is charged by the EVM when these smart contracts are executed. The contracts and the services they provide are handled by the network and remain in the network as long as the network remains or can be self-destroyed if they are programmed in that way (Hildenbrandt et al., 2018).
Chapter 3

Background Literature

In this chapter, research regarding the existing practices in medical products’ supply chain, different challenges and risks associated with it, benefits of traceability and existing technologies to achieve it are described in detail.

3.1 Vaccine Supply Chain (VSC)

It is extremely important for the supply chain of medical products to be resilient, robust and transparent. Timely delivery of drugs and vaccines is important to prevent large scale disruptions in case of seasonal outbreaks and to achieve herd immunity in case of pandemics and epidemics.

In the last few years, considerable research has been done to improve the vaccine supply chain. Westerink-Duijzer et al. (2019) have studied the need of central planner in case of an outbreak or a situation when the vaccine availability is scarce. When the availability of vaccines for an infectious disease is limited, it becomes important to prioritize and fairly allocate the vaccines amongst the affected population. They conclude that even though cooperation might lead to increased health benefits, a decentralized system when both the parties involved in the analyses agree to cooperate without a central planner like CDC or WHO can lead to a smarter solution.

Lemmens et al. (2016) examine general supply chain network design (SCND) models and research the literature outlining the strategic, tactical and operational criteria that has been considered while developing such models. The paper also highlights the key issues that are prevalent while designing supply chains networks of vaccines like allocation, location, shelf life
and cold chain distribution. Vaccine supply chains are different than other supply chains because factors such as the volume and temperature at which it must be stored, would, however, have a significant impact on the supply chain. The end-to-end supply chain of vaccines need to follow all the safety protocols and regulations in order to avoid any concerns of contamination of the final product. It is important to be vigilant in tracking the physical flow of the supply chain and ensure that all the safety standards are met.

Vaccine supply chains need to be evaluated beyond the economic criteria. Vandaele and Decouttere (2013) propose to evaluate the supply chains on economic, technological and value criteria which give a better indication of how responsive, flexible, cost-effective and also demand satisfying the vaccine supply chain network is. In case of a pandemic like COVID-19, it is important to evaluate the supply chain in terms of responsiveness and flexibility. Disruptions and Uncertainty can also adversely affect the supply chain and need to be considered when designing the SCND model (Lemmens et al., 2016). Having visibility in the network helps in such a case. Even the raw materials procured for the manufacturing of products and their origins need to be tested in order to make sure that they meet the necessary requirements (Peysson, 2010).

One of the other reasons why vaccines need to be monitored and regulated is because of the complex supply chain caused by globalization. Authorities need to make sure that the information and physical flow of vaccines is regulated to ensure a timely delivery to the patients. Before reaching the patients, vaccines go through a number of controlled processes and a number of organizations. It is often the case that the end-to-end process is done across multiple countries. This reduces the oversight capacity of national regulatory authorities or makes it more complex (World Health Organization, 2021). Blockchain technology can provide the solution to that problem as well.
3.2 Last-mile in VSC

As we move deeper into the vaccine supply chain, the oversight capacity decreases and complexity increases. Similarly, as opposed to early phases of drug distribution, there is limited regulation and auditing from different enforcement authorities and local boards of pharmacy (Sykes, 2018). Although 4-5 million deaths are avoided every year due to immunization, an estimated 19.7 million children under the age of one fail to receive basic vaccines (https://www.who.int/en/news-room/fact-sheets/detail/imunization-coverage).

Apart from the higher complexity caused due to door-to-door delivery of products, some reasons specific to vaccine last mile logistics make it more difficult. Firstly, vaccines require a highly controlled temperature. Once the vaccine is dispatched from the distribution center where it is stored at -70°C, it expires in 5 days when stored at 2-8°C (Rosen et al., 2021). It has to be transported to remote areas and areas with limited infrastructure because unlike other products where people pull the product through the supply chain, vaccines need to be pushed through the supply chain for the benefit of the people (Owolabi et al., 2017). Lastly, a considerable amount of the population refuse to take the vaccine or are hesitant before taking it (Saint-Victor, 2013).

Every citizen irrespective of the background needs to be provided a vaccine when a location needs to be immunized. One of the primary concerns in the logistics of vaccines delivery is that it needs to be supplied to areas with limited infrastructure or regions without an accurate map describing the settlements in the location as well (Barau et al., 2014). Geospatial last mile delivery solutions that are affordable and open-source is identified and tested to be one of the solutions in mitigating this problem of last mile delivery (Owolabi et al., 2017).

Some of the developing countries are not well-equipped with techniques to improve their last mile logistics and have little clarity on the guidelines and regulations that govern them.
(Bhatnagar et al., 2018). Technological solutions are widely explored to counter the problem of delivery to remote locations. Drone usage to deliver products to the end user where road infrastructure is limited is one of the solutions that is being widely researched. However, cost, temperature and the storage quantity still remains a concern in these solutions too (Wurbel, 2017).

Apart from the issues discussed above, one of the biggest concerns causing last mile delivery issues is vaccine hesitancy in the patients. All the efforts to deliver the vaccine to the patient will turn futile if the patient refuses to get immunized. Although the response to immunization from the people is different in case of disease eradication and in disease control, monitoring and surveillance is still considered as a solution to counter the vaccine refusal among the population (Saint-Victor et al.). Studies have shown that people refusing to take vaccines have similar reasons to do so in various geographical regions. Lack of trust and effectiveness is one of the main reasons given by people who are hesitant to get vaccines (Chevallier et al., 2021). Lee et al. (2021) suggested the use of communication programs, coordination among the local and central governments and swift vaccination of people willing to get vaccinated to mitigate the problem of last mile in the times of COVID-19. Although this might have some positive impact on the vaccine distribution, solving the problem of last mile still has ample opportunity for improvement (Turcotte et al., 2021).

Visibility in to the vaccine supply chain and knowing how and where the vaccine that is being delivered to them, was manufactured and transported may allow the people to trust the immunization process. Since the data recorded in the system can be viewed by the users before getting the vaccine, blockchain can help to tackle this issue of last mile by providing the transparency that the patients are looking for.
3.3 Benefits and challenges of traceability

Before understanding the benefits of traceability in the supply chain, it is important to understand why it is necessary. Counterfieting and substandard products is one of the reasons why traceability is important. Tremblay (2013) explains that there are four main reasons why counterfeited products could be distributed:

- **Regulatory Incentives:** A corrupt and lethargic enforcement system of regulations in the medical industry leads to opportunity because of slacked detection strategies.

- **Pricing incentives:** The difference between prices for the products across borders causes a gap which can be filled by unfair means.

- **Information incentive:** If the methods used to differentiate between a legitimate product and a counterfieted product is easy to copy, then it creates an opportunity to use products of lower quality. A realtime visibility in to the supply chain of the product can help mitigate this problem.

- **Consumer incentive:** Patients with conditions that they are ashamed of or patients with a poor background look for products through concealed means and at lower prices. This creates an opportunity that can be exploited to expand the market for counterfeited products.

These markets for counterfeited vaccines have caused incidents in the past when such vaccines were distributed largely in the same markets. In 2020, falsified Rabies vaccines were falsely distributed largely in Phillipines from multinational companies ([www.who.int/news/item/31-01-2020-medical-product-alert-n-8-2019-(english-version)]).

Similarly in 2018, two chinese vaccine makers were found to have fabricated manufacturing
record. This resulted in the administration of more than 600,000 substandard Diphtheria, Pertussis, and Tetanus (DPT) vaccines to the kids (The Lancet, 2018).

Such incidents lead to a lot of negative health consequences. One of the main outcomes of the distribution of substandard and counterfeited vaccines is that in case of an outbreak, population might get a false sense of security because of the administration of these vaccines. This not only fails the efforts to reach the required herd immunity, it catalyses the transmission of the disease. Finally, the population tends to lose confidence in the vaccines and it also affects the credibility of the manufacturers and the regulatory authorities (Jarrett et al., 2020).

Traceability plays an important role in preventing these issues. Current measures include using Global Trade Item Number (GTIN) to identify different vaccines uniquely based on the manufacturer, lot number and expiry date (https://www.gs1.org/standards/id-keys/gtin). However, it is really important to have real-time visibility in to other data as well for safety monitoring and pharmacovigilance.

Apart from counterfeiting, improper maintainence of temperature and time is another important challenge in today’s vaccine supply chain. Vaccines kept at incorrect temperatures or stored beyond their shelf-life not only can render the vaccine ineffective but also can cause ill-effects to the health in some situations. In 2015, Pharmacy & Medical Packaging News published a poll of supply chain experts and came up with the following results: 51 percent of temperature-sensitive items delivered were at room temperature, 31% were refrigerated, 17% were frozen, and 32% should not be able to freeze (Sykes, 2018).

According to WHO’s policy paper on Traceability of Medical Products (2021), through real-time monitoring of a product and an effective implementation of a traceability scheme, supply
chain performance and accountability can be further improved. WHO lists the following as benefits of an efficient traceability system for a medical products’ supply chain:

- Circulation of only authorized products in the legal supply chain
- Falsified, expired, recalled products’ distribution is prevented
- Fast and efficient market recalls
- Efficient inventory management at all levels of supply chain
- Identifying and monitoring the shortages and stockouts

3.4 Technology in VSC

Technology can play an important part in achieving traceability and increasing accountability in vaccine supply chains. There is a general consensus that with the help of technology in VSC, the integrity in supply chain can be maintained. Counterfeiting and Expired Products in Vaccine supply chain can lead to critical effects on the patients which consequently leads to effects on the health and economy of the society. Currently, the technology in VSC is highly focused on Mobile technologies, Radio Frequency Identification (RFID) and barcoding. Other technologies like Machine Learning and Blockchain have not been explored significantly. Existing technologies have helped in enhancing product traceability to a certain extent and reduced counterfeiting in the vaccine supply chain.

Mackey and Nayyar (2017) reviewed the emerging technologies to combat the trade of fake and substandard medical products. They identified five different categories of ‘digital’ technology solutions in their review which included mobile technologies, RFID-based solutions, Advanced Computation solutions which involved the use of machine learning and big data, Online
pharmacy verification solutions and a brief description of the scope of blockchain technology to encounter this problem.

Since most developed and developing countries have a very high reach of mobile phones, mobile driven solutions. These solutions use the capabilities of built-in sensors, cameras, GPS, etc. Most of the mobile based solutions are offered as support services to other solutions described below (Isah, 2012). Internet of Things (IoT) is another such technology that can be used to have visibility across the supply chain. A smart infrastructure can be used to collect data, products and track it across the whole network. A framework to build such a smart system for the Supply Chain Management (SCM) using IoT is provided by Abdel-Basset et al, (2018).

To implement the IoT system, a vaccine traceability management system based on radio frequency identification technology (RFID) records of the entire process details of vaccine development, output, storage, and vaccination can be used. Electronic tags are used to monitor the entire vaccine development, output, storage, vaccination, and other information. Users may also use the interface to query specific vaccine information that can be tracked back to the function for validity and safety of the vaccines (Wang et al. 2018)

With the use of Machine Learning and Artificial intelligence in many different fields in manufacturing, supply chain, advertising and even healthcare, some research has been done in utilizing ‘big data’ and advanced computational algorithms to sift through large amounts of records and identify patterns and irregularities which could prove helpful in detecting and mitigating the problem of counterfeiting and drug supply chain incursions. Studies have been conducted to enhance the digital drug safety using social media data like that of Twitter (Freifeld et al. 2014)
3.5 Literature on Blockchain

Another emerging technology that is finding its feet in a variety of fields including banking, finance, healthcare, supply chain, etc. is the Blockchain technology (Nakamoto 2008). It has gained a lot of attention in different fields apart from cryptocurrency. Healthcare is one of the fields that is envisioned to have a use case for healthcare where patient records and their privacy is of great importance. Updates to the records for every patient like appointment scheduling, medication changes, prescription, etc. are assigned a timestamp and are immutable (Angraal et al., 2017)

Lin, et al. (2020) explore the implementation of Blockchain in Agricultural systems. The produce grown in farms changes a lot of hands before it reaches the table of people as food. Although it is important to have the food processed according to its requirements, it is also important to eliminate nodes in this network that add additional costs to the whole system. It is important to establish a tradeoff between efficiency and trust in this system. For this purpose, a number of platforms including their advantages and disadvantages are reviewed in the paper.

Monetary transactions is a well-established use case of blockchain technology. It proposes a decentralized system which helps mitigate the issues caused by lack of trust when a third-party intermediary is involved. Insurance industry is another field where the use of blockchain is identified to mitigate the problems caused by its slow claims process and high manual work. High amount of paperwork and long time required for the claims process leads to loss of data and capital for either of the party involved in the transaction. Blockchain’s smart contracts offer a reliable and trustworthy solution to this industry (Ikeda and Hamid, 2018)
The application of blockchain in various parts of supply chain have also been researched. It has application in Engineering Design because it provides a single platform and easy verification by certifying agents, thereby accelerating the process. Smart contracts can help maintain data of all terms and conditions agreed with various partners. Integration with ERP can help solve disputes seamlessly. Safe and optimal sharing of documents and information (EDI, excel, etc.) and improved sourcing because of transparency are among other benefits. It also helps in smoother operations in Manufacturing and Logistics because of better visibility (Banerjee, 2018).

Procurement problem is an important problem in the supply chain that leads to increase in the end-to-end costs across the network. 4PL in the logistics system leads to an overall increase in supply chain cost, errors and lead time in the long term because of collusion and centralized control of 4PL. Blockchain and smart contracts eliminate these middlemen in the network and connect vendors and retailers directly with the option for 3PL to directly get in touch with the transactions through a single portal. The study explores the use of the concept in the megacity logistics and shows its advantages (Polim et al., 2017).

Food supply chain is another use case where blockchain can provide solutions to the problem of visibility and trust. Existing current architecture has central database collecting information of the characteristics the food products after it reaches the distributor and subsequently to the retailer. However, the information regarding products upstream is maintained as database silos. Author suggests the use of RFID, GPS, Sensors, etc. in the upstream supply chain to have more accurate data and use that in the blockchain network for transparency across multiple nodes in the network with immutable entries preventing any malicious activities (Mohan, 2018).

Vaccines being a product that can have fatal consequences on a patient if they are substandard or expired, it is extremely important that they are monitored. As seen in the recent
times of COVID-19, vaccination is important to maintain public health and ultimately, it also influences the economic and social aspects of the society. Vaccine supply chains, however, continue to be plagued by problems which lead to a loss of credibility of the regulating agencies and production companies, leading to vaccine hesitancy. Thus, it is important to establish trust between different stages of vaccine supply chain by providing immutability, scalability and universal access to information. As studied in various studies, with sufficient funding and more research, blockchain can provide that opportunity. This research aims to explore that solution.
Chapter 4
Methodology and Analysis

Background literature demonstrates that even though there are IoT systems in place to track the transportation conditions and the product, lack of trust within the system does not allow the sharing of this data in the end-to-end network. Blockchain provides the best opportunity for product provenance, especially in a vaccine supply chain where we need to know everything about the product.

In the unprecedented times of a global pandemic as that of COVID-19, it becomes extremely important to make sure that maximum number of people get vaccinated as soon as a vaccine is approved. The COVID-19 Vaccine Coverage Index (CVAC) by Surgo ventures show that various communities and various counties encounter different challenges. Some counties have a historic under vaccination rate, while some have sociodemographic barriers. Apart from these 2 themes, the CVAC covers 3 other themes viz. Resource-constrained healthcare system, Healthcare accessibility barriers and Irregular care seeking behaviors. Figure 4-1 shows the CVAC across all the counties in USA.

It is clear that end-to-end visibility in a vaccine supply chain is very important not only for the operational purposes of all the parties involved in the supply chain, but also from the point of view of consumption. This kind of visibility can help tackle the community barriers and allow the vaccination quickly, safely and equitably.
Since there is no one size fits for all when it comes to vaccine rollout, targeted programs for communities and appropriate resource allocation can be taken up.

4.1 Blockchain based traceability

As discussed previously in the literature review, a vaccine supply chain involves various players from manufacturers to the end consumer. This section aims to provide a proof-of-concept study of the use of blockchain to improve product provenance and curb counterfeiting.

Figure 4-2 tries to show all the nodes involved in a vaccine supply chain network through which the physical flow of product takes place. Out of the network shown below, this study covers the network from manufacturer to the patient for the sake of simplicity in demonstration. Additional nodes can be added to the network following the same basic structure used for the existing nodes. As it can be seen, data is collected at each node through the use of IoT.
traceability systems are already using barcoding as a part of effort to improve traceability in this era of globalization. This process along with additional RFID resources can be used in the blockchain implementation as well.

The existing barcoding system did not allow transparency throughout the network because of lack of mutual trust between the interacting parties in the network. Blockchain system gives an opportunity to achieve high level of visibility along with an increased sense of trust.

4.1.1 System Setup

For the purpose of this study, a test blockchain network was used in a local environment due to lack of resources to deploy smart contracts on the Ethereum main network. Creation of smart contracts and deploying them on Ethereum Virtual Machine (EVM) requires a setup of various other tools. These tools help create a test environment that replicates the production environment of Ethereum Main network, thus helping in the proof-of-concept study for this thesis.
The programming language used for the purpose of this thesis is Solidity which is a Turing complete language.

4.1.1.1 Ethereum Virtual Machine

Ethereum is a protocol for running a virtual machine called the Ethereum Virtual Machine (EVM) in a distributed and transparent environment. Ethereum smart contracts are the public and immutable programs that are run on EVM (Hildenbrandt et al., 2018).

When the code is not null, the account is controlled by the code; this type of account is referred to as a contract. Since a smart contract is stored in a state in Ethereum, it has its own memory along with the functions and transactions that it can perform like any other external user. The account is otherwise managed by the owner of the private key that corresponds to the address of an external account. The execution of a contract requires computational expenses that require allocation of resources from EVM. These fee or charge taken by EVM for the use of these resources is called Gas. Gas is charged in small fragments of Ether (Eth) called as “gwei”. The amount of gas used for a contract changes according to the resources used by EVM.

An external account will either create a contract or call an account to start a transaction. EVM's entire state transformation is deterministic until a transaction is started. As an account contacts another account, the transferred balance, gas, and data are sent along with the call. A quick balance shift occurs when the called account is an external account. If the called account is a contract, the called contract's code is executed after the balance transition. The executing account's storage can be changed by the code execution. The execution can read the balances and codes of all accounts. The amount of gas paid by the initiating external account limits the code execution's resource usage. A certain amount of gas is used while an order is executed. The
execution stalls when the gas runs out (out-of-gas failure). Except for gas intake, such errors revert all state changes made during the current call.

4.1.1.2 Remix

Remix is an integrated development environment which has open-source web and desktop application. Through Remix, you have 3 environments for executing the transactions:

- **JavaScript VM**: To simulate a real blockchain, a sandbox blockchain created in the browser using JavaScript.
- **Injected Web3**: a web3 injector, such as Mist or Metamask, that connects you to your own private blockchain.
- **Web3 Provider**: a remote Ethereum node using geth, parity, or some other Ethereum client. Can be used to bind directly to the actual network or the private blockchain without passing via Metamask.

This study uses the Injected Web3 environment of Remix IDE along with the injector of Metamask and a private blockchain of Ganache.

4.1.1.3 Ganache

As discussed earlier, Remix by itself cannot create real user accounts to transact amongst them. It also provides very less control over the configuration of the blockchain created by itself. Lastly, it is very difficult to monitor the execution of transaction by using just remix. Ganache provides an opportunity to do that.
Ganache helps in setting up a personal Ethereum blockchain in order to test the smart contracts in a near real world scenario. As shown in the figure 4-3, it provides a list of 10 accounts which make up your personal blockchain. Out of these 10 accounts, accounts can be assigned to 2 Manufacturers, 2 Distributers, 2 Clinics, 1 Patient and 1 admin.

As it can be seen, these accounts come with their respective public and private keys which can be accessed by clicking the key symbol on the right. These accounts come with a balance of 100 ethers and a very high gas limit for test purposes. Having a balance of fake ether helps in replicating the real-world scenario.

Figure 4-3: Ganache blockchain with accounts having public and private keys

Another great feature that Ganache provides is that of tracking the transactions in a separate tab. Along with that, every account’s involvement in total number of transactions can also be seen across the transactions. Finally, the blocks formed and added can also be tracked right from the
genesis block. This level of visibility in the testing phase helps in debugging and analyzing the smart contracts before deploying.

4.1.1.4 Metamask

Metamask offers services as a cryptocurrency wallet to interact with the Ethereum blockchain for decentralized applications. All the accounts in the personal blockchain obtained using Ganache can also be linked with Metamask for the purposes of testing. Thus, in an injected Web3 environment, different accounts from the personal blockchain of Ganache act as nodes in the Supply chain network.

This helps in replicating a real-life scenario where different manufacturers, distributors, etc. have their own accounts on blockchain. Thus, to execute a transaction, it is important to switch the account on Metamask to the respective node’s account. This also helps because Metamask

![Image of Metamask settings]

Figure 4-4: Connecting Ganache network with Metamask wallet using URL
maintains a separate wallet and transaction history for each account. Thus, the feature of financial transaction transfer can be tested using Metamask.

To add the other 9 accounts from Ganache on Metamask wallet, it is important to use the private key of the account. Figure 4-6 and 4-7 shows how account 6 was added to the Metamask wallet using the ‘Import Account’ feature.

To summarize the above description of the tools used, a replication a real-life scenario was achieved. Different blockchain accounts for players in the supply chain were obtained using Ganache, their wallets were maintained using Metamask and the back end of the smart contract coded in Solidity as described in the next section was executed using Remix IDE. Given below is the procedure to link a private blockchain network of Ganache with Remix IDE and set up its wallet on Metamask.
Figure 4-6: Importing other Ganache accounts (step 1)

Figure 4-7: Importing other Ganache account using private key (step 2)
4.1.2 Smart Contract

Because smart contracts are becoming more popular, their security is being scrutinized more closely. Bugs in these contracts can be financially disastrous for the parties concerned. The DAO hack, which resulted in the theft of 150 million dollars’ worth of Ether and an unprecedented hard fork of the Ethereum blockchain, is an example of such a disaster. DAO is a Decentralized Autonomous Organization which is represented by smart contracts in a transparent way. It is controlled by the members of the organization and not influenced by any central authority. Worse, the DAO is one of several smart contracts that did not execute as planned, resulting in significant losses. Hence, it is extremely important to make sure that the code for smart contracts is debugged and tested on test networks of blockchain before deploying to mainframe networks. Thus, for the purpose of this thesis, we deploy a test code of smart contract in a private vaccine supply chain network. The procedure followed for the development of the smart contract involved defining the

Figure 4-8: Activity diagram for Vaccine Supply chain smart contract
variables that changed throughout the process in end-to-end supply chain, nodes of the network, the functions which enabled state changes and functions that enabled the output which gave data on provenance. The activity diagram of the smart contract is given in the figure 4-8.

Consider a case of vaccine supply chain with 2 manufacturers, 2 distributors and 2 clinics. For this network, for the sake of illustration we will also consider 2 patients. These 8 entities of the supply chain along with admins (which can be regulatory authorities) form the private blockchain network for our supply chain. In the table given below, the entities along with their account number, private key and account address of the blockchain are given.

<table>
<thead>
<tr>
<th>Account</th>
<th>Node</th>
<th>Public ID</th>
<th>Private ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Contract deployment</td>
<td>0xA59dbd3940E273270473 8E9bd974553Ea8DcDf4d</td>
<td>b29c2bb4465023c026150dd6e851e5b7b633d1 7da527beb90ca28f8c329815</td>
</tr>
<tr>
<td>2</td>
<td>Contract deployment</td>
<td>0x2c2445697e8e3B862B5C E8B21CD93EF43B55e9D</td>
<td>3a9d6bc5796aad3cc035403018af4d9136d024a68 39298d88797e4eb283e71</td>
</tr>
<tr>
<td>3</td>
<td>Manufacturer</td>
<td>0x0623Fbe9c22e27EE6 C6cA8202e5f0B862E68</td>
<td>80318e7a126e462cfe9e9158a6e45457df2668 3f7f4f9d95527e747edca</td>
</tr>
<tr>
<td>4</td>
<td>Manufacturer</td>
<td>0xf28F00F92C086DFEae08 89BBDE8e8c6FBB51C99D3E</td>
<td>a6aed6b9e9f701770773e35bd6a25d724e67775 e5aeadac4e630da82da8e12d7</td>
</tr>
<tr>
<td>5</td>
<td>Distributer</td>
<td>0xB5bE77b420c808D437aC 0C69281F38B21200966</td>
<td>c6e126182a00481647474085075c8ed1e3e5d318f 8e1e16f2d8fe92605f1854</td>
</tr>
<tr>
<td>6</td>
<td>Distributer</td>
<td>0xC7F8a8e5d49C6bA32250 A6D4baAC0f9aF18F32E</td>
<td>5d38f8536ae29059bfe2b9ef48b29bba1d407c630 4fba3920300c0dde945819a</td>
</tr>
<tr>
<td>7</td>
<td>Clinic</td>
<td>0xa59D9922BDc16eA4E5E9 5DD4AbEc36076F6d925106</td>
<td>21597f51d384da8757cfe26b3bd5aa97fe048e3 6453d661881c0383d781515d</td>
</tr>
<tr>
<td>8</td>
<td>Clinic</td>
<td>0xFddf578e444eb6dA8F54e6E8744a89B52262F81</td>
<td>0604bda2cede6ba37a40e9f041646724adbb91 287861b1cb2459a1446e68a</td>
</tr>
<tr>
<td>9</td>
<td>Patient</td>
<td>0xA908c131E4b477AF56165aC632440CECAe36fd91C</td>
<td>9c25232560d671ccac0f8c1f54a955d6f161ee33a 22631d83a93f3c2b10020e6</td>
</tr>
<tr>
<td>10</td>
<td>Patient</td>
<td>0xC79911F31A3FC3907748fD81B6cdd1aD3Fcc669</td>
<td>89260841bc568ce8889ec4b32b8e7c64ae081e1c3e9 9645ffca4af0fb9d924d68b</td>
</tr>
</tbody>
</table>
Before developing the smart contract, it is important to program the nodes in the network i.e., the Manufacturer, Distributer, Clinic and Patient. The program defining these players of the network are given in Appendix. A subsidiary contract to the Vaccine Supply Chain contract is the migrations contract which defines the address of the owner of the asset once it’s transfer is recorded. Another subsidiary contract which checks basic authorization control in the main smart contract is the ‘Ownable’ contract. All the smart contracts are further provided in the Appendix.

The Vaccine Supply Chain smart contract involves defining the vaccine parameters that need to be logged, the eight events that are defined in the activity diagram above and the functions to input and output the data for logging the vaccine states. The code for the smart contract is given in the appendix. In this section, we will define the procedure for executing the end-to-end provenance for the vaccine supply chain.

- **Step 1:** This step involves logging in the 4 players of the network and their account IDs into the system. This will help in identifying the ownership of the vaccine at a given point of time and will in future help in establishing accountability.
Step 2: Once the entities are registered, data is recorded by the entities at all stages of the supply chain as defined by the events. The first entity that records data is the Manufacturer, and the first event is Manufacturing of the Vaccine. At this stage we record the Unique product code of the vaccine, Name of the manufacturer, and a tuple of information about manufacturing details including manufacturing temperature, manufacturing time, batch number of productions, plant of manufacturing and the location to which the vaccine is dispatched for distribution. Before the data is recorded, approval for the record is asked at each stage from the concerned entity. Next two stages when the data is recorded by the manufacturer is when the vaccines are packed and dispatched to the distributor. Additional Figure 4-10: Data recorded at vaccine manufacturing stage
stages can be added to this process according to the convenience and requirement of the entities of the supply chain.

- **Step 3:** Data is recorded when the distributer receives this vaccine. This helps to track the changes to the vaccines during transportation, as the data is recorded at the beginning and the end of the transport of the vaccine. Once the vaccine is procured by the distributer, it is stored before shipping to the clinic/ vaccine administration center. Data is recorded before shipping the vaccine to the clinic to make sure that the temperature and time is recorded to maintain the efficacy of the vaccine.

![Figure 4-11: Data recorded by Distributer at procurement stage](image-url)
- **Step 4:** Once the vaccine reaches the clinic, the time and temperature are recorded again to make sure they match the guidelines and also for traceability purposes in the future. At this stage the vaccine is stored for a prescribed number of days until a patient comes for immunization.

- **Step 5:** In the final step of the process, the patient receives the vaccine and is immunized. At this stage, the patient can check all the data recorded in the lifecycle of the vaccine before getting immunized, to have visibility in the supply chain and make sure that the vaccine was preserved at the prerequisite temperatures and stored for not more than the prescribed number of days.
Chapter 5
Conclusions and Future Work

Throughout the whole process of recording the data, the system makes sure that the events occur in the order of it’s chronology. If an attempt is made to record data out of sequence, the system will not accept that record. This makes sure that there is not counterfeiting and the vaccine moves through the system with the entities in the order it is supposed to move. Figure 5-1 shows an example of an error thrown by the blockchain system when an attempt was made to skip the event of Vaccine receival by the clinic and the direct Immunization of patient was attempted.

The system also makes sure that no data is recorded by an unauthorized entity. If a distributor tries to enter manufacturing time or makes an attempt to alter that data, the system does not allow it and throws an error. This helps in making sure that no false records are maintained.
in the system and each entity is held accountable for its own data records. Figure 5-2 shows an example of an error thrown by the system when a distributor tried to dispatch the vaccine when it was in the ownership of the manufacturer. The system makes sure that the ownership is transferred sequentially through the events programmed in the blockchain. The code ensuring the transfer of ownership through the blockchain is given in the Appendix. This property of immutability allows the blockchain to be transparent and trustworthy. At the same time, utmost care is to be taken while developing the smart contract so that the system replicates a real-life scenario.

Finally, as described in the problem statement earlier, it is very important to have visibility at each stage of the supply chain to achieve complete provenance. A blockchain system can be queried about the information recorded by manufacturers, distributors and clinics. In case of a faulty vaccine being administered to a patient, the blockchain records can help identify the exact batch of production, the manufacturing plant, the time and temperature of manufacturing, distribution and administration. For the purpose of this thesis, two separate functions are designed to give information about the manufacturing of the vaccine and the details about the whole supply chain.
chain in general. Figure 5-3 shows the manufacturing details of two unique product ids ‘123’ and ‘789’ and figure 5-4 shows end-to-end visibility into the supply chain.

Figure 5-3: Visibility in to manufacturing data captured by blockchain

Figure 5-4: End-to-end data captured by the blockchain
Future work in this field can be done by incorporating IoT devices in the research. A case study can be conducted using an actual supply chain use case to test the blockchain network. Separately, multiple small manufacturers and distributors can be involved in case of an emergency use case for rapid production of vaccines to achieve herd immunity faster. The transparency, immutability and trust provided by a blockchain network helps in achieving that. Multiple pharmaceutical SMEs when involved in the manufacturing, not only provide a faster production rate but also a very highly connected supply chain which helps government agencies provide vaccines to even the remote areas quickly. However, it also increases the problem of substandard vaccines and counterfeiting multifold. Blockchain networks with high number of nodes and visibility offer a very good solution to tackle that problem.
Bibliography


Appendix

This chapter contains all the codes used to develop the smart contract in Remix IDE. The code is written using Solidity language which is Turing complete.

**Migrations.sol**

```solidity
pragma solidity ^0.5.8;

contract Migrations {
    address public owner;
    uint public last_completed_migration;

    constructor() public {
        owner = msg.sender;
    }

    modifier restricted() {
        if (msg.sender == owner) _;
    }

    function setCompleted(uint completed) public restricted {
        last_completed_migration = completed;
    }

    function upgrade(address new_address) public restricted {
        Migrations upgraded = Migrations(new_address);
        upgraded.setCompleted(last_completed_migration);
    }
}
```
pragma solidity ^0.5.8;

/// Provides basic authorization control
contract Ownable {
    address private origOwner;

    // Define an Event
    event TransferOwnership(address indexed oldOwner, address indexed newOwner);

    /// Assign the contract to an owner
    constructor () internal {
        origOwner = msg.sender;
        emit TransferOwnership(address(0), origOwner);
    }

    /// Look up the address of the owner
    function owner() public view returns (address) {
        return origOwner;
    }

    /// Define a function modifier 'onlyOwner'
    modifier onlyOwner() {
        require(isOwner());
        _;
    }

    /// Check if the calling address is the owner of the contract
function isOwner() public view returns (bool) {
    return msg.sender == origOwner;
}

/// Define a function to renounce ownership
function renounceOwnership() public onlyOwner {
    emit TransferOwnership(origOwner, address(0));
    origOwner = address(0);
}

/// Define a public function to transfer ownership
function transferOwnership(address newOwner) public onlyOwner {
    _transferOwnership(newOwner);
}

/// Define an internal function to transfer ownership
function _transferOwnership(address newOwner) internal {
    require(newOwner != address(0));
    emit TransferOwnership(origOwner, newOwner);
    origOwner = newOwner;
}

Vaccineaccesscontrol

Nodes.sol

pragma solidity ^0.5.8;

// dev Library for managing addresses assigned to a Node
library Nodes {

struct Node {
    mapping (address => bool) bearer;
}

// dev give an account access to this role
function add(Node storage node, address account) internal {
    require(account != address(0));
    require(!has(node, account));
    node.bearer[account] = true;
}

// dev remove an account's access to this role
function remove(Node storage node, address account) internal {
    require(account != address(0));
    require(has(node, account));
    node.bearer[account] = false;
}

function has(Node storage node, address account) internal
    view
    returns (bool)
{
    require(account != address(0));
    return node.bearer[account];
}

Manufacturer.sol
pragma solidity ^0.5.8;

import "./Nodes.sol";
contract Manufacturer {
    using Nodes for Nodes.Node;

    event ManufacturerAdded(address indexed account);
    event ManufacturerRemoved(address indexed account);

    Nodes.Node private Manufacturers;

    constructor() public {
        _addManufacturer(msg.sender);
    }

    modifier onlyManufacturer() {
        require(isManufacturer(msg.sender),"Only Manufacturers allowed.");
        _;
    }

    function isManufacturer(address account) public view returns (bool) {
        return Manufacturers.has(account);
    }

    function addManufacturer(address account) public onlyManufacturer {
        _addManufacturer(account);
    }

    function renounceManufacturer() public {
        _removeManufacturer(msg.sender);
    }
}
function _addManufacturer(address account) internal {
    Manufacturers.add(account);
    emit ManufacturerAdded(account);
}

function _removeManufacturer(address account) internal {
    Manufacturers.remove(account);
    emit ManufacturerRemoved(account);
}

Distributor.sol
pragma solidity ^0.5.8;

import "./Nodes.sol";

contract Distributor {
    using Nodes for Nodes.Node;

    event DistributorAdded(address indexed account);
    event DistributorRemoved(address indexed account);

    Nodes.Node private distributors;

    constructor() public {
        _addDistributor(msg.sender);
    }
modifier onlyDistributor() {
    require(isDistributor(msg.sender), "Only distributors allowed.");
    _;
}

function isDistributor(address account) public view returns (bool) {
    return distributors.has(account);
}

function addDistributor(address account) public onlyDistributor {
    _addDistributor(account);
}

function renounceDistributor() public {
    _removeDistributor(msg.sender);
}

function _addDistributor(address account) internal {
    distributors.add(account);
    emit DistributorAdded(account);
}

function _removeDistributor(address account) internal {
    distributors.remove(account);
    emit DistributorRemoved(account);
}
Distributor.sol

pragma solidity ^0.5.8;

import './Nodes.sol';

contract Clinic {
    using Nodes for Nodes.Node;
    event ClinicAdded(address indexed account);
    event ClinicRemoved(address indexed account);

    Nodes.Node private Clinics;

    constructor() public {
        _addClinic(msg.sender);
    }

    modifier onlyClinic() {
        require(isClinic(msg.sender),"Only Clinics allowed.");
        _;
    }

    function isClinic(address account) public view returns (bool) {
        return Clinics.has(account);
    }

    function addClinic(address account) public onlyClinic {
        _addClinic(account);
    }
}
function renounceClinic() public {
    _removeClinic(msg.sender);
}

function _addClinic(address account) internal {
    Clinics.add(account);
    emit ClinicAdded(account);
}

function _removeClinic(address account) internal {
    Clinics.remove(account);
    emit ClinicRemoved(account);
}

Patient.sol
pragma solidity ^0.5.8;

import "./Nodes.sol";

contract Patient{
    using Nodes for Nodes.Node;
    event PatientAdded(address indexed account);
    event PatientRemoved(address indexed account);

    Nodes.Node private Patients;

    constructor() public {

```solidity
    _addPatient(msg.sender);
}

modifier onlyPatient() {
    require(isPatient(msg.sender), "Only Patients allowed.");
    _;
}

function isPatient(address account) public view returns (bool) {
    return Patients.has(account);
}

function addPatient(address account) public onlyPatient {
    _addPatient(account);
}

function renouncePatient() public {
    _removePatient(msg.sender);
}

function _addPatient(address account) internal {
    Patients.add(account);
    emit PatientAdded(account);
}

function _removePatient(address account) internal {
    Patients.remove(account);
    emit PatientRemoved(account);
}
```
Accesscontrol.sol
pragma solidity ^0.5.8;

import './Patient.sol';
import './Distributer.sol';
import './Manufacturer.sol';
import './Clinic.sol';

contract AccessControl is Manufacturer, Distributor, Clinic, Patient {
    constructor() public {}}

Vaccinebase
VaccineSupplyChain.sol
pragma solidity ^0.5.8;
pragma experimental ABIEncoderV2;

import '../vaccineaccesscontrol/AccessControl.sol';
import '../vaccinecore/Ownable.sol';

// Define a contract 'Supplychain'
contract SupplyChain is Ownable, AccessControl {

    // Define 'owner'
    // address owner;
// Define a variable called 'upc' for Universal Product Code (UPC)
uint upc;

// Define a variable called 'sku' for Stock Keeping Unit (SKU)
uint sku;

// Define a public mapping 'items' that maps the UPC to an Item.
mapping (uint => Item) items;

// Define a public mapping 'itemsHistory' that maps the UPC to an array of TxHash,
// that track its journey through the supply chain -- to be sent from DApp.
mapping (uint => string[]) itemsHistory;

// Define enum 'State' with the following values:
enum State
{
  New,            // 0
  Manufactured,   // 1 manufacturer
  Packed,         // 2 manufacturer
  Dispatched,     // 3 manufacturer
  Procured,       // 4 Distributor
  Shipped,        // 5 Distributor
  Received,       // 6 clinic
  Immunized       // 7 patient
}

State constant defaultState = State.Manufactured;
// Define a struct 'Item' with the following fields:

struct Item {
    uint    sku;  // Stock Keeping Unit (SKU)
    uint    upc; // Universal Product Code (UPC), generated by the Manufacturer, goes on the package, can be verified by the Patient
    address ownerID;  // Metamask-Ethereum address of the current owner as the product moves through 8 stages
    address manufacturerID; // Metamask-Ethereum address of the manufacturer
    string  manufacturerName; // manufacturer Name
    uint batchnumber; //
    string manufacturerplant;
    uint manufacturetemperature; // can this be a global variable and updated at every node?
    string manufacturetime;
    string dispatchedtolocation; //distributer site distached to
    string dispatchtime;
    uint   productID;  // Product ID potentially a combination of upc + sku
    string productName; //Vacciname;
    State  itemState;  // Product State as represented in the enum above
    string procuredtime;
    uint procuretemperature;
    string shippedtime;
    uint shippedtemperature;
    string receivedtime;
    uint receivedtemperature;
    string immunizedtime;
    address distributorID; // Metamask-Ethereum address of the Distributor
    string distributersite;
    string shippedtolocation; //clinic shipped to
    address clinicID; // Metamask-Ethereum address of the Clinic
    string clinictemperature;
string clinicTime;
address patientID; // Metamask-Ethereum address of the Patient
}

// Define 8 events with the same 8 state values and accept 'upc' as input argument
event Manufactured(uint indexed upc, address manufacturerID);
event Packed(uint indexed upc);
event Dispatched(uint indexed upc);
event Procured(uint indexed upc,address distributorID);
event Shipped(uint indexed upc);
event Received(uint indexed upc,address clinicID);
event Administered(uint indexed upc, address patientID);

modifier onlyItemOwnerOrOwner(uint _upc) {
    require(items[_upc].ownerID == msg.sender || isOwner(),"caller is not the owner of the item");
    _;
}

// State assignment functions
// Creating new vaccine
modifier newItem(uint _upc) {
    require(items[_upc].itemState == State.New,"item already exists");
    _;
}

// Manufacturer manufactures vaccine
modifier manufactured(uint _upc) {

require(items[_upc].itemState == State.Manufactured,"invalid state, expected Manufactured");
;
}

//Manufacturer packs the vaccine
modifier packed(uint _upc) {
    require(items[_upc].itemState == State.Packed,"invalid state, expected Packed");
    ;
}

//Manufacturer dispatches vaccine to distributor
modifier dispatched(uint _upc) {
    require(items[_upc].itemState == State.Dispatched,"invalid state, expected Dispatched");
    ;
}

//Distributor procures the dispatched vaccine
modifier procured(uint _upc) {
    require(items[_upc].itemState == State.Procured,"invalid state, expected Procured");
    ;
}

//Distributor ships vaccine to the clinic
modifier shipped(uint _upc) {
    require(items[_upc].itemState == State.Shipped,"invalid state, expected Shipped");
    ;
}

//Clinic receives vaccine from distributor
modifier received(uint _upc) {
    require(items[_upc].itemState == State.Received,"invalid state, expected Received");
    ;
}

//Patient receives vaccine from clinic and is Immunized
modifier immunized(uint _upc) {
    require(items[_upc].itemState == State.Immunized,"invalid state, expected Immunized");
    _;
}

constructor() public {
    // owner = msg.sender;
    sku = 1;
    upc = 1;
}

struct ManufacturerInfo{
    uint _manufacturetemperature; // can this be a global variable and updated at every node?
    string _manufacturetime;
    uint _batchnumber; //
    string _manufacturerplant;
    string _dispatchedtolocation; //distributer site distached to
}

//manufacturing a new item - Done by Manufacturer
function manufactureVaccine(uint _upc, address _ManufacturerID, string memory _ManufacturerName, string memory _productName, ManufacturerInfo memory manuInfo)
)
    public
    onlyManufacturer
    newitem(_upc){
items[_upc].upc = _upc;
items[_upc].sku = sku;
items[_upc].productId = sku + _upc;
items[_upc].manufacturerID = _ManufacturerID;
items[_upc].ownerID = _ManufacturerID;
items[_upc].manufacturerName = _ManufacturerName;
items[_upc].productName = _productName;
items[_upc].itemState = State.Manufactured;
items[_upc].manufacturetemperature = manuInfo._manufacturetemperature;
items[_upc].manufacturetime = manuInfo._manufacturetime;
items[_upc].batchnumber = manuInfo._batchnumber;
items[_upc].manufacturerplant = manuInfo._manufacturerplant;
items[_upc].dispatchedtolocation = manuInfo._dispatchedtolocation;

sku = sku + 1;
emit Manufactured(_upc, _ManufacturerID);
}

function packVaccine(uint _upc)

    public manufactured(_upc)
    onlyItemOwnerOrOwner(_upc)
    onlyManufacturer
    {
    items[_upc].itemState = State.Packed;
    emit Packed(_upc);
    }

function dispatchedVaccine(uint _upc,
string memory _dispatchtime)
    public packed(_upc)
      onlyItemOwnerOrOwner(_upc)
      onlyManufacturer
    {
      items[_upc].itemState = State.Dispatched;
      items[_upc].dispatchtime = _dispatchtime;
      emit Dispatched(_upc);
    }

function procuredVaccine(uint _upc,
    string memory _distributersite,
    string memory _procuredtime,
    uint _procuretemperature
)
    public dispatched(_upc)
      onlyDistributor
    {
      items[_upc].itemState = State.Procured;
      items[_upc].ownerID = msg.sender;
      items[_upc].distributorID = msg.sender;
      items[_upc].procuredtime = _procuredtime;
      items[_upc].procuretemperature = _procuretemperature;
      items[_upc].distributersite = _distributersite;
      emit Procured(_upc, items[_upc].distributorID);
    }
function shippedVaccine(uint _upc,
    string memory _shippedtime,
    uint _shippedtemperature)
    public procured(_upc)
    onlyDistributor
    onlyItemOwnerOrOwner(_upc)
{
    items[_upc].itemState = State.Shipped;
    items[_upc].shippedtime = _shippedtime;
    items[_upc].shippedtemperature = _shippedtemperature;

    emit Shipped(_upc);
}

function receivedVaccine(uint _upc,
    string memory _receivedtime,
    uint _receivedtemperature
)
    public shipped(_upc)
    onlyClinic
{
    items[_upc].itemState = State.Received;
    items[_upc].receivedtime = _receivedtime;
    items[_upc].receivedtemperature = _receivedtemperature;
    items[_upc].ownerID = msg.sender;
    items[_upc].clinicID = msg.sender;
    emit Received(_upc, items[_upc].clinicID);
function immunizedPatient(uint _upc,
    string memory _immunizedtime)
    public received(_upc)
    onlyPatient
{
    items[_upc].itemState = State.Immunized;
    items[_upc].immunizedtime = _immunizedtime;
    items[_upc].ownerID = msg.sender;
    items[_upc].patientID = msg.sender;
    emit Administered(_upc, items[_upc].patientID);
}

// Define a function 'fetchItemBufferOne' that fetches the data
function fetchItemBufferOne(uint _upc) public view returns (
    uint    itemSKU,
    uint    itemUPC,
    address _ownerID,
    address _manufacturerID,
    string memory _manufacturerName,
    uint _manufacturetemperature,
    string memory _manufacturetime,
    uint _batchnumber,
    string memory _manufacturerplant,
string memory _dispatchedtolocation,
string memory _dispatchtime
)
{
// Assign values to the 8 parameters
itemSKU = items[_upc].sku;
itemUPC = items[_upc].upc;
_ownerID = items[_upc].ownerID;
_manufacturerID = items[_upc].manufacturerID;
_manufacturerName = items[_upc].manufacturerName;
_manufacturetemperature = items[_upc].manufacturetemperature;
_manufacturetime = items[_upc].manufacturetime;
_batchnumber = items[_upc].batchnumber;
_manufacturerplant = items[_upc].manufacturerplant;
_dispatchedtolocation = items[_upc].dispatchedtolocation;
_dispatchtime = items[_upc].dispatchtime;

return (
  itemSKU,
  itemUPC,
  _manufacturerID,
  _manufacturerID,
  _manufacturerName,
  _manufacturetemperature,
  _manufacturetime,
  _batchnumber,
  _manufacturerplant,
  _dispatchedtolocation,
  _dispatchtime
)
struct OtherProductInfo{
    string _procuredtime;
    uint _procuretemperature;
    string _shippedtime;
    uint _shippedtemperature;
    string _receivedtime;
    uint _receivedtemperature;
    string _immunizedtime;
}

// Define a function 'fetchItemBufferTwo' that fetches the data
function fetchItemBufferTwo(uint _upc) public view returns
(
    uint    itemSKU,
    uint    itemUPC,
    uint    _productID,
    string memory _productName,
    uint    _itemState,
    address _distributorID,
    address _clinicID,
    address _patientID,
    OtherProductInfo memory opi
)
{
    // Assign values to the 9 parameters
    itemSKU = items[_upc].upc;
itemUPC = items[_upc].sku;
_itemProductID = items[_upc].productID;
_itemProductName = items[_upc].productName;
_itemState = uint256(items[_upc].itemState);
_distributorID = items[_upc].distributorID;
opi._procuredtime = items[_upc].procuredtime;
opi._procuretemperature = items[_upc].procuretemperature;
opi._shippedtime = items[_upc].shippedtime;
opi._shippedtemperature = items[_upc].shippedtemperature;
_clinicID = items[_upc].clinicID;
opi._receivedtime = items[_upc].receivedtime;
opi._receivedtemperature = items[_upc].receivedtemperature;
_patientID = items[_upc].patientID;
opi._immunizedtime = items[_upc].immunizedtime;
return
(
    itemSKU,
    itemUPC,
    _productID,
    _productName,
    _itemState,
    _distributorID,
    _clinicID,
    _patientID,
    opi
);
}