The Drug Counterfeiting Problem

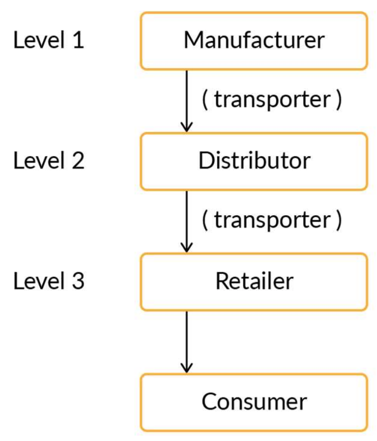
Supply Chain Management by Serialization

Sanath Swaroop
gmail@sanathswaroop.com

The Drug Counterfeiting Problem

Welcome to the Capstone Project on the application of blockchain in Life Sciences, this use case is on detecting and preventing counterfeiting of drugs.

SUPPLY CHAIN ARCHITECTURE



Workflow

The workflow required for the case study is divided into the following four units:

1. **Company Registration:**

1. All the entities who wish to be part of the supply chain network must be first registered or, in other terms, stored on the ledger.

2. **Drug Registration:**

    0. As a part of this process, any drug manufactured has to be registered on the ledger by the manufacturing company.

3. **Transfer Drug:**

    0. A buyer of the product will raise a Purchase Order for a particular drug.

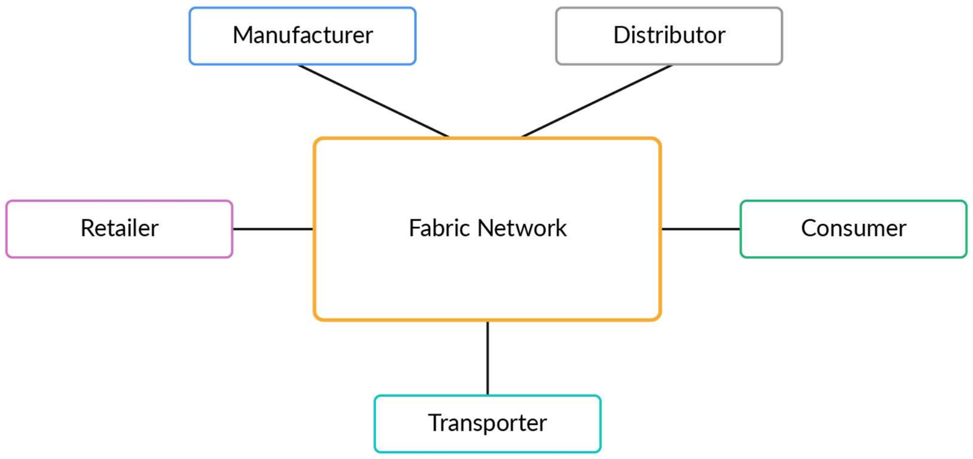
    1. The Purchase Order will be generated for a batch of drugs. It will include information like the name of the drug, the quantity required, Buyer, etc.

2. Based on the Purchase Order, the seller of the drug will initiate the process of shipment of the drug with the help of a transporter company like ‘FedEx’, and a shipment object will be created.
3. The shipment object will contain information like, the name of the transporter, origin, destination, etc.
4. Once the consignment is received by the buyer, the buyer will become the new owner of each item of the batch.
5. If the buyer is a consumer, then the Purchase Order and the shipment process need not be initiated. Only the owner of the drug is changed from the retailer to the consumer.

4. **View Lifecycle:**

    0. It is the process to view the lifecycle of the asset to date.
    1. Imagine a consumer or a retailer wishes to view the lifecycle of a drug called ‘amoxicillin’ with serial number ‘medi-001’. The ‘View Lifecycle’ functionality of the smart contract will allow any participant in the network to view the entire lifecycle of the asset.

NETWORK ARCHITECTURE



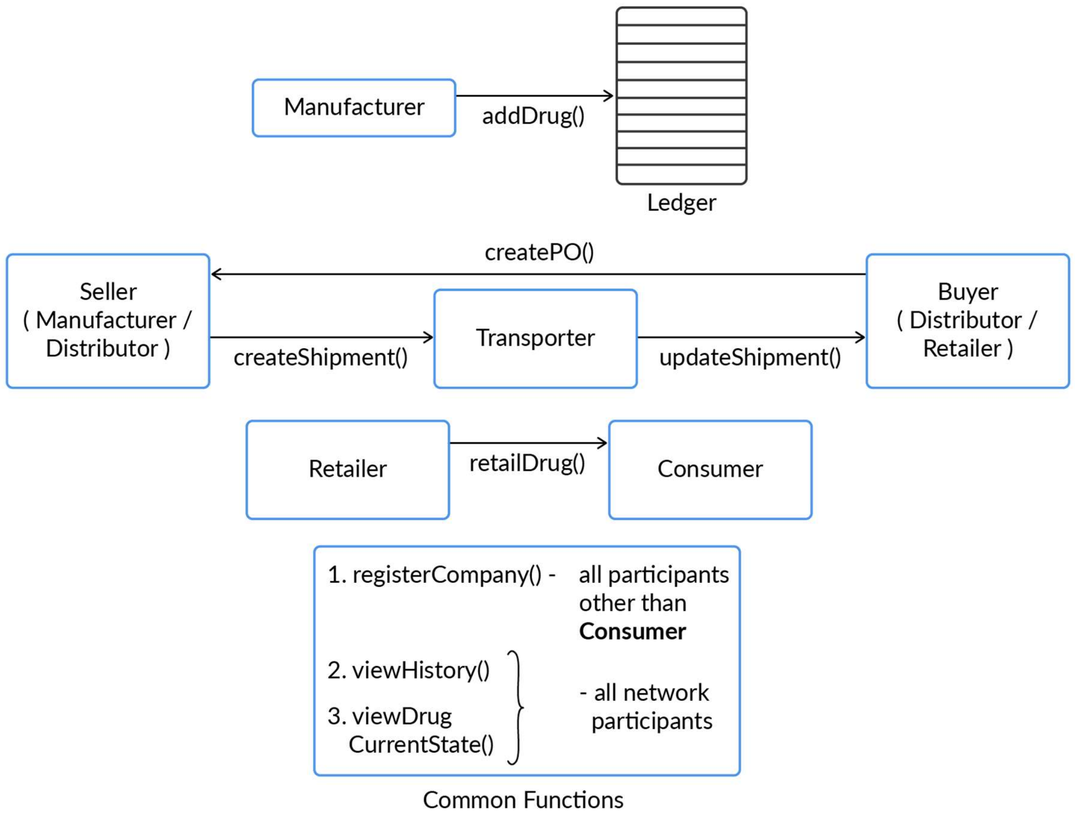
1. Name of the network: pharma-network
2. The network will consist of the following organisations.

1. Manufacturer- 2 peers
2. Distributor- 2 peers
3. Retailer- 2 peers
4. Consumer- 2 peers

5. Transporter- 2 peers

3. Initially, each organisation will have only 'Admin' as a User.
4. Each organisation must set up a fabric-ca service.
5. TLS should be disabled on the network.
6. Orderer Type: Solo
7. The name of the channel should be '**pharmachannel**'.
8. The name of the chaincode should be '**pharmanet**'.
9. The chaincode should be installed on both the peers of all the organisations.
10. Endorsement Policy: Any one of the organisations should endorse the transaction.
11. Peer0 of each organisation should be selected as the anchor peer.

SMART CONTRACT ARCHITECTURE



Manufacturer
addDrug()
Ledger
createPO()
Seller (Manufacturer / Distributor)
createShipment()
Transporter
updateShipment()
Buyer (Distributor / Retailer)
Retailer
retailDrug()
Consumer
1. registerCompany() - all participants other than **Consumer**
2. viewHistory()
3. viewDrug CurrentState()
- all network participants
Common Functions

The image above provides a graphical view of the transactions that will be defined as part of the smart contract. Details of all these transactions are provided below.

I. Entity Registration

1. registerCompany (companyCRN, companyName, Location, organisationRole)

Use Case: This transaction/function will be used to register new entities on the ledger. For example, for “VG pharma” to become a distributor on the network, it must register itself on the ledger using this transaction.

Company Data Model: The company asset will have the following data model:

- companyID: This field stores the composite key with which the company will get registered on the network. The key comprises the Company Registration Number (CRN) and the Name of the company along with appropriate namespace. CRN is a unique identification number allotted to all the registered companies. For this use case, you are free to use any random sequence of characters for the CRN number.
- name: Name of the company
- location: Location of the company
- organisationRole: This field will take either of the following roles:
    - Manufacturer
    - Distributor
    - Retailer
    - Transporter
- hierarchyKey: This field will take an integer value based on its position in the supply chain. The hierarchy of the organisation is as follows:
  Manufacturer (1st level) → Distributor (2nd level) → Retailer (3rd level).
  For example, the value of this field for “VG Pharma”, which is a distributor in the supply chain, will be ‘2’.

Note: There will be no hierarchy key for transporters.

II. Drug Registration

1. addDrug (drugName, serialNo, mfgDate, expDate, companyCRN)

Use Case: This transaction is used by any organisation registered as a ‘manufacturer’ to register a new drug on the ledger.
Validations:

- This transaction should be invoked only by a manufacturer registered on the ledger.

Drug Data Model: The new drug asset will be created on the ledger with the following fields:

- productID: Product ID will store the composite key using which the product will be stored on the ledger. This key comprises the name and the serial number of the drug along with an appropriate namespace.
- name: Name of the product

- manufacturer: Composite key of the manufacturer used to store manufacturer's detail on the ledger
- manufacturingDate: Date of manufacturing of the drug
- expiryDate: Expiration date of the drug
- owner: Key of the drug owner. For example, when the drug is in the manufacturing plant, the company manufacturing the drug is the owner. When the drug is being shipped, then the owner is the transporter. Similarly, when the drug is purchased by the Consumer, then they become the owner of the drug.
- shipment: Used to store the list of keys of all the shipment objects that will be associated with this asset. When the drug is added to the ledger, this field will store no value.

Note: If you fail to understand the above statement, do not worry. Come back to this field after reading the description for updateShipment.

III. Transfer Drug

1. createPO (buyerCRN, sellerCRN, drugName, quantity)

Use Case: This function is used to create a Purchase Order (PO) to buy drugs, by companies belonging to 'Distributor' or 'Retailer' organisation.
Validations:

- You need to make sure that the transfer of drug takes place in a hierarchical manner and no organisation in the middle is skipped. For example, you need to make sure that a retailer is able to purchase drugs only from a distributor and not from a manufacturing company.

PO Data Model: A purchase order with the following fields is created:

- poID: Stores the composite key of the PO using which the PO is stored on the ledger. This key comprises the CRN number of the buyer and Drug Name, along with an appropriate namespace.
- drugName: Contains the name of the drug for which the PO is raised.
- quantity: Denotes the number of units required.
- buyer: Stores the composite key of the buyer.
- seller: Stores the composite key of the seller of the drugs.

2. createShipment (buyerCRN, drugName, listOfAssets, transporterCRN)

Use Case: After the buyer invokes the createPO transaction, the seller invokes this transaction to transport the consignment via a transporter corresponding to each PO.
Validations:

- The length of 'listOfAssets' should be exactly equal to the quantity specified in the PO.

- The IDs of the Asset should be valid IDs which are registered on the network.

Shipment Data Model: Based on the PO, a shipment object will get created with the following details:

- shipmentID: Composite key of the shipment asset, which will be used to store the shipment asset on the ledger. This composite key is created using the buyer's CRN and the drug's name along with appropriate namespace.
- creator: Key of the creator of the transaction.
- assets: A list of the composite keys of all the assets that are being shipped in this consignment. For example, if three strips of 'paracetamol' are being shipped in a batch, then the composite keys of all these three strips will be contained in this field.
- transporter: The composite key of the transporter, created using transporterName and transporterCRN along with appropriate namespace.
- status: This field can take two values: 'in-transit' and 'delivered'. The status of the shipment will be 'in-transit' as long the asset does not get delivered to the system. As soon as the package is delivered, the status will change to 'delivered'.

Note: The owner of each item of the batch should also be updated.

3. updateShipment(buyerCRN, drugName, transporterCRN)

Use Case: This transaction is used to update the status of the shipment to 'Delivered' when the consignment gets delivered to the destination.
Validations:

- This function should be invoked only by the transporter of the shipment.

Outcomes:

- The status of the shipment is changed to 'delivered'.
- The composite key of the shipment object is added to the shipment list which is a part of each item of the consignment. For example, imagine there are 10 strips of 'paracetamol' in a particular consignment. When this consignment is delivered to the buyer, then each item of the consignment is updated with the shipment object's key.
  Note: Refer to the note added in the definition for addDrug() transaction.
- The owner field of each item of the consignment is updated.

4. retailDrug (drugName, serialNo, retailerCRN, customerAadhar)

Use Case: This transaction is called by the retailer while selling the drug to a consumer.
Validations:

- This transaction should be invoked only by the retailer, who is the owner of the drug.

Outcomes:

- Ownership of the drug is changed to the Aadhar number of the customer.
  Note: For this transaction, no PO creation is required.

IV. View Lifecycle

1. viewHistory (drugName, serialNo)

Description:

- This transaction will be used to view the lifecycle of the product by fetching transactions from the blockchain.
- The function should return the transaction id along with the details of the asset for every transaction associated with it.
- Hint: Refer to this resource to implement this function.

2. viewDrugCurrentState (drugName, serialNo)

Description:

- This transaction is used to view the current state of the Asset.

The folder structure inside pharma-net. Project is as follows:

1. **application** — folder contains different connection profiles for each stakeholder of the organisati on, wallet to store the identity of the Admin user of each organisation, node modules corresponding to every function defined in the smart contract and a node server for this application using the Expre ss library and expose these node modules for application as server-side endpoints.

2. **chaincode** — folder contains the smart contracts which are defined in the Smart Contract Archite cture.

3. **network** — folder contains the fabric network setup which are defined in the Network Architectu re.

4. **test** — folder contains the json files for postman.

Follow to see the network setup and execution of smart contracts from postman.

Steps to Test:

1. Generating crypto-material and channel-artifacts for the network.

cd pharma-net

sudo ./fabricNetwork.sh generate

2. Instantiate all the docker containers for each peer from the docker-compose.yaml file.

sudo ./fabricNetwork.sh up

3. Install and instantiate on the hyperledger fabric network.

sudo ./fabricNetwork.sh install

4. These are the steps that has to be done before execution of the chain code using Postman.

cd ../application

sudo dpkg — configure -a

sudo apt install npm

Create a manufacturer with the following details:

- Name: 'Sun Pharma'
- CRN number: 'MAN001'
- Location: 'Chennai'

Create a transporter with the following details:

- Name: ‘FedEx’
- CRN number: ‘TRA001’
- Location: ‘Delhi’

Create another transporter with the following details:

- Name: ‘Blue Dart’
- CRN number: ‘TRA002’
- Location: ‘Bangalore’

Create a distributor with the following details:

- Name: ‘VG Pharma’
- CRN number: ‘DIST001’
- Location: ‘Vizag’

Create a retailer with the following details:

- Name: ‘upgrad’
- CRN number: ‘RET002’
- Location: ‘Mumbai’

Create 4 strips of a drug named ‘Paracetamol’ with serial number starting from ‘001’ to ‘004’.

Test Case 2: Supply Chain

Create a collection file named ‘Supply Chain’ which contains the HTTP requests to perform the following functions.

Part a:

1. Purchase Order raised by ‘VG Pharma’ to purchase 3 strips of paracetamol from ‘Sun Pharma’.
   Expected Output: Display the PO object created in the response body.

2. Shipment created by ‘Sun Pharma’ in response to the raised purchase order. ‘FedEx’ acts as the transporter.
   Expected Output: Display the Shipment object created in the response body.

3. ‘FedEx’ delivers the shipment to ‘VG pharma’.
   Expected Output: Display the data of each asset of the shipment.

Part b:

1. Purchase Order raised by ‘upgrad’ to purchase 2 strips of paracetamol from ‘VG Pharma’.
   Expected Output: Display the PO object created in the response body.

2. Shipment created by ‘VG Pharma’ in response to the raised purchase order. ‘Blue Dart’ acts as the transporter.
   Expected Output: Display the Shipment object created in the response body.

3. ‘Blue Dart’ delivers the shipment to ‘upgrad’.
   Expected Output: Display the data of each asset of the shipment in the response body.

Part c:

1. A customer named ‘Akash’ with Aadhar Number 'AAD001' buys 1 paracetamol strip from the retailer ‘upgrad’.
   Expected Output: Display the data of the asset bought by Akash in the response body.

Test Case 3: History Track Down

Create a collection file named ‘History’ which contains the HTTP requests to perform the following functions.

1. The customer ‘Akash’ wishes to check the history of the paracetamol that he bought from ‘upgrad’.
   Expected Output: The response body should display the entire lifecycle of the asset.
2. The customer 'Akash' wishes to check the current state of the paracetamol that he bought from ‘upgrad’.
   Expected Output: The response body should display the current state of the asset.