Blockchain Implementation in Agriculture

Sai Pujitha Selvakumar
M. S. Computer Science
University of Florida
Gainesville, FL, USA
saipujitselvakum@ufl.edu

Adil Shaik
M. S. Computer Science
University of Florida
Gainesville, FL, USA
adilshaik@ufl.edu

Venkata Sindhu Kandula
M. S. Computer Science
University of Florida
Gainesville, FL, USA
v.kandula@ufl.edu

***Abstract*— In recent years, many case studies and implementations have been done by Walmart-IBM, Skuchain, Oregon, LDC, etc., on implementing blockchain for agriculture mainly in the domains of food traceability and supply chain management using Hyperledger fabric, Popcodes, Sawtooth, Ethereum, Quorum[21,22] to create Agri-Based Decentralized systems. In this project proposal, we have analyzed various white papers and practical implementations in different organizations, the technologies, and protocols they use and investigated on the challenges and limitations of these implementations. In this project, we aim to extend blockchain architecture to make product details accessible to consumers, which many of the current papers/organizations haven't taken into consideration and the challenges involved in the same. We have also specified the road map and goals for the project[25]. In addition, we have given various solutions and ways to tackle the vulnerabilities and limitations identified in the implementation.**

Keywords—agriculture, Blockchain, project, proposal, roadmap, challenges, limitations, food, traceability, P2P, supply chain management, consumers, Hyperledger fabric, Sawtooth

I. INTRODUCTION

In July 2017, papayas in the US market were linked to a multi-state outbreak of Salmonella. It took almost three weeks to track the origin back to a single farm in Mexico. With introducing blockchain to agriculture, this time can be reduced to as far as 7.7sec[2]. This has been one of the main reasons for many companies like Walmart-IBM[6], Skuchain, Maersk, LDC, Cargill, Bungee, Provenance etc., on implementing blockchain for agriculture mainly in the domains of food traceability and supply chain management. Their main focus has been to aid the industries and organizations involved in the food industry. Our focus for this project deals with bringing this technology to the consumer, designing an architecture and identifying various challenges and limitations involved in this process.

II. MOTIVATION FOR THE PROJECT

Food fraud is estimated to cost the global food industry $40 billion a year. In 2011, China witnessed a massive pork mislabeling debacle, along with a contamination hoax in which donkey meat products were recalled because they were found to include fox meat (Bradsher, 2011; Clemons,2014)[2,7]. All the agri-food industries that are implementing blockchain are using it as an efficient solution for their business problems and management and is visible and accessible to a closed network[3]. This network includes farmers, distributors, processors and others involved in the production till the retailers.

With increasing trust issues in customers and scale of fraud involved. It is important to provide product details to consumers through trusted infrastructure. Motivation of our project is to make every detail of the product visible to consumers. For this, we are using data of products stored in blockchain and presenting them through an interface to customers for transparency.

III. RELATED WORK

In blockchain implementation, Information related to a product is obtained by incorporating IoT devices in each stage of farming, distribution, processing, marketing and retailing[4,8]. Even the quality of the product can be tracked using soil and water level sensors in the farming stage and storage conditions in warehouse can also be tracked through temperature sensors. Location based sensors like GPS are used in real time implementation for updating smart contracts of the blocks[12]. All these details are transferred through gateway software and smart contracts to blocks and consensus algorithms used in various blockchain models discussed below are different[24]. Business logic for updating the status and other event based triggers can be programmed in application level and induced to smart contracts. Like updating data when a product reaches the distributor from farmer. Once data is stored in block it is immutable and untamperable and accessibility to it depends on the permissions of the blockchain model. Above discussed model is used in companies like Walmart, Oregon etc.,[20]

IV. CASE STUDIES

1. HyperLedger Fabric

Walmart, Nestle, and Golden state foods have used HyperLedger Fabric to develop blockchain applications for Agri based supply management[1,14]. HyperLedger Fabric is a modular extensible open-source system for deploying and operating

permissioned blockchains. The most attractive feature of HyperLedger fabric which makes it the most selected platform for building Agri based is due to its openness towards the choice of programming language. It allows the creation of Distributed Ledger systems using general programming languages such as Go, Java, and NodeJs. HyperLedger also allows integration to industry-standard identity management systems for the creation of distributed applications.

Walmart, Nestle and Golden state use blockchain for internal tracking of their food products and for detecting fraud at any point. HyperLedger being a permissioned blockchain allows this[14,15,17,18]. The food suppliers, processors and retailers make up for the entities on the blockchain. No outside entity can access the blockchain network. Having a permissioned blockchain helps secure the blockchain network from attacks and also hides business trade from the public.

Hyperledger believes that there is no one-size-fits-all solution to BFT[9]. Hence, it provides flexibility in the consensus model and trust model. Hyperledger uses an Execute-Order-Validate Model which is different from other blockchain platforms that have order-execute architecture.

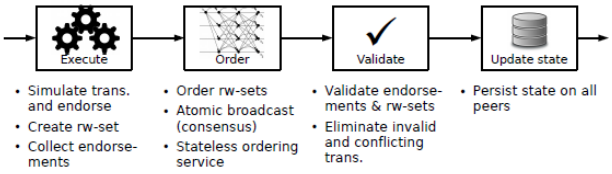


Figure 1.a

a. Hyperledger Fabric Architecture:

Fabric Architecture uses the Execute-Order-Validate Model. The transaction made by any supplier is executed by a subset of the trusted node, which can even handle non-deterministic cases. The trusted nodes endorse the transaction by simulating it. After getting endorsements from the trusted peers the node is broadcasted to all the nodes by Ordering Service. Each node validates the endorsements and adds the node to their local ledger.

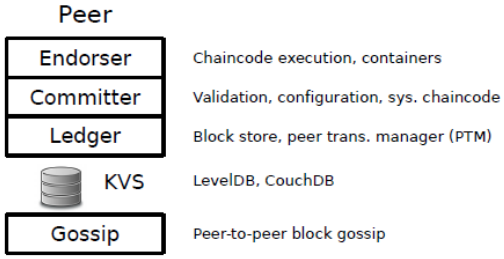


Figure 1.b

b. Hyperledger Fabric Node Components:

Each peer/node has the following Component - Endorser, Committer, Ledger, KVS and Gossip Protocol (fig-1.b). The Endorser defines the execution policy for the transaction. The Committer component defines the protocol for validation of the transaction. Ledger is the local blockchain of the node. KVS is a database that gives a snapshot of the transaction in a key-value structure.

2. Hyperledger Sawtooth:

Intel and Oregon State University collaborated on a food traceability project for blueberries of Oregon brand using Hyperledger sawtooth. Their main focus was to collect data regarding the temperatures of berries as they are very sensitive to heat and it affects their quality and to eliminate manual tracking of farming practices and supply chain conditions. To enable pallet-level control and monitoring of goods for inspection of shipped goods[1].

a. Hyperledger Sawtooth architecture:

This is an enterprise blockchain platform that supports hyperledger applications. Business logic that consists of conditions for data creation, modification. Transaction processing can be specified in the application layer using Python, JavaScript, Go, C++, Java or Rust. This dynamic environment allows us to customize transaction rules, consensus algorithms and permissions[5].

Multiple application types are allowed in the same instance and design of smart contract, business logic is done in the transaction processing layer. We can use either permissioned or permission less blockchain.

Parallel transaction execution is a valued feature of sawtooth. It isolates the execution of transactions from one another while maintaining contextual changes. As parallel scheduling provides a substantial potential increase in performance over serial execution. It allows applications to subscribe to specific events defined in the smart contract and transaction processes.

The consensus algorithm used in this model is PoET(Proof of Elapsed Time)[10,11]. PoET is a modification of PoW (Proof of Work) that allows scalability as it works on secure instruction execution and cuts down the power consumption compared to PoW.

It also interoperates with Ethereum through an integration project. which enables smart contracts in Ethereum virtual machines to be deployed in sawtooth.

Live tracking of products using GPS sensors was implemented by intel with Intel connected logistics platform[19]. Companies like Curry & Co. have decent supply chain monitoring and in such cases they require a model that does not interfere with existing workflow

and so sawtooth is opt choice as they are pluggable frameworks that do not interfere with the present flow.

The key difference between Fabric and Sawtooth is that fabric uses only a permissioned network while sawtooth can use either of it as per requirement.

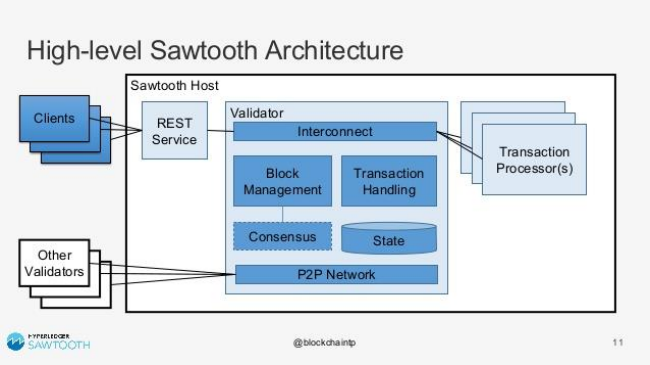


Figure 2

3. *Ethereum blockchain architecture:*

Ethereum blockchain is a public/permission less network that allows everyone to participate. Smart contracts and all transaction processing logics can be written using solidity. For applications developed on Ethereum blockchain, there is no need for a 3rd party payment platforms as the transactions can be done using native tokens.

Ethereum provides interoperability by supporting applications developed using other hyperledger technologies like sawtooth to work on Ethereum Virtual Machine. Consensus mechanism is established by mining based on Proof of work algorithm (PoW).

This hyperledger model is best fit for generalized applications. But for enterprise related implementations Sawtooth and Fabric are suitable as they provide permissioned blockchain, high resilience, confidentiality and scalability.

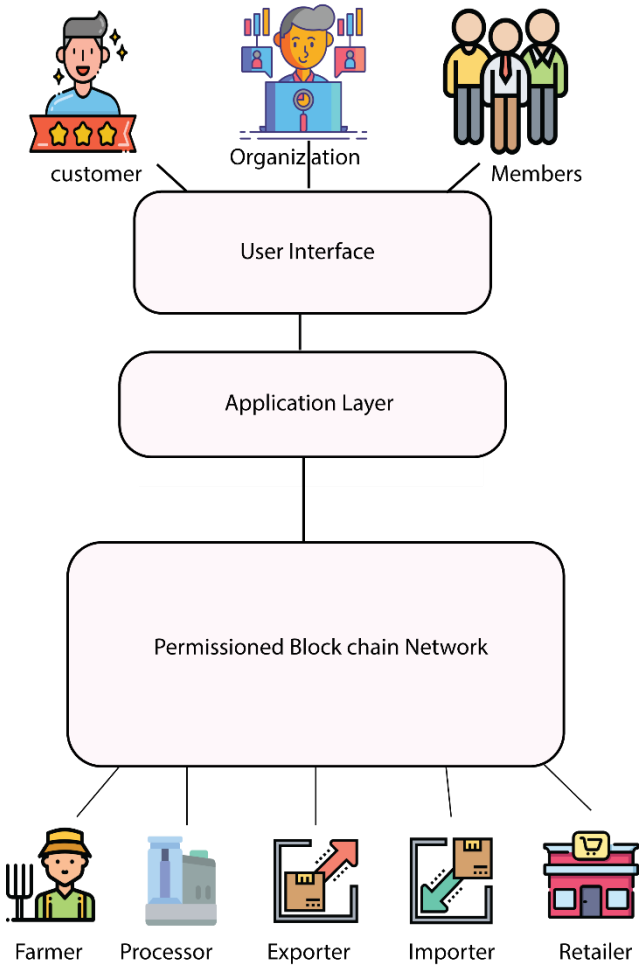
Quorum is an Ethereum based blockchain platform that is developed to cater for business purpose, and it has many additional features to basic Ethereum like choice in choosing consensus algorithms (RAFT, IBFT or Clique POA), privacy options, cross environment compatibility and scalability. This new venture is supported by giants like JPMorgan and Microsoft.

V. SYSTEM ARCHITECTURE

There are different stages in the implementation of this application, starting from publishing data to blockchain to customization of blockchain implementation to retrieval of data from blockchain and presenting in an UI.

Blockchain Model:

After comparing Hyperledger fabric, sawtooth and Ethereum we have chosen Hyperledger sawtooth as it provides better scalability, separation of core system from application domain which makes development easier and levels of customization in transactions,



consensus and permissions. These features make it easier for legendary systems to adopt and include blockchain management in their industrial process.

Different stages involved in application flow:

1) Storing hash of input data:

Data being stored in blockchain is hashed before it is published to the network. SHA-512 algorithm can be used to hash the data. This hash code is also stored in block along with original data. By doing hash data integrity will not be compromised. This process is done on application level and provided to blockchain.

2) Publishing data to Blockchain:

All the members in the network input their data into blockchain using UI and that organization level application takes care of processing (hashing, selecting fields) and publishing data into blockchain.

Farmers, distributors, exporters, importers, processors and retailers are the members involved in this network.

When a block is created, a batch code is assigned by UI to the creator. This unique id must be presented by members updating status of that block. Rest of the

details are covered in smart contract description. ***3) Details stored in block:***

When a block is created for a product, It captures details of member and status of it. Details of member consists of their registration number and other properties relevant to the status of the product.

In different stages of product, different details are stored into the block and appended to previous blocks.

Stages and details stored in block are:

product Id, product name, hash value, location of product and status are common properties stored in every stage of product.

a) Farmers: Can have temperature, water levels and other growing conditions of the product stored.

b) Distributors: Days of storage, temperature, batch size etc., can be stored.

c) Exporters: Container conditions, days taken for exporting, status and details of export.

d) Importers: Condition of product when received, days taken to get delivered and storage details.

e) Processors: Ingredients added for preserving and processing, life of the product

f) Retailers: days on shelf, best before date of the product.

4) Block structure:

Product Id, product name, location of product and status are common properties stored in every stage of product.

Additional stage specific details are appended in block created in that stage. These details can be customized as per requirement and product type.

One of the important details of product stored is status. Status indicates the current level of product and how far it is from completion stage. Product id and status is the link between appended blocks related to that product.

5) smart contracts, transactions and streams:

Smart contracts bear the logic that drives the whole picture of this process. Transactions are the way to interact with the network and implement smart contracts.

Smart contracts in this case will have logic to check if it is an insert of new product or update to existing product.

New product: Product id is stored and status of it will be at farmers.

Existing product: If the product id is existing, then most recent block status of this product is used to check if product is in track or not (Like when a stage is skipped). Also, depending on the status of product, smart contract decides on which details to add to current block being appended.

Figure 3 - Our Proposed Architecture

Finished product: When a product status is marked as completed, it means that it is available on shelf to customers for purchasing.

In sawtooth, Application specific events can be created to execute certain actions. So, we can trigger an event when status is changed.

Streams: In Hyperledger sawtooth, there is a feature to group blocks that satisfy condition as per requirement. If this can be explored and implemented, products can be categorized and monitored in the blockchain level.

6) Retrieval of data from blockchain:

Products that reach their completion stage will have their status as completed. StateDeltaEvents is an event listener feature in sawtooth. Using this feature, we can trigger event when status of a product is updated and get the details of block in that status and changes made. Application must subscribe to event in order to get notifications and data of it through web sockets using a REST API component.

Once data is retrieved, all details of the product is stored along with its hash values. Product details can be mapped against its hash value while storing.

7) Authentication check for retrieving data:

To avoid intrusion of adversarial connections subscribing to blockchain events, public key authentication can be introduced at this level in the application layer. This check is introduced so that only entity with a public key to permissioned blockchain network of supply chain can subscribe and retrieve data.

For this, public key check can be implemented on blockchain part where event description is done. So that subscription to event is only valid if public key is provided.

8)Selected fields stored in database:

While all the details stored in block might not be of use on the application side. Database level segregation can be done on the details of product just required by customers and details required only for organizational observation.

If data is segregated and stored, hashing of segregated data must be performed and stored along with original attributes of product.

These can be customized at schema level of database of organization to accommodate retrieved data.

9)Processing of retrieved data:

Application level logic is specific to organization and this is where stored data is processed (if required) and pushed to front end (client side). In this scenario, application logic might contain conditions for abstraction of data depending user role permissions of

the client-side user. For example, If a customer is querying for product details. He should be allowed to view only the location and duration details of product and not farmer and other members identity details and prices involved in it. If the user is a member of organization, then he must be able to view more details than permitted to users with role as customer. Functions to get hash values of data should be there so that client side functions can call these and cross check the hash value after and before transmission.

10) server-side validation of data hash:

There is a scope for data compromise while processing and transmission to client side. As data becomes vulnerable in the application and network environment.

Security checks are required in every stage of data storage and usage once it is retrieved from blockchain to maintain the immutable feature of it.

As a measure, data should be hashed on server or client side of the application and cross checked with its original hash value before displaying to customers to cross check that there was no data manipulation while processing it in application layer and while transported through network layer.

11) Product details published on UI:

User Interface can be designed for three purposes.

Customers: Customers should be able to view details of their products from farm to shelf. But the details should be abstracted such that they do not get member related sensitive details. Only view permissions are provided for them.

Observation: This access is for organization and safety authorities to inspect and track origins of the product and standards of its storage and processing in different stages.

Members of network: For members like farmers, exporters etc., different components of client-side pages must be accessible. Like forms to input data related to product and to track the status of product from a chain they are involved in.

Security measures used in our model:

Data integrity check and authentication check are the security measure steps included in the application design to protect data against vulnerabilities.

Data integrity check hashes the input values before publishing to blockchain and uses this hash values to check its integrity at every other step after retrieving from blockchain.

Authentication check uses public key credentials while subscribing to events to make sure adversaries cannot listen to these events and retrieve data from blockchain network.

VI. WALK THROUGH WITH EXAMPLE PRODUCT

For better understanding of system design and functions, a product example of Rice is considered and complete cycle of it is explained.

Publishing data to Blockchain:

Farmer enters details of harvested rice crop through organization's UI and gets a unique ID (product ID) as response when all these details are published to block chain by creating a block.

When block is created, product details (product ID, product name, location, and status) are hashed and added along with other product details in blockchain.

Processor is considered to be the next member accessing the product in this case. When processor gets the harvested rice from farmer, Product ID is also provided to processor as this will be the identifier used to link updates to corresponding blockchain.

Processor will provide updated product details like location and status along with this code and so block created, updates status and inserts additional details of corresponding product through smart contract. Additional details usually are specific to process involved in that stage. Like temperatures of storage, condition of product when arrived and later when handing-off to next stage etc.,

In the same fashion batch code is shared to succeeding members of the product cycle like distributors, exporters, importers, and retailers etc., and every member input details through UI to block chain.

Storing hash of input data:

In every stage when data is updated to block, hash of the latest copy of all details are is formed and updated to block. So, when product achieves ‘complete’ stage. Latest hash is the valid hash of most recent data. This value is used further for security checks.

smart contracts, transactions and streams:

In smart contract, logic is written to trigger event when a product status is changed to "completed" (when product is with retailer, ready for purchase).

In this case, when rice batch reaches retailer after passing from farmer, processor, distributor, exporter and importer it will come to completed stage.

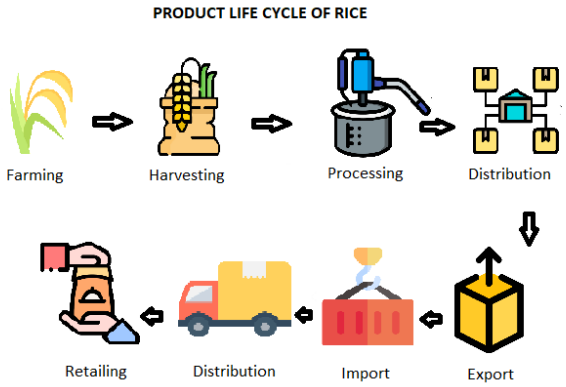
At this point, event listeners from application will get notification regarding the product completion and will receive the data stored in blockchain.

To avoid intrusion of adversarial connections subscribing to block chain events, public key authentication can be introduced at this level in the application layer. This check is introduced so that only entity with a public key to permissioned block chain

network of supply chain can subscribe and retrieve data. Further details are discussed in detail Section 5.6 and 5.7

After data is retrieved, It is processed to be displayed on user interface. While displaying server/client-side validation should be done by hashing the data again and crosschecking with hash value in retrieved data to verify that no data compromise has taken place while processing and transmission to client side.

User who have purchased rice can enter their product code and view all the information related to their purchased product (rice).



Selected fields stored in database:

Data retrieved may not be requested by clients immediately. So, they are stored in a permissioned data storage with high security measures.

So, product details are again hashed in SQL using stored procedures and cross-checked with hash stored in blockchain to clarify that data was not manipulated. If it is manipulated, then both hash values will not match, and security breach can be detected.

If both hash values match, then data is stored into database ready for further processing. Also, to ensure that data is not accessed by members maintaining database. Permissions are granted based on role and requirement. This explained in detail in further sections

Processing of retrieved data:

Even though we have all the details related to product in database, we need only some of it for displaying it to customers. Because we are maintaining data abstraction for different members accessing UI.

For customers we display only high-level data like track of locations in product life cycle and details related to product description and what is guaranteed.

For example, if rice is labelled as organic, then it is shown that no-artificial processing is done on it. While rest of the extra product details are masked.

But if Organization or inspectors are accessing UI, then they will be enabled to view all the product details. As this can help them in back tracking the product origins.

Client-side validation is done to the data being displayed by hashing it in application level and cross-checking with the hash before and after transferring to client-side.

VII. IMPLEMENTATION / PROTOTYPE:

For the purpose of demonstration, we used Ethereum as it allows a quick environmental setup. Truffle suite with Ganache is used for development of our demo.

We have created a contract called AgriChain. AgriChain stores details about agricultural product. Each product has a product id, name, status, current location and hash value. The hash value is generated by sha256 using product id, name, status, current location. Sha256 is available as a function in solidity. In our real implementation we recommend a Sha512 algorithm for hashing. The usage of hashing as explained earlier is for data integrity check when we view in front end. We have the following parameters in our Contract:

1. Struct – which defines the product. It has product name, status of product, hash value and current location.
2. Mapping of product id to Struct – Storing data for each product id.

In the current stage of our implementation we have defined Following methods:

- addProduct(): To add product details to the blockchain. This can be done by a farmer, processor, importer, exporter and retailer. (See figure 5)
- getProductDetail(): A getter to retrieve the product info based on productid. (See figure 4)

GitHub link for demo:
https://github.com/Sugooi/AgriChain

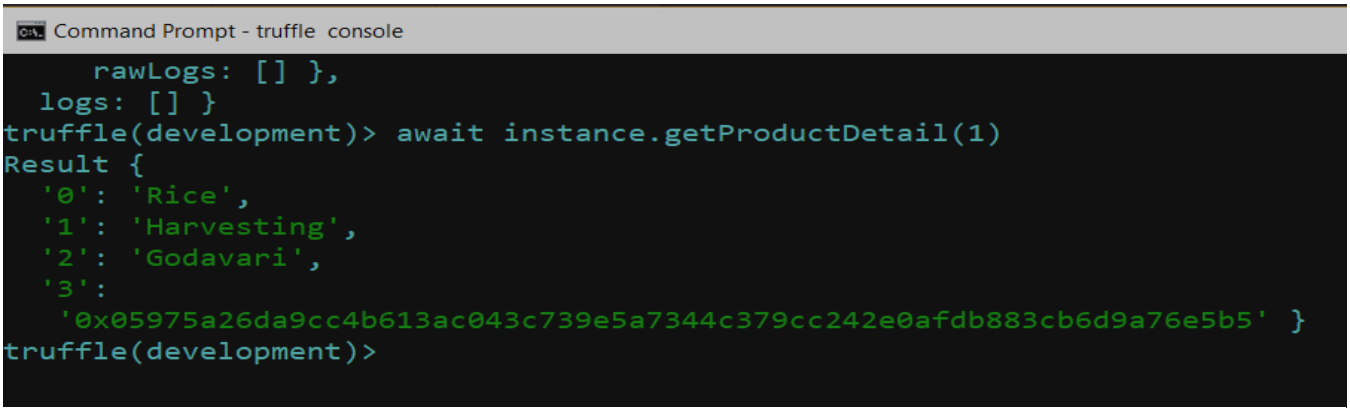


Figure 5

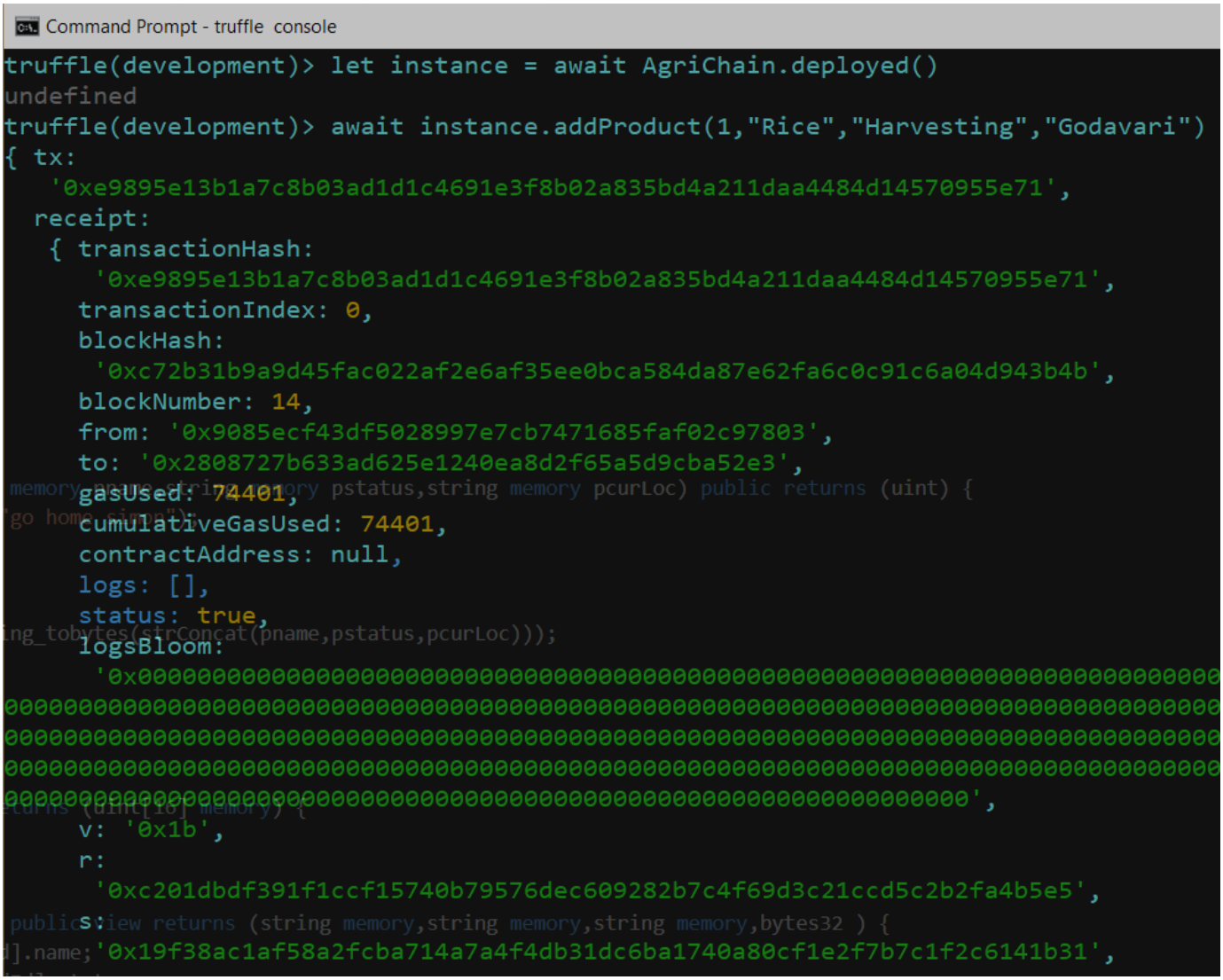


Figure 4

VIII. VULNERABILITIES

We have identified some vulnerabilities in our proposed model and have provided solutions for them to overcome these susceptibilities in the next section.

Vulnerabilities found in our model are:

a) Chance of data manipulation while retreiving from blockchain to database

b) Unathorized access to database used by application while processing data

c) Risk of unathorized users listening to event listeners enabled in blockchain

d) Data credibility of input provided by the members of blockchain

IX. SOLUTIONS

Solutions for the identified vulnerabilities are:

A. For risk of data manipulation while retreiving from blockchain to database

Microsoft SQL Server is compatible with Hyperledger Sawtooth. The data for each product in the Database consists of product ID, name, status, current location and hash value. In order to maintain data consistency in the application, we will hash the retrieved Product ID, name, status and current location using SHA-256 algorithm from the database and compare it with the hash value that is already present in the retrieved data for that product and ensure that both the hash values are same.[27]

We can make use of stored functions and user-defined functions for this purpose which provides an added layer of security. The similar hash values indicate that the data hasn't been tampered with by any kind of adversary and ensures data security till the end user.

B. For unauthorized access to database risk, we can implement LUA tactics in SQL server [28].

Developing an application through a least-privileged user account (LUA) is an important aspect of a defensive, in-depth approach for countering security risks. The LUA tactic ensures that users abide by the principle of least privilege and register and login with limited user accounts. Administrative duties are broken-down using persistent server roles, and the usage of the sysadmin fixed server role is significantly restricted. By granting the minimum permissions necessary to a user or position to get details about the product traceability details ensures the security of the application.

C. For risk of unathorized users listening to event listeners enabled in blockchain, We are proposing the features present in hyperledger sawtooth.

Sawtooth encrypts data simultaneously—that is, communications between Sawtooth nodes and external protocols or between components within a sawtooth node (such as between Validator, REST API (of application), and Transaction Processor node processes). Sawtooth uses ZeroMQ known as ZMQ or 0MQ for communications. CurveZMQ is used for ZMQ encryption and authentication, which uses a 256-bit ECC key with elliptical curve Curve25519[26].

This feature can be used to encrypt the communications between sawtooth and application. Doing so will avoid any risk of adversarial perties listening or trying to access blockchain through any event subscription.

If permissioned blockchain is used, risk of unathorized user access is very minimal. But for the sake of protecting data and to allow access to event listeneners on need basis, above mentioned feature is useful.

D. To minimize the risk of incorrect data input into blockchain, we can use more IoT devices and GPS trackers. We are already using these devices as part of data input itself. But, these can also be used to cross-check the data input provided by members manually.

Smart contracts and transaction processing logics can be coded to accommodate such check. This is most feasible solution as it is impossible to facilitate any field checks by organization often due to the scale of implementation and costs incurred.

But, fault tolerance mechansims have to be used in these devices to detect any attempt to tamper the actual data.

X. CHALLENGES & LIMITATIONS

List of issues to be addressed:

a) Allocate more ways to retrieve data and handle security issues related to it.

b) Consider and explore more robust techniques to implement a mechanism so that there is a secure channel from blockchain to the front end/customer view.

c) Designing application that is not just customer centric and takes into consideration the business needs of companies like Walmarts, Nestle and Covanta.

d) All the blockchain platforms like Hyperledger fabric, sawtooth, quorum is at the initial stage, so it is difficult to find enough real-time data that is being used. The usage data of companies like Walmart are also not publicly available.

XI. OUTLINE OF THE APPROACHES FOR THE LIMITATIONS:

Approaches listed in the order of challenges:

a) By designing retrieval methods at different levels of data status updates and implementing multiple retrieval methods in application level logic.

b) Investigate on all possible security threats that are feasible and design counter measures for them.

c) Implementing data abstraction and providing customized permissions for members in block chain and their node visibility, user role permissions for users accessing UI.

XII. ROAD MAP & MILESTONES FOR THE PROJECT:

1) ***Detailed study of blockchain architectures:***
First step in the roadmap is to study and analyze the different type of architectures that can be used for blockchain in agriculture like Hyperledger Fabric, Ethereum, Sawtooth, Quorum, Hypergrid etc.,

2) ***Analysis of case studies & real-time implementations:***
Different approaches have been taken forward by different organizations like Walmart-IBM, Skuchain, Oregon, Covantis, LDC, Nestle, Bungee, etc., in food traceability and supply chain management. An in-depth analysis of these different implementations, architectures and protocols that are being used must be done. We have identified that all these are beneficial towards the entities involved in selling and identified that our focus will be towards the consumer.

3) ***Projection of system architecture***
As mentioned in (b), the focus is proposing an architecture that makes the consumers trace the product in the food chain. This is the aspect where there was no proper if at all research done. We plan to base our project in this area as most of the proposed ideas deal with a private blockchain network whereas with consumer introduction, it might become a public or a privileged open private blockchain network.

4) ***Outline of challenges and limitations***
Coming up with challenges faced in the system architecture proposed and the study of limitations involved in the proposed design.

5) ***In-depth component, architecture and protocol design***
Studying further on the components involved in the architecture, and doing an in-depth analysis on the system design at the consumer end and proposing a concrete application design

6) ***Identifying the vulnerabilities in the proposed architecture***
Checking for security vulnerabilities, endpoint vulnerabilities. Feasibility study of the proposed implementation of the architecture in real time.

7) ***Ideas to curb the vulnerabilities***
coming up with ideas to overcome the challenges faced in the design proposal.

8) ***Final report***
Composing the final report and submission.

XIII. MILESTONES

Milestone 1: PROJECT PROPOSAL

i. Detailed study on existing works and coming up with challenges and limitations in the existing approaches
Roadmap sections: a, b, d

ii. **System Architecture Overview**
Coming up with the system architecture overview with involving the consumer. This is discussed more in detail in section VI of this paper.
Roadmap section: c

iii. **Milestone 3: MID-TERM**
In-depth component, architecture and protocol end-to-end design of the application
Roadmap section: e

iv. **Milestone 4: FINAL**
Identifying vulnerabilities, ideas to curb them, feasibility of the proposed design in real time and composing a final report on the entire findings that were done.
Roadmap section: f, g, h

XIV. CONCLUSION

In this paper, we put forth implementation of blockchain in agriculture field for food traceability and supply chain management. First, we have done a detailed analysis on the existing blockchain architectures of Hyperledger Fabric, Hyperledger Sawtooth, Ethereum, Quorum etc., we have chosen Hyperledger sawtooth as it provides better scalability, separation of core system from application domain. A study on existing white papers and implementations in real time yielded results that almost all of them are organization centric while our vision of this implementation is to make the food traceability accessible to the consumer or the end user. We have identified how the data from the IoT devices would be included in the blockchain, the content of the data, providing security for the data in the blockchain through hashing algorithms and gave a detailed explanation on the end to end flow of the blockchain architecture. Because of hardware limitations, we have implemented a prototype for our application and shown how the product data looks like.

In addition, we have identified various vulnerabilities and limitations that can occur in terms of security and various other aspects and came up with solutions in this paper.

XV. ACKNOWLEDGEMENTS

We thank our Professor Dr. My T. Thai and our Teaching Assistant Mr. Truc D. T. Nguyen for their feedback, valuable suggestions and encouragement. This group project is done as part of Blockchain Optimization and Applications course offered in University of Florida, CISE department.

REFERENCES

[1] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. D. Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolic, S. W. Cocco, and J. Yellick, "*Hyperledger fabric: A distributed operating system for permissioned blockchains*," in Proceedings of the 13th ACM SIGOPS European Conference on Computer Systems, Porto, Portugal, 2018.

[2] R. Kamath, "*Food traceability on blockchain: Walmart's pork and mango pilots with IBM*", The JBBA, 1 (1) (2018), p. 3712, 2018.

[3] M. P. Caro, M. S. Ali, M. Vecchio, and R. Giaffreda, ''Blockchain-based traceability in Agri-Food supply chain management: A practical implementation,'' in Proc. IoT Vertical Top. Summit Agricult. (IOT Tuscany), May 2018, pp. 1–4.

[4] K.R. Özyilmaz, A. Yurdakul, "Work-in-progress: Integrating low-power IoT devices to a Blockchain-Based Infrastructure", Proceedings of the 13th ACM International Conference on Embedded Software 2017 Companion, EMSOFT (2017), Özyilmaz and Yurdakul, 2017

[5] Tracking Perishable Goods with Blockchain Technology- *https://www.intel.com/content/www/us/en/internet-of-things/industrial-iot/logistics/connected-logistics-platform-study.html*

[6] B. Peterson. IBM told investors that it has over 400 blockchain clients –including Walmart, Visa, and Nestle. Business Insider UK, http://uk.businessinsider.com

[7] An Agri-food Supply Chain Traceability System for China Based on RFID & Blockchain Technology -F Tian- 2016 13th international conference on service systems …, 2016 - ieeexplore.ieee.org

[8] Jun Lin, et al., "Blockchain and IoT based food traceability for smart agriculture", in: Proceedings of the 3rd International Conference on Crowd Science and Engineering, ACM, 2018, p. 3.

[9] J. Sousa, A. Bessani, and M. Vukolić. A Byzantine fault-tolerant ordering service for the Hyperledger Fabric blockchain platform. In International Conference on Dependable Systems and Networks (DSN), 2018.

[10] Sawtooth: An Introduction K Olson, M Bowman, J Mitchell… - The Linux …, 2018 - hyperledger.org

[11] Performance Modeling of Hyperledger Sawtooth Blockchain B Ampel, M Patton, H Chen - 2019 IEEE International …, 2019 - ieeexplore.ieee.org

[12] Blockchain platforms overview for industrial IoT purposes N Teslya, I Ryabchikov - 2018 22nd Conference of Open …, 2018 - ieeexplore.ieee.org

[13] *Performance evaluation of the quorum blockchain platform A Baliga, I Subhod, P Kamat, S Chatterjee - arXiv preprint arXiv …, 2018 - arxiv.org*

[14] Hyperledger Fabric. http://github.com/hyperledger/fabric.

[15] C. Cachin. "Architecture of the Hyperledger blockchain fabric", In Workshop on Distributed Cryptocurrencies and Consensus Ledgers, 2016.

[16] J. P. Morgan. Quorum whitepaper. https://github.com/jpmorganchase/

[17] Hyperledger. http://www.hyperledger.org.

[18] Hyperledger Fabric. http://github.com/hyperledger/fabric.

[19] Hyperledger Sawtooth. http://sawtooth.hyperledger.org.

[20] ibm-blockchain-enterprise-customers-walmart-visa-nestl-2018-3/, 2018

[21] Proposal for Protocol on a Quorum Blockchain with Zero Knowledge. T Espel, L Katz, G Robin - IACR Cryptology ePrint Archive, 2017 - eprint.iacr.org

[22] Á García-Pérez, A Gotsman, "*Federated byzantine quorum systems*" - 22nd International Conference on …, 2018 - drops.dagstuhl.de

[23] Survey of consensus protocols on blockchain applications LS Sankar, M Sindhu… - 2017 4th International …, 2017 - ieeexplore.ieee.org

[24] Technical aspects of blockchain and IoT HF Atlam, GB Wills - Advances in Computers, 2019 - Elsevier

[25] Ensure traceability in European food supply chain by using a blockchain system G Baralla, A Pinna, G Corrias - … Engineering for Blockchain , 2019 - ieeexplore.ieee.org

[26] Link for Hyperledger Security measures as presented in their official site: *https://www.hyperledger.org/blog/2018/11/23/hyperledger-sawtooth-blockchain-security-part-two*

[27] *https://blog.matesic.info/post/hashbytes-hashing-in-MS-SQL-Serve*

[28] *https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/sql/authorization-and-permissions-in-sql-server*