Hierarchical Blockchain Topologies for Quality Control in Food Supply Chains

Spyros Voulgaris*, Nikos Fotiou*, Vasilios A. Siris*, George C. Polyzos*,
Artemios Tomaras†, Sotiris Karachontzitis†

* Mobile Multimedia Laboratory, Department of Informatics
School of Information Sciences and technology, Athens University of Economics and Business
76 Patision, 10434 Athens, Greece
Email: {voulgaris, fotiou, vsiris, polyzos}@aueb.gr

† Synelixis Solutions S.A.
Perissou 157 & Chalkidos, N. Chalkidona, Athens, Greece
Email: {tomaras,karachontzitis}@synelixis.com

***Abstract*—The use of blockchains to improve product quality and safety control in food supply chains through transparent, trusted, and secure end-to-end traceability frameworks, has received increased attention in the last few years. The use of blockchains, though, does not come at no cost. Poor design of blockchain-based applications can lead to prohibitive costs, intolerable delays, and nonscalable systems. In this work we explore different architectures for blockchain-based traceability and quality control of produce, proposing, evaluating, and comparing four different scenarios. Our evaluation uses public and private Ethereum instances, and assesses the considered architectures in terms of cost and overall throughput.**

I. Introduction

The Food Supply Chain (FSC) sector is currently under intensive transformation to meet the challenges of the supply chain 4.0 vision [1]. The application of emerging technologies, such as the Internet of Things (IoT), big data analytics, autonomous robotics, etc., drive its digitization and adds value to the supply chain management through increased automation, transparency, and product traceability, improved product quality and safety control, and better-quality relationships between suppliers and buyers [2], [3], [4]. Nevertheless, the majority of the proposed solutions still rely on heavily-centralized infrastructures and central control authorities to manage data and offer end-to-end services. As a result, this limits transparency and, by nature, creates major concerns related to data security and integrity, protection of enterprise (sensitive) data, and building of mutual trust between chain members. In this scope, blockchains represent an innovative technological solution not only to support automation in data and transactions management but also to build trust amongst trustless entities [5], [6].

Traceability in FSC has gained considerable importance during the last years, particularly as a means to assure food safety and quality, minimize the production and distribution of poor quality products, and increase consumers' confidence in labels and brands [7], [8], [9]. The implementation of system prototypes that apply blockchain technology in specific FSC scenarios to offer traceability and food safety is an active field. In [10], a product traceability system is developed based on

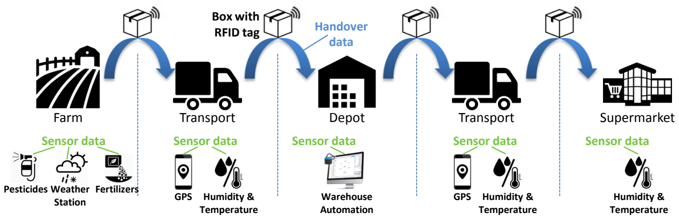

Fig. 1. Food Supply Chain with five stages.

blockchain technology, smart contracts, and IoT, orchestrating product transferring and tracking through the collaboration of a number of smart contracts ensuring that data is transparent and tamper-proof, providing different levels of data query functions for different actors. A similar design is proposed in [11] where blockchain is combined with EPC information service network to reduce data redundancy and enforce corporate privacy and data protection at an enterprise level.

Despite that many prototypes exist that explore the capabilities of blockchain technology in securing product quality and optimizing supply chain management, most of these refer to scenarios where a single blockchain network is deployed to enable transactions and streamline information over the supply chain. An analysis of how more complex and hybrid topologies of blockchain networks can orchestrate the FSC management and affect performance criteria related to the cost, the speed and the throughput of transactions remains to be determined. Combining different types of blockchains, and in particular public and permissioned blockchains, allows different trade-offs in terms of trust decentralization, transparency, privacy, transaction cost, and delay.

Motivated by this, in this paper we propose a fully decentralized, blockchain-based, end-to-end traceability and data logging solution for FSCs and we evaluate its performance for four alternative hierarchical topologies of cooperating ledgers that record data and achieve immutability and transparency. To the best of our knowledge, such an evaluation has not appeared in the literature before.

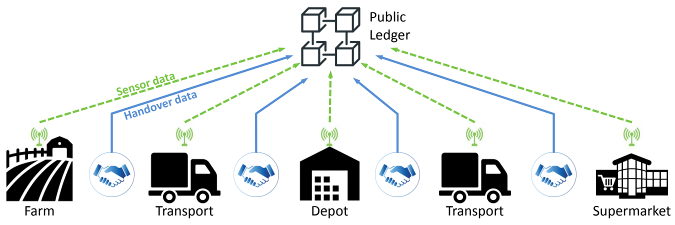

Fig. 2. Scenario 1 – Public ledger

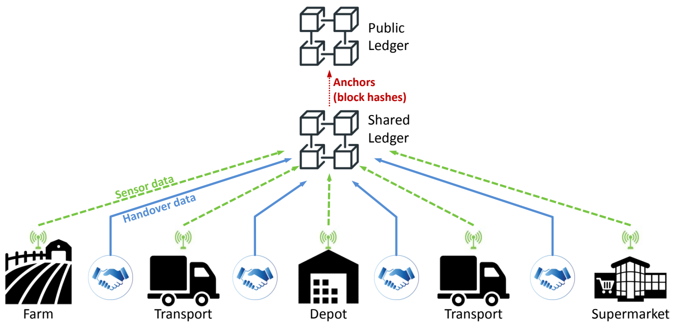

Fig. 3. Scenario 2 – Single shared ledger

II. Food Supply Chain Model

We consider a food supply chain consisting of five *stages*, each being a distinct business entity with separate IT infrastructure. The FSC extends all the way from the farm to the supermarket. All produce are packaged and transported in fixed sized *boxes*, which constitute the universal transport unit in our model. Boxes are equipped with RFID tags, so they can be scanned at any stage. Finally, each stage is equipped with sensors recording the conditions that affect the produce quality.

Figure 1 depicts the FSC model we are considering in our study. Its stages are described below:

Farm: This stage is responsible for managing the field(s), growing, harvesting, and packaging the produce. Their responsibility extends until the produce has been handed over to the transporter at Stage 2.

Transport: This stage is managed by a transportation company that is in charge of transferring packaged produce from Stage 1 (the farm) to Stage 3 (the depot). Produce is transferred in trucks with refrigerated carriages.

Depot: This is where produce is collected, stored, and distributed to retail shops, i.e., super markets. The depot is essentially a storage and distribution center, a large warehouse forming the link between farms and supermarkets.

Transport: This stage is analogous to Stage 2, but it is in charge of transferring produce between Stages 3 and 5.

Supermarket: This is the final stage of the supply chain, where produce is displayed and sold to customers.

Data collected in our FSC model can be split into two types. *Handover data* constitutes the first type. It concerns all data related to individual boxes in the FSC, such as a box' entry in the first stage to start a new transport session, its consecutive handovers between adjacent stages, and its exit from the FSC when this session is complete. This data type includes metadata, such as the time, box state (e.g., its weight), and employees involved in each action. Handover data are recorded *on demand*, as soon as a handover action takes place.

Sensor data forms the second type. It refers to all environmental and location data collected by each individual stage concerning the conditions that may affect produce quality. Sensor data are recorded *periodically*.

Collecting data about a given box requires joining handover and sensor data. The former are used to trace the box' precise itinerary, while the latter serve in assessing the conditions this box has been subjected to.

III. Architecture Space

The goal of the proposed FSC framework is to provide produce traceability in a multi-party business environment of independent entities with potentially conflicting interests. Thus, transparency, data integrity, and building of trust between chain members are key elements in this system. We satisfy these relying on distributed ledger technology.

We consider four different architectures for the proposed FSC framework. Two of them use a single public ledger, while the other two employ hierarchical designs involving a combination of public and private ledgers.

More specifically, we consider the following four architecture scenarios:

Scenario 1 – Public ledger: In this scenario, both handover data as well as sensor readings, are directly stored in a public ledger. This is a straightforward architecture, inherently guaranteeing immutability and trust among chain members. However, as we will see in our evaluation (Section IV), the large volume of data to be stored place an enormous burden in terms of cost and increase delay.

Scenario 2 – Single shared ledger: Architecturally this scenario is identical to the previous one, other than using a shared private ledger (run collaboratively by all chain members) in place of a public ledger, to avoid high costs and delays. A public ledger, however, is still needed to add strong immutability guarantees to the private ledger. More specifically, the private ledger's latest block hash is periodically stored on the public ledger to strengthen the former's immutability, a process referred to as *anchoring*.

Scenario 3 – One private ledger per pair: This scenario employs multiple private ledgers, one per pair of adjacent chain stages. It improves on Scenario 2 with respect to data privacy, as well as overall throughput (Section IV-B). Anchoring to a public ledger is necessary here too, to guarantee immutability.

Scenario 4 – Private storage: This scenario maximizes privacy with respect to sensitive data, by each business entity storing all their data in private storages. These storages need not be ledgers (although they could be). They can be local databases, cloud storages, or even local permissioned ledgers. In the absence of a ledger to store mutually approved handover transactions between adjacent stages, handover records should be *signed* by both stages involved, and stored independently by both of them. In order to guarantee immutability of private storages, each stage is

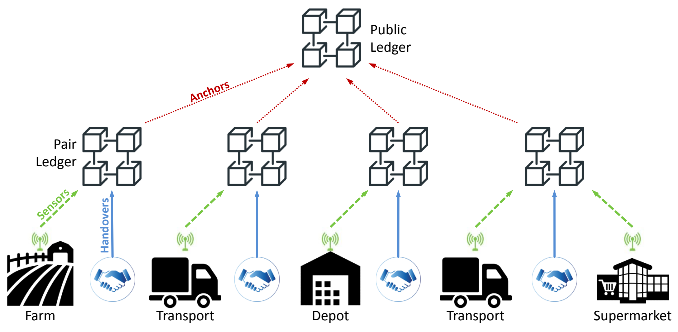

Fig. 4. Scenario 3 – One private ledger per pair

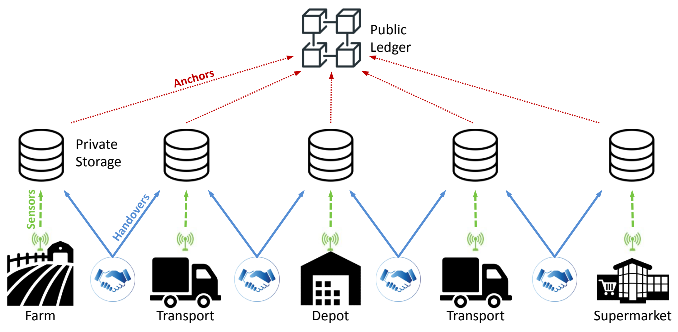

Fig. 5. Scenario 4 – Private storage

responsible to implement anchoring for their private storage by periodically storing a hash digest (Merkle tree root or alternative cryptographic tool of their choice) of their contents.

Clearly, an unlimited number of architecture designs can be devised, either as combinations of the above or by introducing completely new schemes. However, the selected scenarios form a wide range of clearly defined baselines, whose evaluation identifies their strengths and weaknesses, and help us make educated decisions in using them as is or in combinations.

IV. Evaluation

We evaluated and compared all four scenarios, implementing the required smart contracts in Solidity (the mainstream Ethereum language) and deploying them on Ethereum [12]. More specifically, we spawned our own private Ethereum instances for Scenarios 2 and 3, while we used the Ethereum Ropsten testbed as a public ledger.

We took a number of configuration decisions to reduce the parameter space and to allow for a fair comparison. With respect to parameters of our local Ethereum instances, we fixed the average block mining time to 15 seconds, and we set the block gas limit to $10,000,000$ gas units. Both these values reflect the respective values in the public Ethereum main net.

With respect to our implementation, we fixed the data schema of transactions across all four scenarios, as well as the lengths of several fields. We also fixed both the anchoring period and the sensor logging period to 5 minutes, for all stages in all scenarios.

Table I presents the five transaction types that form the basis for the FSC functionality supported in all considered scenarios.

TABLE I. Transaction Types

Transaction	Description
Box entry	Marks the start of a new session for a given box, and assigns this box to a given farmer.
Handover	Hands over a box from one stage to the next, recording the two employee IDs involved, the box weight, and a timestamp.
Box exit	Marks the end of the current session of a given box. Called when the box is emptied and ready for a new session.
Sensor logging	Logs a stage's sensor readings to the ledger. Applied periodically, every 5 minutes.
Anchoring	Stores a private Ethereum's latest block hash into the public Ethereum. Called periodically, every 5 minutes.

Finally, Table II shows, for each of the basic transactions, the fields it stores in the ledger. Note that we assumed uniform format across all four handovers, and uniform length across all five stages' sensor logging transactions.

TABLE II. Transaction Fields

Box Entry	Handover	Box Exit	Sensor Logging	Anchoring
Session ID (256)	Empl ID (32)	Empl ID (32)	Sensor data (256)	Block hash (256)
Empl ID (32)	Empl ID (32)	Time (32)		
Time (32)	Weight (32)			
	Time (32)			

A. Cost Evaluation

We start our evaluation focusing on the cost associated with each scenario for executing smart contracts.

In Ethereum, each call to a smart contract function incurs a cost to the caller. This cost is measured in *gas units*, an Ethereum-specific metric deterministically derived based on the amount of CPU cycles, network traffic, and ledger storage caused by the call. Gas has no fixed monetary value. Instead, when calling a smart contract function you are expected to specify the rate (in ETH, i.e., the Ethereum coin) you are willing to pay per gas unit to whoever mines a block that includes your transaction. This policy serves as an incentive for miners to include your transaction.

In Figure 6 we present the gas cost of an individual call to each basic operation. Implementations are similar across different scenarios. However, box entry and handovers are slightly more costly when multiple ledgers are used (Scenario 3), as the box ID has to be stored anew with each handover. Note that anchoring is not applicable for Scenario 1, as it already stores everything in the public ledger, while transactions (except for anchoring) are not applicable to Scenario 4, as it employs local storages. Non-applicable entries are marked N/A.

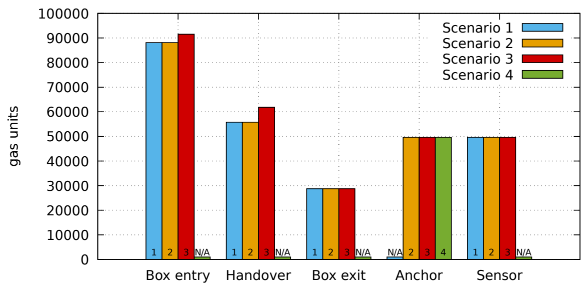

Fig. 6. Ledger cost (in gas) for single execution of basic operations.

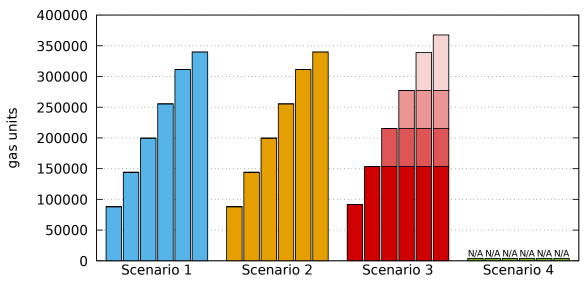

Fig. 7. Aggregate cost (in gas) for a full session of a single box through the FSC. Color coding denotes which ledger each cost fraction refers to.

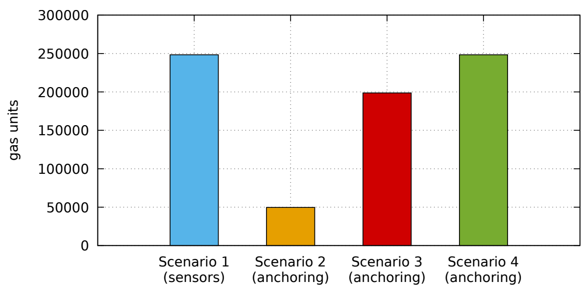

Fig. 8. Public ledger gas costs incurred every 5 minutes due to periodic operations. For Scenario 1 this accounts to the five sites directly recording sensor readings on the public ledger. For the other scenarios it accounts to periodic storing of block hashes on the public ledger (anchoring).

Figure 7 shows the aggregate gas spent for an entire route of a single box through the FSC. Only the costs directly associated with tracing a box are included (i.e., box entry/exit and four handovers), leaving the costs of periodic anchoring and sensor logging out. As expected, the first two scenarios exhibit precisely the same costs, in gas. The third scenario incurs a slightly higher overall gas cost, however this cost is distributed among the four pair ledgers. Finally, ledger operations are not applicable to Scenario 4, as explained above.

It is important to realize that only the gas spent in Scenario 1 translates into actual monetary value (in ETH, and indirectly in EUR). Gas spent on privately managed Ethereum instances can be paid for with ethers collected through very lightweight mining or an ether faucet. However, as we will see in Section IV-B, the gas required by each operation has an effect on the overall throughput of the system in *all* scenarios, as each mined block can include a limited amount of gas.

Figure 8 displays the public ledger gas costs associated with each periodic action (anchoring and sensor logging). Note that Scenario 1 incurs periodic costs only for sensor logging, while anchoring is irrelevant to it as data is recorded directly in the public ledger. In contrast, Scenarios 2, 3, and 4 incur only periodic anchoring costs to the public ledger, as all sensor logging takes places in private ledgers or private storage.

Table III summarizes the costs presented above, giving an estimate of the total cost for a full day's (24 hours) operation in each scenario. We assume a typical daily turnover of 6000 boxes. For the conversion of gas to ether, we assume a price of 10^{-8} ETH per gas unit (referred to as 10 nanoether or 10 gwei). Finally, for the conversion to euros, we assume the price of €200 per Ether, an average price for the last few months. Figure 9 illustrates these costs as a function of the number of boxes processed through the supply chain.

TABLE III. FULL DAY OPERATION COST

	Cost for 6000 boxes			Full-day periodic costs			**Total**
	Gas	Ether	EUR	Gas	Ether	EUR	**EUR**
Scenario 1	2040M	20.4	€4080	71.5M	0.715	€143	**€4223**
Scenario 2	0	0	€0	14.3M	0.143	€28	**€28**
Scenario 3	0	0	€0	57.2M	0.572	€114	**€114**
Scenario 4	0	0	€0	71.5M	0.715	€143	**€143**

B. Throughput Evaluation

Cost is not the only concern when employing blockchains in a real-world application. The time to execute a transaction, and in particular the volume of transactions the system can process in a given time window (i.e., the system's *throughput*), may be a deciding factor on designing its architecture.

The limitations on transaction delay and throughput are explicitly imposed by blockchain rules that prevent the creation of arbitrary-size blocks. In the case of Ethereum, this limit is imposed by means of a maximum amount of gas a miner is willing to pack in a single block. Although this is a miner-specific parameter rather than a globally configured one, in main net public Ethereum it has converged to a maximum of 10 million gas units per block. In that regard, optimizing smart contract functions to spend less gas allows us to fit more calls per block, thus increasing the overall throughput.

To assess the throughput of different scenarios, we ran experiments in which we submitted 1000 boxes at once at the beginning of the FSC, and we let the system run as fast as possible to record the rate at which boxes go through the chain. In these experiments, we emulated all harvesting, transportation, storage, and selling times to be zero, in order to assess the net delays imposed by our tracing system. Figure 10 presents results of these experiments, namely the number of boxes that have traversed the entire FSC as a function of time. The stepwise shape of these plots is due to the grouping of multiple transactions in blocks, which are being generated roughly every 15 seconds. As each box needs six transactions to traverse the entire FSC, no box makes it to the end before the sixth block (at $\sim$90 seconds).

Scenario 3 has a clear advantage over the rest, with a throughput of around 285 boxes per minute, while Scenarios 1 and 2 perform at around 111 and 133 boxes per minute, respectively.

Although intuitively one might expect Scenario 3's throughput to be four times as high as that of Scenario 2, given the fact that transactions are distributed across four private ledgers instead of just one, it is in fact just over twice as high. The reason for this is that in queuing systems throughput is governed by the speed of the slowest component in a chain, i.e., its bottleneck. In this case, Scenario 3's bottleneck is the leftmost pair ledger (shared between stages 1 and 2), as it registers both box-entry transactions for registering new boxes, as well as handovers from stage 1 to stage 2. As shown in Figure 7, the total gas used on the single shared ledger of Scenario 2 (which is what determines how many transactions fit in one block) is a bit over twice as much as the total gas

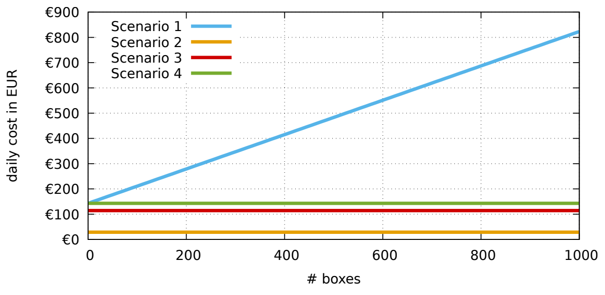

Fig. 9. Total cost (in EUR) for a full day run in each scenario, as a function of the number of boxes transferred. Calculated based on an exchange rate of 200 EUR per 1 ETH, and a gas price of 10^{-8} ETH per gas unit.

spent on the leftmost pair ledger of Scenario 3, a ratio reflected on their respective throughputs.

Finally, Scenario 1 seems to perform slightly worse than Scenario 2. This is because it ran on a public Ethereum instance, Ropsten, where our transactions were not guaranteed to be included in the next block as they were competing against other users' transactions.

V. Conclusion

In this paper we explored the use of blockchain topologies for providing end-to-end traceability of products in a food supply chain, where transparency, data integrity, and record immutability are key factors. We defined and evaluated four different scenarios leveraging single and hierarchical combinations of public and private ledgers.

It turns out that the use of a public ledger to directly store FSC data (Scenario 1) is exorbitantly expensive, prohibiting this use from further consideration. Even worse, it demonstrates the lowest throughput of all options.

Using a single shared private ledger (Scenario 2) to store FSC data and periodically anchoring it to a public ledger is the least expensive solution. However, it demonstrates mediocre throughput, which might prove detrimental if the system scales up with a large number of farms, supermarkets, and transportation companies getting on board.

Splitting a single shared ledger into multiple ones, each serving a subset of the chain (Scenario 3), has a very positive effect on the system's throughput, while cost remains within very reasonable levels. This approach is expected to demonstrate unlimited scalability with respect to the number of businesses joining the FSC, as each new pair may spawn a new ledger. However, more research is needed in that direction in the future.

Finally, using local private storages (Scenario 4) for FSC data appears to be the simplest architecture, as entities may manage them independently without having to maintain a joint ledger. Additionally threshold is not hindered by any ledger operation in this architecture. However, availability is expected to be lower compared to the other scenarios, a topic that is hard to quantify, and which deserves more attention in future work.

These are preliminary research results in the *Secure Open Federation for Internet Everywhere* (*SOFIE*)[1] European Union

[1]https://www.sofie-iot.eu/

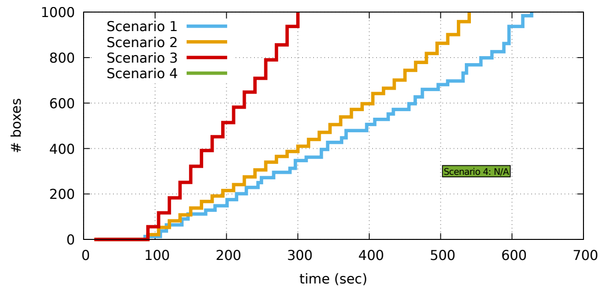

Fig. 10. Box handling throughput.

project. We plan to extend this work to consider FSCs comprising a tree of product routes, rather than a linear chain.

Acknowledgment

This research was supported by the EU funded Horizon 2020 project SOFIE (Secure Open Federation for Internet Everywhere), under agreement No. 779984, and by the Athens University of Economics and Business Research Center.

References

[1] E. A. B. Tjahjono, C. Esplugues and G. Pelaez, "What does Industry 4.0 mean to Supply Chain?" *Procedia Manufacturing*, vol. 13, pp. 1175–1182, 2017.

[2] S. W. K. D. Thoben and T. Wuest, "Industrie 4.0'and Smart Manufacturing–A Review of Research Issues and Application Examples," *International Journal of Automation Technology*, vol. 1, pp. 4–16, 2017.

[3] G. Büyüközkan and F. Göçe, "Literature Review and a Proposed Framework for Future Research," *Computers in Industry*, vol. 97, pp. 157–177, 2018.

[4] E. M. Ben-Daya and Z.Bahroun, "Internet of Things and Supply Chain Management: a Literature Review," *International Journal of Production Research*, vol. 57, pp. 4719–4742, 2019.

[5] K. Francisco and D. Swanson, "The Supply Chain has no Clothes: Technology Adoption of Blockchain for Supply Chain Transparency," *Logistics*, vol. 2, 2018.

[6] N. Kshetri, "Blockchain's Roles in Meeting Key Supply Chain Management Objectives," *International Journal of Information Management*, vol. 39, pp. 80–89, 2018.

[7] van Dorp and Kees-Jan, "Tracking and tracing: A structure for development and contemporary practices," *Logistics Information Management*, 2002.

[8] K. P. Th. Kelepouris and G. I. Doukidis, "Rfid-enabled traceability in the food supply chain," *Industrial Management and Data Systems*, vol. 107, pp. 183–200, 2007.

[9] M. M. Aung and Y. S. Chang, "Traceability in a Food Supply Chain: Safety and Quality Perspectives," *Food Control*, vol. 39, pp. 172–184, 2014.

[10] Y. Z. S. Wang, D. Li and J. Chen, "Smart Contract-Based Product TraceabilitySystem in the Supply Chain Scenario," *IEEE Access*, vol. 7, pp. 115 122–115 133, 2019.

[11] X. P. Q. Lin, H. Wang and J. Wang, "Food Safety Traceability System Based on Blockchain and EPCIS," *IEEE Access*, vol. 7, pp. 20 698–20 707, 2019.

[12] V. Buterin, "A Next-Generation Smart Contract and Decentralized Application Platform," Nov 2013. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper