Double-Edged Sword: Incentivized Verifiable Product Path Query for RFID-enabled Supply Chain

Saiyu Qi*, Yuanqing Zheng†, Xiaofeng Chen*, Jianfeng Ma*, Yong Qi‡
*State Key Laboratory of Integrated Service Networks (ISN), Xidian University, China
†Department of Computing, The Hong Kong Polytechnic University, Hong Kong
‡Department of Computer science and technology, Xian Jiaotong University, China
syqi@xidian.edu.cn, csyqzheng@comp.polyu.edu.hk, xfchen@xidian.edu.cn, jfma@mail.xidian.edu.cn, qiy@mail.xjtu.edu.cn

***Abstract*—Querying the path information of individual products in a supply chain is key to many applications. RFID (Radio-Frequency IDentification) is a main technology to enable product path information query today. With RFID technology, supply chain participants can efficiently track products in transit and record their production information in databases. In this paper, we investigate the following question: how can we conduct privacy-preserving product path information query with verifiability on an RFID-enabled distributed supply chain? We address this question with Double Edged(DE)-Sword, an incentivized verifiable query system. DE-Sword introduces a novel double-edged reputation incentive mechanism to encourage supply chain participants to behave; and couples it with cryptographic primitives and careful protocol design. We evaluate DE-Sword through security analysis and performance experiments. The security analysis shows that DE-Sword guarantees both verifiability and privacy. The experiment results show that DE-Sword achieves low overhead in RFID-enabled supply chain applications.**

I. Introduction

Querying the path information of individual products during their distribution is key to many supply chain applications, such as contamination localization, counterfeit detection, and targeted product recall. RFID is a main technology to enable product path information query today. RFID tags are tiny wireless microchips used to identify physical objects. By attaching RFID tags with unique identifiers to products, supply chain participants can identify and track the products in transit and create RFID-traces to record their production information in databases.

The pharmaceutical industry provides a good example for the importance of product path information queries. The World Health Organization estimates that about 10% of the half trillion dollar pharmaceutical market is counterfeit [1]. In response, some USA states require that the complete history of a drug within the supply chain should be recorded and verifiable [2], [3]. Moreover, the USA Food and Drug Administration (FDA) has identified RFID as the most promising technology to meet this need [4].

The correctness of RFID-product path information query can only be achieved if participants honestly reveal their recorded RFID-traces in the query procedure, which is hard to be satisfied due to commercial and privacy concerns. According to the characteristics of the supply chain applications, participants may subvert the path information query to reap illegal benefits. Considering the existence of dishonest participants, verifiable RFID-product path information queries could benefit the supply chain applications by enhancing the correctness of the product path information.

Dishonest participants could exist, for example, in a contamination localization application, where a product quality administration (PA) agency needs to query the path information of some bad products. With the information, the PA agency can quickly locate the contamination source and recall other affected products. However, dishonest participants may alter their RFID-traces or even hide the existence of them to deny their liabilities, so that they can escape from economic loss and reputation degradation. For instance, in the horsemeat scandal happened in 2013 [5], all of the involved supply chain participants denied their liabilities and accused their up/downstream partners.

Previous works [6]–[10] assume that supply chain participants are honest and mainly focused on storage and query efficiency of product RFID-traces. Indeed, querying product path information from dishonest participants is challenging as a query issuer does not know which participants created and stored RFID-traces during product distribution. Traditional cryptographic schemes cannot be directly used to ensure the query correctness in the presence of dishonest participants. In specific, their security guarantees rely on a honest-data owner model while in our scenario, participants are the data owners (owners of RFID-traces) and may be dishonest.

In this paper, we present Double Edged(DE)-Sword, an incentivized verifiable query system to enable privacy-preserving product path information query with verifiability. Instead of enforcing supply chain participants, DE-Sword adopts a reputation-based incentive mechanism to encourage them to behave, and thus has a volunteer nature. A trustworthy query proxy maintains reputation scores for supply chain participants, answers product path information queries for supply chain applications, and updates the reputation scores of the queried participants.

At a high level, DE-Sword is separated into two phases: a distribution phase and a subsequent query phase. The participants commit their RFID-traces into a product ownership credential (POC) list and submit it to the proxy in the distribution phase, who then uses it to issue product path information queries in the query phase. The goal of DE-Sword is to encourage the participants to construct correct POC list so that the proxy can later conduct verifiable product path information queries. DE-Sword comprises three novel aspects:

DE-Sword. To the best of our knowledge, DE-Sword is the first work to support secure product path information query in the presence of dishonest participants. It gives a complete solution including system model, threat model, system design and analysis. We hope that DE-Sword is a start point to investigate secure RFID-enabled supply chain query.

Double-edged reputation award strategy. The proxy adopts a double-edged reputation award strategy according to the quality of the queried products. This strategy gives a double-edged reputation incentive to encourage the participants to construct correct product ownership credentials (POC) list in the distribution phase.

Cryptographic construction. DE-Sword uses a novel cryptographic primitive named zero knowledge elementary database (ZK-EDB) [11], [12] to build the critical components of double-edged reputation award strategy, including POC list construction in the distribution phase and good/bad product path information query in the query phase.

We argue that DE-Sword is technically and economically viable. The reputation scores maintained by the proxy are based on the quality of queried products, but not advertisement or third-party recommendation, and can be publicly accessed by customers. We believe that such a reputation-based incentive mechanism is reliable for participants to build trustworthy reputations to improve customer confidence and encourage honest participation.

II. Background and overview

In this section, we first present the system model and security requirements. We then give an overview of DE-Sword.

A. System model

An RFID-enabled supply chain consists of participants and products. Each product is labeled with an RFID tag containing a unique product identifier. RFID tags have limited memory and can be remotely identified by RFID readers. Once a product flows through a participant, the participant can use its RFID reader to read the product identifier and create in its database an RFID-trace to record the production information. The trace can include details about the product such as process operation, ingredients, parameters, etc.

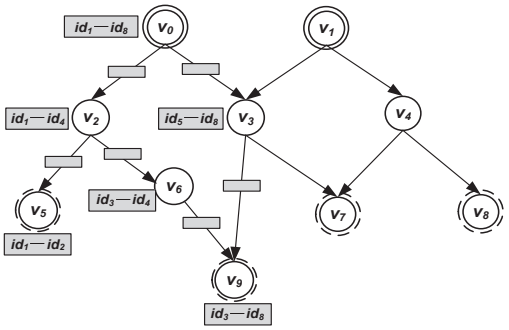


Figure 1: The relationships among supply chain participants.

The relationships among supply chain participants can be represented with a digraph as in Figure 1. A directed edge from one participant v_i to another participant v_j means that v_j is a child of v_i. The edge represents that a product might proceed to v_j after being processed by v_i. A participant is an initial participant if it has no incoming edges and a participant is a leaf participant if it has no outcoming edge. The digraph can be dynamic: new participants/directed edges can be added into the digraph and old participants/directed edges can be removed from the digraph.

Products are processed and distributed in distribution tasks. A distribution task involves a procedure where products are distributed from an initial participant to several leaf participants following directed edges. In the task, when a participant v_i receives a batch of products, it processes the batch and creates an RFID-trace for each product in the batch. The RFID-trace of a product *id* is $t_{v_i}^{id} = (id,\ da_{v_i}^{id})$, where *id* is the product identifier and $da_{v_i}^{id}$ is the information about *id*. After that, v_i divides the batch into multiple smaller batches and distributes them to its children. When a child receives its product batch, it similarly processes and distributes the batch until all the products reach leaf participants. In a distribution task, each product flows through a path in the supply chain and each participant on the path creates an RFID-trace for the product in its database. The collection of all these RFID-traces forms the path information of the product.

Figure 1 illustrates an example consisting of 10 participants (i.e., v_0, v_1,..., v_9). Initial participants (i.e., v_0 and v_1) are encircled with real lines and leaf participants (i.e., v_5, v_7, v_8 and v_9) are encircled with dotted lines. In a distribution task, 8 products id_1-id_8 are distributed from the initial participant v_0 to the leaf participants v_5 and v_9. The path information of id_1 is $\{t_{v_0}^{id_1}, t_{v_2}^{id_1}, t_{v_5}^{id_1}\}$, which indicates that the product id_1 followed the path $v_0 \rightarrow v_2 \rightarrow v_5$ in the supply chain.

B. Security requirements

DE-Sword aims to support product path information query with verifiability and privacy. Verifiability means that the

correctness of a query should be verifiable. Correctness means that the query result should contain the exact path information of the product.

DE-Sword aims to guarantee verifiability and privacy as follows:

Verifiability: In a product path information query, the violation of correctness by any dishonest participants should be detected.

Privacy: participants should not reveal non-trivial information about RFID-traces of non-queried products to any third party.

C. Overview of DE-Sword

DE-Sword relies on a trustworthy query proxy (e.g., the USA Food and Drug Administration), which is responsible to monitor the product quality. The main responsibilities of the proxy is to maintain reputation scores for supply chain participants, issue product path information queries for supply chain applications (e.g., contamination localization application), and update reputation scores of the queried participants. The scores reflect the quality of the queried products, and can be publicly accessed by customers.

At a high level, DE-Sword is divided into two phases, a distribution phase and a subsequent query phase. The distribution phase could include several distribution tasks. Here we assume that it only contains one distribution task for simplicity. In the distribution phase, all the involved participants commit their RFID-traces into product ownership credentials (POCs) to form a POC list. POC is a compact commitment of a set of RFID-traces. With a POC, one can prove whether an RFID-trace is committed into it. In the query phase, the proxy issues product path information queries by requiring participants to return RFID-traces of queried products and uses the POC list to verify the responses. The goal of DE-Sword is to encourage participants to construct correct POC list so that the proxy can conduct verifiable product path information queries.

Design challenge: The design challenge of DE-Sword is how to ensure the correctness of POCs generated by potentially dishonest participants. With traditional cryptographic scheme, such as digital signature scheme, a participant v can construct a POC as a set of signed messages. For each of the RFID-traces t_v^{id}, v (1) generates a signature σ_t for t_v^{id} and (2) generates a signature σ_v for $v||id||\sigma_t$ where v is the identity of the participant and id is the product ID. v then submits all the signed messages $(v||id||\sigma_t,\ \sigma_v)$ as a POC to the proxy. The proxy collects the POCs from all the involved participants and stores them as a POC list. Later in the query phase, suppose the proxy wants to query the path information of a product $\overline{id}$. For each POC in the POC list containing a signed message $(v||\overline{id}||\sigma_t,\ \sigma_v)$, the proxy sends $\overline{id}$ to v to request an RFID-trace $t_v^{\overline{id}}$. There are two cases. (1) v returns a response: the proxy checks if σ_t is a valid signature of the response; (2) v refuses to return a response: as $(v||\overline{id}||\sigma_t,\ \sigma_v)$ contains a valid signature σ_v of v on $v||\overline{id}||\sigma_t$, the proxy confirms that v indeed processed $\overline{id}$ and is dishonest. Obviously, if all the POCs are correctly constructed, the proxy can easily detect the dishonest participants' violation of correctness during the query.

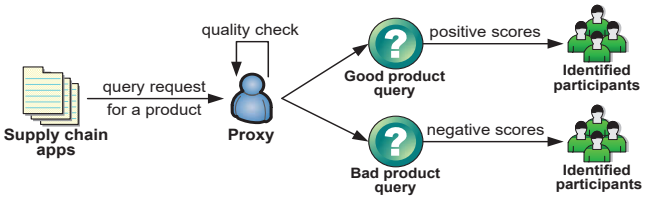


Figure 2: Double-edged reputation award strategy.

Unfortunately, dishonest participants may submit incorrect POCs, either due to the malicious goal of violating query correctness or due to the worry of sensitive production message leakage. For instance, they may hide their original signed messages by submitting some fake but seemingly correct messages to avoid to be queried in the later query phase. Actually, the security guarantees of traditional cryptographic schemes (e.g., signature/hash/mac/commitment schemes) stem from a honest-data owner model, where a data owner needs to honestly generate crypto-tags (signatures/hashs/macs/commitments) over its data. With the correct crypto-tags, a verifier can later detect dishonest behaviours violating the correctness of the data. In our scenario, however, participants are the owners of RFID-traces and may be dishonest. A participant can just generate incorrect crypto-tags (POC) at first, disabling the security guarantees of the cryptographic schemes in the later verifications.

Our idea: Instead of relying on cryptographic schemes, DE-Sword adopts a reputation-based incentive to encourage participants to behave. The key insight behind DE-Sword is that in the distribution phase, participants cannot predict which products will be queried and the actual quality of them in the subsequent query phase. In real distribution procedure, products suffer a small risk of being bad, i.e., a product being good with an overwhelming probability while being bad with a relatively small probability, e.g., due to unpredicted mistakes or accidents.

Based on this insight, the proxy adopts a double-edged reputation award strategy. As illustrated in Figure 2, to issue a product path information query, the proxy issues a good/bad product path information query towards the supply chain according to the quality of the queried product. During the query, the proxy identifies the involved participants, asks them to return RFID-traces of the product, and verifies their responses by the POC list. At the end of the query, the proxy assigns positive/negitive reputation scores to the identified participants. To guarantee fairness, the proxy can

give diverse positive/negitive reputation scores based on the responsibilities of the identified participants. The proxy can also adjust the query frequency by sampling products from the market, and issue queries for them by itself.

Such a double-edged reputation award strategy gives a double-edged reputation incentive to encourage participants to behave in the POC list construction in the distribution phase as illustrated in Figure 3. Without the prior knowledge about which products will be queried and the actual quality of them in the subsequent query phase, the participants cannot confirm if they can acquire definite reputation benefits by deleting/adding RFID-traces when generating their POCs. Considering a participant *v* with *n* RFID-traces $\{t_v^{id_i}\}_{i=1}^n$. If *v* deletes the RFID-trace of *id*($id \in \{id_i\}_{i=1}^n$), later in the query phase, if *id* is queried, *v* will not be identified by the proxy. If *id* is bad, *v* will avoid the risk to be given negative reputation scores on one edge. However, if *id* is good, *v* will lose the opportunity to earn positive reputation scores on another edge. If *v* adds a fake RFID-trace for id ($id \notin \{id_i\}_{i=1}^n$), later in the query phase, if id is queried, *v* will be identified by the proxy. If id is good, *v* will win positive reputation scores on one edge. However, if id is bad, *v* will be given negative reputation scores on another edge.

DE-Sword adopts a novel cryptographic primitive named zero knowledge elementary database (ZK-EDB) [11], [12] to construct POCs. With ZK-EDB, a participant *v* could commit its RFID-traces $\{t_v^{id_i}\}_{i=1}^n$ into a compact commitment as its POC. The commitment does not reveal non-trivial information about the committed RFID-traces. Moreover, *v* can only generate (1) ownership proof for a product *id* (*id* $\in \{id_i\}_{i=1}^n$), which states that there is an RFID-trace t_v^{id} indexed by *id* committed in the commitment; and (2) non-ownership proof for a product *id* (*id* $\notin \{id_i\}_{i=1}^n$), which states that there is no RFID-trace indexed by *id* committed in the commitment.

DE-Sword also designs verifiable good/bad product path information query based on ZK-EDB. Due to the difference of good/bad product case, participants may adopt diverse dishonest behaviours to violate the query correctness. DE-Sword considers such diversity and guarantees verifiability with correct POC list. To query a product *id*, the proxy identifies the participants who have processed *id* based on two observations: (1) they cannot show non-ownership proofs for *id*; and (2) they can show ownership proofs for *id*. The proxy uses the first observation in bad product case and the second observation in good product case.

III. Security guarantees of DE-Sword

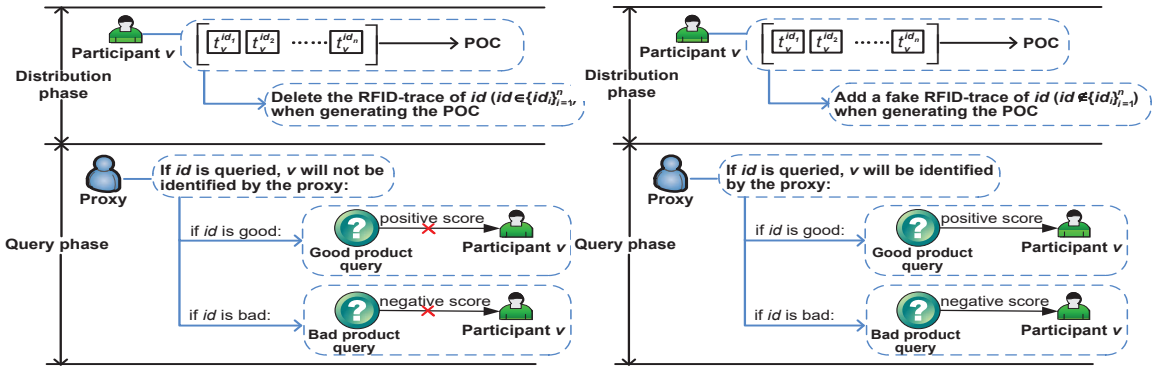
With the clarity of the work flow of DE-Sword, we consider potential dishonest behaviours of participants in POC list construction in the distribution phase and in product path information query in the subsequent query phase, and describe the corresponding security guarantees provided by DE-Sword. With the combination of these security guarantees, DE-Sword achieves product path information query with verifiability and privacy.

A. Distribution phase

Threat model: In the distribution task, a participant *v* with *n* RFID-traces $\{t_v^{id_i}\}_{i=1}^n$ may adopt dishonest behaviours to generate incorrect POC to violate query correctness or prevent sensitive production information leakage: (1) Deletion: delete an RFID-trace of *id* (*id* $\in \{id_i\}_{i=1}^n$). *v* may adopt this behaviour to hide the existence of the RFID-trace to avoid to be queried later. (2) Addition: add a fake RFID-trace of *id* ($id \notin \{id_i\}_{i=1}^n$). After a deletion, *v* may adopt this behaviour to maintain the POC in a seemingly correct format. (3) Modification: modify the information part da^{id} of an RFID-trace of *id* (*id* $\in \{id_i\}_{i=1}^n$). *v* may adopt this behaviour due to the worry of sensitive production information leakage from the POC. Finally, the participants on a same path may coordinate to adopt same types of dishonest behaviours to escape detection. For example, all the participants on a path may delete the RFID-traces of their processed products so that the proxy cannot query any RFID-traces about these products in the supply chain.

Security guarantees: DE-Sword relies on double-edged reputation incentive to discourage the first two types of dishonest behaviours and the security of ZK-EDB to discourage the last type of dishonest behaviour in POC list construction. The security guarantees also applied to multiple coordinated participants on a same path.

- *Deletion:* If *v* adopts a deletion behaviour, it will face a double-edged situation in the subsequent query phase: avoid a risk to be given negative reputation scores on one edge but lose an opportunity to earn positive reputation scores on another edge. As *v* cannot confirm to acquire definite reputation benefits by adopting deletion behaviour, it is discouraged from doing so.
- *Addition:* If *v* adopts an addition behaviour, it will face a double-edged situation in the subsequent query phase: win an opportunity to earn positive reputation scores on one edge but suffer a risk to be given negative reputation scores on another edge. As *v* cannot confirm to acquire definite reputation benefits by adopting addition behaviour, it is discouraged from doing so.
- *Modification:* Each participant uses ZK-EDB to construct its POC. Due to the design of ZK-EDB, the POC does not reveal non-trivial information about the committed RFID-traces. Moreover, in later product path information queries, the identified participants only return the RFID-traces of queried products as well as ownership/non-ownership proofs. Due to the design of ZK-EDB, these messages do not reveal non-trivial information about the RFID-traces of non-queried products.



(a) Delete an RFID-trace

(b) Add a fake RFID-trace

Figure 3: Double-edged reputation incentive.

B. Query phase

Threat model: To query a product *id*, the proxy asks the participants one after another and collects RFID-traces of *id* from them. To ask a participant *v*, the proxy first identifies if *v* processed *id*. If yes, the proxy then collects an RFID-trace of *id* from *v*. Finally, the proxy asks *v* the identity of the next participant that processed *id*. According to the quality of the queried product, *v* may adopt dishonest behaviors to maximize its reputation scores and affect the scores of the next participant: (1) Claim non-processing: in bad product case, *v* may claim that it did not process *id* although it done so. (2) Claim processing: in good product case, *v* may claim that it processed *id* although it did not do so. (3) Return wrong RFID-trace: in both bad/good product cases, if *v* has been identified, *v* may return a wrong RFID-trace to the proxy. (4) Return the identity of a wrong participant: in both bad/good product cases, *v* may return the identity of a wrong participant. Finally, the participants on a same path may coordinate to adopt same types of dishonest behaviours to escape detection. For example, all the participants on a path may return wrong RFID-traces to let the proxy to collect wrong while seemingly correct path information of a product.

Security guarantees: Given the correct POC list, DE-Sword relies on the security of ZK-EDB to detect all the types of dishonest behaviours in product path information query. The security guarantees also applied to multiple coordinated participants on a same path.

- *Claim non-processing:* If *v* processed *id*, it would commit an RFID-trace indexed by *id* into its POC, and thus cannot be able to generate a non-ownership proof for *id*. As a result, the proxy can ask *v* to return a valid non-ownership proof for verification if *v* claim that it did not process *id*.
- *Claim processing:* If *v* did not process *id*, it would not commit an RFID-trace indexed by *id* into its POC, and thus cannot be able to generate an ownership proof for *id*. As a result, the proxy can ask *v* to return a valid ownership proof for verification if *v* claim that it processed *id*.
- *Return wrong RFID-trace:* If *v* processed *id*, it would commit an RFID-trace indexed by *id* into its POC, and thus cannot be able to generate fake ownership proof for *id*. As a result, the proxy can ask *v* to return a valid ownership proof and recover the valid RFID-trace from it.
- *Return the identity of a wrong participant:* There are two cases. (1) *v* returns the identity of a participant that did not process *id*. In this case, the proxy can identify if the next queried participant processed *id* by asking it to return ownership/non-ownership proof for verification. (2) *v* returns the identity of a participant that processed *id*, but is not the next participant that processed *id*. In this case, the proxy can check if the next queried participant is one of the children of *v* in the POC list.

IV. System Design

In this section, we start with an introduction of ZK-EDB. We next describe the distribution phase and the query phase of DE-Sword. Finally, we show how to extend DE-Sword to the case that the distribution phase contains multiple distribution tasks.

A. Background: ZK-EDB

In DE-Sword, the construction of POC is built on a cryptographic primitive called zero knowledge elementary database (ZK-EDB) [11], [12], which allows a prover to commit to a database while being able to non-interactively prove whether a key-value pair exists in the database without revealing any further information about the database (including the cardinality of the database).

An elementary database D (EDB) consists of a set of pairs $(x, y) \subset \{0, 1\}^\star \times \{0, 1\}^\star$, where x is a key and y is a value. The support $[D]$ of D is the set of $x \in \{0, 1\}^\star$ with

Table I: The design of POC scheme

PS-Gen(λ) $\rightarrow$ ps:
—- $\sigma \leftarrow$ CRS-Gen(λ);
—- $ps \leftarrow \sigma$;
—- Output ps.

POC-Agg(D_{v_i}, v_i, ps) $\rightarrow$ (POC_{v_i}, $DPOC_{v_i}$):
—- (Com, Dec) $\leftarrow$ EDB-commit(D_{v_i}, ps);
—- $POC_{v_i} \leftarrow v_i||Com$;
—- $DPOC_{v_i} \leftarrow Dec$;
—- Output a POC pair (POC_{v_i}, $DPOC_{v_i}$).

POC-Proof(ps, POC_{v_i}, $DPOC_{v_i}$, D_{v_i}, id) $\rightarrow$ $o\pi_{id}$/$no\pi_{id}$:
—- **If** there is an $t^{id}_{v_i} \in D_{v_i}$:
—- ZK-$\pi_{id} \leftarrow$ EDB-proof(ps, POC_{v_i}, $DPOC_{v_i}$, id);
—- $o\pi_{id} \leftarrow$ Ow-proof$||ZK$-π_{id};
—- Output $o\pi_{id}$.
—- **Else**:
—- ZK-$\pi_{id} \leftarrow$ EDB-proof(ps, POC_{v_i}, $DPOC_{v_i}$, id);
—- $no\pi_{id} \leftarrow$ Now-proof$||ZK$-π_{id};
—- Output $no\pi_{id}$.

POC-Verify(ps, POC_{v_i}, id, $o\pi_{id}$/$no\pi_{id}$) $\rightarrow$ $t^{id}_{v_i}$/valid/bad:
—- **For** $o\pi_{id}$:
—- **If** $y \leftarrow$ EDB-Verify(ps, POC_{v_i}, id, $o\pi_{id}$) $\wedge$ $y \neq \perp$:
—- $t^{id}_{v_i} \leftarrow$ (id, y);
—- Output $t^{id}_{v_i}$.
—- **Else** Output bad.
—- **For** $no\pi_{id}$:
—- **If** $y \leftarrow$ EDB-Verify(ps, POC_{v_i}, id, $no\pi_{id}$) $\wedge$ $y = \perp$:
—- Output valid.
—- **Else** Output bad.

an associated value $y \in \{0, 1\}^\star$ such that (x, y) $\in D$. For $x \notin [D]$, $D(x) = \perp$. For $x \in [D]$, the associated value $y = D(x)$ must be unique: if both (x, y) and (x, y') $\in D$, then $y = y'$. A ZK-EDB scheme is formally defined by a tuple of algorithms (CRS-Gen, EDB-commit, EDB-proof, EDB-Verify) as follows:

- CRS-Gen(λ) $\rightarrow$ σ: On input a security parameter λ, outputs a common reference string σ.
- EDB-commit(D, σ) $\rightarrow$ (Com, Dec): On input the database D and the common reference string σ, it outputs a commitment/de-commitment pair (Com, Dec).
- EDB-proof(σ, Com, Dec, x) $\rightarrow$ ZK-π_x: On input the common reference string σ, the commitment/de-commitment pair (Com, Dec) and a key $x \in \{0, 1\}^\star$, outputs a proof ZK-π_x.
- EDB-Verify(σ, Com, x, ZK-π_x) $\rightarrow$ y/*bad*: on input the common reference string σ, a commitment Com, a key x and a proof ZK-π_x, outputs either a value y (which is $\perp$ if $x \notin [D]$) if it is convinced that $D(x) = y$ or *bad* if it believes that the proof is invalid.

B. Distribution phase

In the distribution task, the involved participants commit their RFID-traces into POCs and construct a POC list for the proxy to store. We next describe the design of POC and POC list.

POC: A POC scheme consists of four algorithms (PS-Gen, POC-Agg, POC-Proof, POC-Verify) as shown in Table 1. Given a security parameter λ, PS-Gen() generates a public parameter *ps*, which is needed as input by the other three algorithms. Given a set D_{v_i} of RFID-traces, a participant ID v_i, and the public parameter *ps*, POC-Agg() aggregates D_{v_i} into a product ownership credential POC_{v_i} and generates a POC-de-commitment $DPOC_{v_i}$. Given the public parameter *ps*, a POC POC_{v_i}, a POC de-commitment $DPOC_{v_i}$, a set D_{v_i} of RFID-traces, and a product ID *id*, POC-Proof() generates either (1) an ownership proof for *id* if there exists an RFID-trace $t^{id}_{v_i}$ indexed by *id* $\in D_{v_i}$; or (2) a non-ownership proof for *id* if there does not exist an RFID-trace $t^{id}_{v_i}$ indexed by *id* $\in D_{v_i}$. Given the public parameter *ps*, a POC POC_{v_i}, a product ID *id*, and an ownership/non-ownership proof $o\pi_{id}$/$no\pi_{id}$, POC-Verify() verifies (1) for $o\pi_{id}$: if it is valid, output an RFID-trace $t^{id}_{v_i}$. Else, output bad. (2) for $no\pi_{id}$: if it is valid, output valid. Else, output bad.

POC list: A POC list is a sub-digraph with each vertex storing the POC of a involved participant. The sub-digraph reflects the relationships of all the involved participants in the distribution task. Given the design of POC scheme, participants construct POC list as follows. Suppose there are n involved participants $\{v_i\}^n_{i=1}$ in the distribution task. These participants need a public parameter *ps*, generated by a honest party, to generate their POCs. To do so, the initial participant v_1 requests *ps* from the proxy and broadcasts it to all the other participants $\{v_i\}^n_{i=2}$.

During the distribution task, each participant v_i commits its set D_{v_i} of RFID-traces into a POC POC_{v_i} and generates a POC-de-commitment $DPOC_{v_i}$. v_i stores $DPOC_{v_i}$ in its database to generate ownership/non-ownership proofs in the subsequent query phase. At the end of the distribution task, each v_i transmits POC_{v_i} to its parents to construct POC pairs. A POC pair (POC_{v_i}, POC_{v_j}) is a pair of POCs with the relation that v_i is the parent of v_j. Finally, each v_i transmits its constructed POC pairs to v_1, who composes all the received POC pairs as a POC list (*ps*, {(POC_{v_i}, POC_{v_j})}), and transmits it to the proxy.

C. Query phase

Upon receiving a query request for a product id from a supply chain application, the proxy adopts double-edged reputation award strategy to issue the query. The goal of the proxy is to give double-edged reputation incentive, so that participants are encouraged to behave in the POC list construction in the prior distribution phase. In specific, the proxy checks the quality of *id* (good/bad), issues a good/bad product path information query for *id*, and assigns positive/negative reputation scores to the identified participants.

The query consists of a sequence of query interactions. In each interaction, the proxy identifies if the queried participant *v* processed *id*. If yes, the proxy further asks *v* to

return an RFID-trace of *id* and verifies it. At the end of the interaction, the proxy asks *v* the next participant *v'* that processed *id*, and starts a new query interaction with *v'*. The query interaction is conducted repeatedly until a leaf participant is reached. The query interaction between the proxy and v is shown as follows.

Good product case: In this case, a dishonest v may pretend that it processed id to earn positive reputation score. To detect this dishonest behaviour, the proxy requests v to return a valid ownership proof for verification if v claims that it processed id.

- *Step 1:* The proxy sends a request (query request, id, POC_v) to v. If v did not process id, it would not commit an RFID-trace t_v^{id} into POC_v and cannot generate a valid ownership proof $o\pi_{id}$ for id.
- *Step 2:* As a result, if v returns a valid ownership proof $o\pi_{id}$, the proxy identifies that v processed id and recovers an RFID-trace t_v^{id} from $o\pi_{id}$. Otherwise, it identifies that v did not process id.
- *Step 3:* If the proxy identifies that v processed id, it asks v the identity of next participant that processed id. Upon receiving the identity v', the proxy searches a POC-pair (POC_v, $POC_{v'}$) from the POC list, and sends a request (query request, id, $POC_{v'}$) to v' to start the next interaction.

Bad product case: In this case, a dishonest v may deny that it processed id to avoid being given negative reputation score. To detect this dishonest behaviour, the proxy requests v to return a valid non-ownership proof for verification if v claims that it did not process id.

- *Step 1:* The proxy sends a request (query request, id, POC_v) to v. If v processed id, it would commit an RFID-trace t_v^{id} into POC_v and cannot generate a valid non-ownership proof $no\pi_{id}$ for id.
- *Step 2:* As a result, if v returns a valid non-ownership proof $no\pi_{id}$, the proxy identifies that v did not process id. Otherwise, it identifies that v processed id and requires v to reveal a valid ownership proof $o\pi_{id}$ to recover an RFID-trace t_v^{id}.
- *Step 3:* if the proxy identifies that v processed id, it asks v the identity of next participant that processed id. Upon receiving the identity v', the proxy searches a POC-pair (POC_v, $POC_{v'}$) from the POC list, and sends a request (query request, id, $POC_{v'}$) to v' to start the next interaction.

D. Multi-distribution tasks

Before the query phase, multiple product distribution tasks may have happened in the supply chain and the distribution phase should include all of these tasks. In this case, the proxy maintains a POC-queue for each initial participant in the supply chain. When a initial participant $\bar{v}$ submits a POC-list, the proxy retrieves the POC $POC_{\bar{v}}$ of $\bar{v}$ and the public parameter *ps* from the list, and inserts the pair (*ps*, $POC_{\bar{v}}$) into the POC-queue of $\bar{v}$. To start a good/bad product path information query for a product id, the proxy needs to first query each initial participant with its POC-queue to check who processed id.

Good product case: In this case, a dishonest initial participant $\bar{v}$ who did not process id may pretend that it processed id to earn positive reputation score. The proxy requests $\bar{v}$ to return a valid ownership proof for id with a POC in the POC queue for verification. As the dishonest $\bar{v}$ did not aggregate an RFID-trace into any POC in the POC-queue, it cannot generate a valid ownership proof for id with any POC in the POC-queue. As a result, if $\bar{v}$ returns a valid ownership proof $o\pi_{id}$ with a pair (*ps*, $POC_{\bar{v}}$) belonging to the POC-queue, the proxy confirms that $\bar{v}$ processed id and recovers an RFID-trace $t_{\bar{v}}^{id}$ from $o\pi_{id}$ and (*ps*, $POC_{\bar{v}}$). If $\bar{v}$ is identified, the proxy uses the POC list containing $POC_{\bar{v}}$ to continue the query as in the good product case in the above subsection.

Bad product case: In this case, a dishonest initial participant $\bar{v}$ who did process id may deny that it processed id to avoid being given positive reputation score. The proxy requests $\bar{v}$ to return a valid non-ownership proof for id with each POC in the POC queue for verification. As the dishonest $\bar{v}$ committed an RFID-trace $t_{\bar{v}}^{id}$ into a POC $POC_{\bar{v}}$ in the POC-queue, it cannot generate a valid non-ownership proof for id with $POC_{\bar{v}}$. As a result, if $\bar{v}$ cannot return a valid non-ownership proof with a pair (*ps*, $POC_{\bar{v}}$) belonging to the POC-queue, the proxy identifies that $\bar{v}$ processed id, and requires $\bar{v}$ to reveal a valid ownership proof $o\pi_{id}$ for id with (*ps*, $POC_{\bar{v}}$) to recover an RFID-trace $t_{\bar{v}}^{id}$. If $\bar{v}$ is identified, the proxy next uses the POC list that contains $POC_{\bar{v}}$ to continue the query as in the bad product case in the above subsection.

V. Security analysis

We prove the security guarantees of DE-Sword in the two phases as stated in Section II.D. Before the proof, we first introduce some security properties of ZK-EDB which are used in the proof.

Definition 1 ZK-EDB soundness. A malicious prover should not be able to prove false statements even if it provides a maliciously chosen public key. More formally, $\forall\ x \in \{0,1\}^*$ and $\forall$ efficient algorithms P':

$$Pr\begin{cases} \sigma \leftarrow \mathsf{CRS\text{-}Gen}(\lambda); \\ (Com, x, \pi_x, \pi'_x) \leftarrow P'(\sigma, D); \\ \mathsf{EDB\text{-}Verify}(\sigma, Com, x, \pi_x) = y \neq bad \\ \wedge\ \mathsf{EDB\text{-}Verify}(\sigma, Com, x, \pi'_x) = y' \neq bad \wedge y \neq y'; \end{cases}$$

The probability *Pr* is negligible. From the soundness of ZK-EDB, we can easily derive following two claims.

Claim 1. Pr is negligible when (*POC*, id, $o\pi_{id}$, $no\pi_{id}$) outputted by P' satisfies the following conditions:

$$\text{POC-Verify}(ps,\ POC,\ id,\ o\pi_{id}) = t_{v_i}^{id}$$
$$\wedge\ \text{POC-Verify}(ps,\ POC,\ id,\ no\pi_{id}) = \text{valid};$$

This claim states that a participant v cannot generate a valid ownership proof $o\pi_{id}$ and a valid non-ownership proof $no\pi_{id}$ for a product id at the same time.

Claim 2. Pr is negligible when (*POC*, id, $o\pi_{id}$, $o\pi'_{id}$) outputted by P' satisfies the following conditions:

$$\text{POC-Verify}(ps,\ POC,\ id,\ o\pi_{id}) = t_{v_i}^{id}$$
$$\wedge\ \text{POC-Verify}(ps,\ POC,\ id,\ o\pi'_{id}) = t_{v_i}'^{id} \wedge t_{v_i}^{id} \neq t_{v_i}'^{id};$$

This claim states that a participant v cannot generate two valid ownership proofs $o\pi_{id}$ and $o\pi'_{id}$ for a product id with two different RFID-traces $t_{v_i}^{id}$ and $t_{v_i}'^{id}$ at the same time.

Definition 2 ZK-EDB zero-knowledge. In an interaction between a verifier and a prover holding a database *D*, the verifier sends a key *x* to the prover, who returns a proof *ZK*-π_x to prove the value *D*(*x*). In this procedure, the verifier learns only *D*(*x*) from the prover.

A. Security guarantees in distribution phase

DE-Sword relies on double-edged reputation incentive to discourage the first two types of dishonest behaviours and the zero-knowledge property of ZK-EDB to discourage the last type of dishonest behaviour in POC list construction.

Deletion: If v conducts a deletion behaviour for any of its processed products, it will face a double-edged situation: avoiding a risk of giving negative reputation score but lose an opportunity of giving positive reputation score. Suppose that v conducts a deletion behaviour for one of its processed products *id*. After the deletion, v can generate non-ownership proof for id. According to claim 1, v cannot generate ownership proof for id. In the subsequent query phase, if id is queried, there are two cases: (1) id is good, as v cannot show ownership proof for id, it cannot be identified by the proxy and loss positive reputation score; (2) id is bad, as v_i can show non-ownership proof for id, it cannot be identified by the proxy and avoid to be given negative reputation score.

Addition: If v conducts an addition behaviour for any of its processed products, it will face a double-edged situation: getting an opportunity of giving positive reputation score but suffering a risk of giving negative reputation score. Suppose that v conducts an addition behaviour for one of its processed products *id*. After the addition, v can generate ownership proof for id. According to claim 1, v cannot generate non-ownership proof for id. In the subsequent query phase, if id is queried, there are two cases: (1) id is good, as v can show ownership proof for id, it can be identified by the proxy and earn positive reputation score; (2) id is bad, as v cannot show non-ownership proof for id, it can be identified by the proxy and given negative reputation score.

Modification: In a distribution task, a involved participant commits its RFID-traces as a commitment of ZK-EDB. A ZK-EDB commitment is revealed to the verifier as a public key in the interaction between the verifier and the prover, which does not reveal non-trivial information of the committed messages. In a later product path information query, the interaction between the proxy and each identified participant is actually an interaction between a verifier and a prover in ZK-EDB. The product *id* is *x* and the returned RFID-trace of *id* is *D*(*x*). As a result, the proxy cannot learn non-trivial information of non-queried products in the query.

B. Security guarantees in query phase

Given the correct POC list, DE-Sword relies on the soundness property of ZK-EDB to detect the four types of dishonest behaviours in product path information query.

Claim non-processing: If *v* processed *id*, it would commit an RFID-trace indexed by *id* into its POC, and thus can generate an ownership proof for *id*. According to claim 1, *v* cannot generate a non-ownership proof for *id*. As a result, the proxy can ask *v* to return a valid non-ownership proof for verification if *v* claims that it did not process *id*.

Claim processing: If *v* did not process *id*, it would not commit an RFID-trace indexed by *id* into its POC, and can generate a non-ownership proof for *id*. According to claim 1, *v* cannot generate a ownership proof for *id*. As a result, the proxy can ask *v* to return a valid ownership proof for verification if *v* claims that it processed *id*.

Return wrong RFID-trace: If *v* did process *id*, it would commit an RFID-trace indexed by *id* into its POC, and can generate an ownership proof for *id* from which *v* can recover that RFID-trace. According to claim 2, *v* cannot generate another ownership proof for *id* from which *v* can recover a different RFID-trace. As a result, the proxy can ask *v* to return a valid ownership proof and recover the valid RFID-trace from it.

Return the identity of a wrong participant: In the first case, the proxy can identify if the next queried participant processed *id*. As the participant did not process *id*, it would not commit an RFID-trace indexed by *id* into its POC. As a result, the proxy can ask the participant to return a valid non-ownership proof for verification if it claims that it did not process *id* in bad product case or a valid ownership proof for verification if it claims that it processed *id* in good product case. In the second case, the proxy can check if the next queried participant is one of the children of *v* in the POC list. As the POC list reflects the production relationships of all the involved participants in the distribution task, the proxy can easily do so.

VI. Evaluation and analysis

In this section, we evaluate the performance of DE-Sword. At RFID-tag side, DE-Sword only requires RFID-tags to carry short product identifiers and support basic read operation with RFID-reader. At backend server side, DE-Sword requires participants to run POC scheme to compute

and send POCs and ownership/non-ownership proofs. As a result, the overhead of DE-Sword is dominated by the POC scheme at the backend server side.

Very informally, the POC scheme involves the construction of a tree structure, with leafs labeled by RFID-traces and commitments of trapdoor mercurial commitment(TMC) scheme [21], [22], and internal nodes labeled by trapdoor q-mercurial commitment(qTMC) scheme [11]. The root of the tree is a *POC* and the tree as a whole is a *DPOC*. To generate a ownership/non-ownership proof for a product *id*, a participant simply opens all the commitments in the path from the root to the leaf containing *id*.

As a result, we evaluate the performance of DE-Sword from two aspects: (1) micro-benchmarks to evaluate the performance of the TMC scheme and qTMC scheme; (2) macro-benchmarks to evaluate the performance of the POC scheme. We implement the POC scheme based on a Java version of pairing Based Cryptography (PBC) library [19], [20]; and run all the experiments on a 3.10 GHz Intel(R) Core(TM) machine, with 4 GB of RAM. All the experiment results are smoothed by 50 times.

A. Micro-benchmarks

TMC scheme: A TMC scheme allows two decommitment procedures: an hard procedure and a soft one. At committing time, the sender can decide as whether to create an hard commitment or a soft one for a message. Hard commitments look like standard ones, in the sense that they can be (hard or soft) opened only with respect to the committed message. Soft commitments, on the other hand, cannot be hard opened, but can be soft opened to any arbitrary message. We evaluate the running time of seven algorithms of the TMC scheme involving hard commitment and soft commitment, and find that the overhead of all the algorithms are lightweight. Even the most expensive algorithm, Hcom, can be completed in 34 ms in average. We thus conclude that the TMC scheme does not dominate the overhead of the POC scheme.

qTMC scheme: A qTMC scheme extends the TMC scheme to commit a sequence of q messages at once. qTMC scheme also allows for two decommitting procedures. At committing time, the sender can decide to produce a commitment in two ways (hard or soft). A hard commitment, like a standard one, commits a sequence of q messages and can be opened (hard or soft) with respect to the message m_i as well as its position in the sequence. Soft commitments, on the other hand, cannot be hard opened, but can be soft opened to messages belonging to any arbitrary sequence of q messages.

We evaluate the running time of seven algorithms of the qTMC scheme involving hard commitment and soft commitment, and list the results in Figure 4. We notice that the hard opening and soft opening of a hard commitment costs same time. As a result, we report the time of hard opening (qHOpen) in Figure 4(a), and report the time of soft opening (qSOpen) of a soft commitment in Figure 4(b). The results show that the overhead of the key generation (qKGen) and the three algorithms involving hard commitment, namely hard commitment generation (qHCom), hard opening (qHOpen) and soft opening of hard commitment (qSOpen), increase linearly with q, while the overhead of all the algorithms involving soft commitment are constant. We thus conclude that processing hard commitment costs far more time than processing soft commitment.

Comparing with the TMC scheme, the cost of the qTMC scheme is much higher. When the sequence q is 128, the cost of hard commitment generation and opening can be as high as 1.3 seconds in average. Thus, we conclude that the cost of the qTMC scheme dominates the cost of the POC scheme.

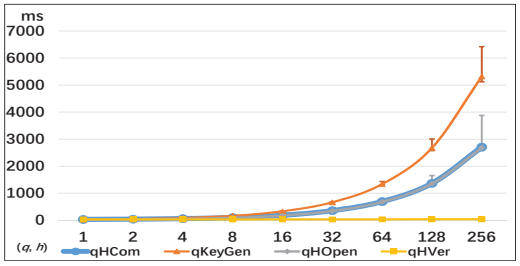
B. Macro-benchmarks

With the TMC scheme and qTMC scheme, the construction of a POC goes as follows. For a set D_{v_i} of RFID-traces, each *id* is assigned to a leaf of a q-ary tree of height h, so that q^h bounds the domain 2^k of *id*. For each *id* with $t^{id}_{v_i} \in D_{v_i}$, the corresponding leaf contains a standard hard mercurial commitment committing $t^{id}_{v_i}$. Each internal node contains a hard q-commitment to messages obtained by hashing its children. The hard q-commitment at the root then serves as a POC to D_{v_i}. To generate an ownership/non-ownership proof for a product id, the participant v_i reveals hard openings/soft openings for all commitments in nodes on the path connecting leaf id to the root. To verify an ownership/non-ownership proof for a product id, the proxy checks if all commitments in nodes on the path connecting id to the root are valid. In the following, we evaluate the overhead of POC scheme with varied breaching factor q and tree height h with the invariant $q^h = 2^{128}$, which is large enough to accommodate a large-scale supply chain.

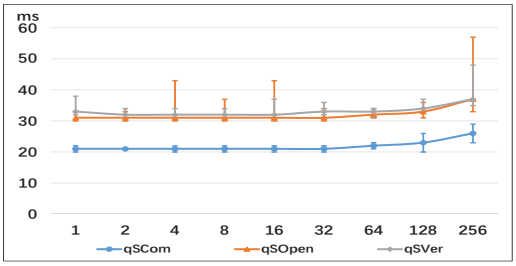
Communication overhead of ownership/non-ownership proof: The communication overhead of ownership/non-ownership proof increases as h increases, and is independent of q. We evaluate the communication overhead of ownership/non-ownership proof with varied (q, h). The results are shown in Table II, which validate

Table II: Communication overhead of the POC scheme

Breaching factor q	Tree height h	Own proof	N-Own proof
8	43	8.94KB	8.08KB
16	32	6.66KB	6.01KB
32	26	5.42KB	4.89KB
64	22	4.59KB	4.14KB
128	19	3.97KB	3.58KB



(a) Running time of processing a hard commitment



(b) Running time of processing a soft commitment

Figure 4: Running time of the qTMC scheme with a sequence of q messages.

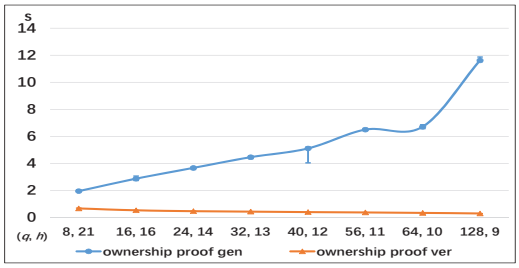


Figure 5: Computation overhead of ownership proof.

our analysis and show that the communication overhead is reasonable even with large h.

Computation overhead of ownership/non-ownership proof: The computation overhead of ownership proof/non-ownership proof generation increases as both q and h increases, while that of ownership proof/non-ownership proof verification increases only as h increases. We evaluate ownership proof generation and verification with varied (q, h) and ignore the non-ownership proof counterpart as the computation overhead of the two are similar. The result is shown in Figure 5. Our evaluation validates our analysis and shows that the computation overhead of ownership proof generation is far larger than that of ownership proof verification.

VII. Related Works

The increasing importance of RFID-enabled product traceability for supply chain management have been identified by the research community of management and computer science [6]–[10], [23]. Aiello et al. [6] evaluate the expected value of the implementation of traceability systems. Piramuthua et al. [7] investigate the effect of selecting a traceability level and technology for contaminated product recall. Chongwatpola et al. [8] propose a traceability approach to improve production scheduling. Zhou et al. [9] model item-level information visibility in a general way. Unnevehr et al. [10] highlight the growing importance of farm to table management of food safety. Different with these works, we focus on RFID-enabled product path information query under the dishonest supply chain participants model.

Several works have concerned security and privacy issues in supply chain [13]–[18]. Sadeghi et al. [13] introduce Internet of Things (IoT), which is an emerging key technology for the next generation of industrial production systems, and identify the related security and privacy challenges. Juels et al. [14] describe two secret sharing schemes as a tool of practical promise for privacy protection in RFID-enabled supply chains. Blass et al. [15] study the problem of private path verification in RFID-enabled supply chains. Zanetti et al. [16] focus on detecting cloned RFID tags in supply chains. However, these works do not consider privacy-preserving product path information query with verifiability on RFID-enabled supply chain.

VIII. Conclusion

Conducting privacy-preserving product path information query with verifiability on an RFID-enabled, distributed supply chain is challenging as traditional cryptographic schemes only work in a honest-data owner model. To solve this problem, this paper presented DE-Sword, an incentivized verifiable query system. DE-Sword introduces a novel double-edged reputation award strategy to encourage the participants to behave in the POC list construction in the distribution phase. With correct POC list, DE-Sword further enables product path information query with verifiability and privacy in the query phase.

IX. Acknowledgement

We acknowledge the support from National Natural Science Foundation of China (No 61602363, No 61572382), China Postdoctoral Science Foundation (No 2016M590927), Hong Kong ECS under Grant PolyU 252053/15E, the Hong Kong PolyU under Grant G-YBMT, Doctoral Fund of Ministry of Education of China (No. 20130203110004), China 111 Project (No. B16037), and the Fundamental Research Funds for the Central Universities (No XJS16001, No JB161509).

REFERENCES

[1] T. Datz, "Drug Busters", CSO Magazine, 2005.

[2] "California Business and Professions Code", Sections 4163, http://www.leginfo.ca.gov/cgi-bin/calawquery?codesection=bpc&codebody=4163&hits=20

[3] "Florida Statutes", Section 499.0121, http://election.dos.state.fl.us/laws/04laws/ch_2004-328.pdf

[4] "Combating Counterfeit Drugs: A Report of the Food and Drug Administration Annual Update", http://www.fda.gov/bbs/topics/NEWS/2005/NEW01179.html

[5] "Q&A: Horsemeat scandal", BBC NEWS, http://www.bbc.com/news/uk-21335872

[6] G. Aiello, M. Enea, and Cinzia Muriana, "The expected value of the traceability information", European Journal of Operational Research, Vol 244, Issue 1, 2015, Pages 176C186.

[7] S. Piramuthua, P. Farahanib, and M. Grunow, "RFID-generated traceability for contaminated product recall in perishable food supply networks", European Journal of Operational Research, Vol 225, Issue 2, 2013, Pages 253-262.

[8] J. Chongwatpola, and R. Sharda, "RFID-enabled track and traceability in job-shop scheduling environment", European Journal of Operational Research, Vol 227, Issue 3, 2013, Pages 453-463.

[9] W. Zhou, "RFID and item-level information visibility", European Journal of Operational Research, Vol 198, Issue 1, 2009, Pages 252-258.

[10] L. J. Unnevehr, "Food safety issues for fresh food product exports from LDCs", Agricultural Economics, Vol 23, Issue 3, 2000, Pages 231-240.

[11] B. Libert, and M. Yung, "Concise Mercurial Vector Commitments and Independent Zero-Knowledge Sets with Short Proofs", in Proceedings of TCC 2010.

[12] D. Catalano, D. Fiore, M. Messina, "Zero-Knowledge Sets with Short Proofs", in Proceedings of EUROCRYPT, 2008.

[13] A-R Sadeghi, C. Wachsmann, and M. Waidner, "Security and Privacy Challenges in Industrial Internet of Things", in Proceedings of DAC, 2015.

[14] A. Juels, R. Pappu, and B. Parno, "Unidirectional key distribution across time and space with applications to rfid security", in Proceedings of USENIX Security, 2008.

[15] E. Blass, K. Elkhiyaoui, and R. Molva, "Tracker: Security and privacy for rfid-based supply chains", in Proceedings of NDSS, 2011.

[16] D. Zanetti, S. Capkun, and A. Juels, "Tailing RFID Tags for Clone Detection", in Proceedings of NDSS, 2013.

[17] S. Qi, Y. Zheng, M. Li, L. Lu, and Y. Liu, "COLLECTOR: A Secure RFID-Enabled Batch Recall Protocol", in Proceedings of IEEE INFOCOM, 2014.

[18] S. Qi, Y. Zheng, M. Li, Y. Liu, and J. Qiu, "Scalable Industry Data Access Control in RFID-Enabled Supply Chain", in ACM/IEEE ToN, Vol PP, Issue 99, DOI: 10.1109/TNET.2016.2536626, 2016, Pages 1-14.

[19] "jPBC: Java Pairing Based Cryptography", http://gas.dia.unisa.it/projects/jpbc.

[20] A. De Caro, and V. Iovino, "jPBC: Java pairing based cryptography", in Proceedings of IEEE ISCC, 2011.

[21] M. Chase, A. Healy, A. Lysyanskaya, T. Malkin and L. Reyzin, "Mercurial commitments with applications to zero-knowledge sets", in Proceedings of EUROCRYPT, 2005.

[22] S. Micali, M. Rabin and J.K. Kilian, "Zero-Knowledge Sets", in Proceedings of IEEE FOCS, 2003.

[23] J. Liu, B. Xiao, K. Bu and L. Chen, "Efficient Distributed Query Processing in Large RFID-enabled Supply Chains", in Proceedings of IEEE INFOCOM, 2014.