Secure Alert Tracking In Supply Chain

Mehdi Khalfaoui[1], Refik Molva[2] and Laurent Gomez[1]
[1]*SAP Labs France, Mougins 06254, France*
[2]*Eurecom , Biot 06410, France*
{*mehdi.khalfaoui, laurent.gomez*}*@sap.com, refik.molva@eurecom.fr*

Keywords: Supply chain, Wireless sensor networks, Privacy preserving, Homomorphic encryption, Alerts detection

Abstract: Risk management practices, techniques and tools with respect to companies' supply chains have begun to receive more attention recently, as the need to improve supply chain performances has increased in order to keep the balance between financial considerations and those of the customer interests. With the multiplication of intermediate actors, a single threat at one point might compromise the safety of the all actors involved in the supply chain process. Therefore, there is a clear need for product tracking in order to trace anomalies for mitigation of potential threats in the future. Traditional approaches rely on operator-assisted verification procedures that mainly suffer from the lack of global coverage. In this paper, we propose an automated process to securely trace the supply chain actors that interact with the product, as well as the operations that were performed, and the alerts that got raised. The core component of this process is wireless sensor nodes attached to the product. Empowered with sensing capabilities, wireless sensor nodes are meant to raise alert in case of detection of an anomaly. Our solution allows for tracing the path taken by a product and the recording of the alerts that got raised, while preserving the actors' privacy. The solution combines a polynomial path encoding technique, together with additive homomorphic encryption to ensure the correctness of the path taken by a product, and to preserve the privacy of the actors, respectively.

1 Introduction

Supply chain management (SCM) involves multiple actors and processes. Those supply chain actors, often, have different roles and responsibilities. Identification of each actor's role and responsibility at supply chain setup is mandatory requirement for making the different actors fully accountable about their actions. Accountability is defined as the requirement or duty to provide an account or justification for one's actions to whomever is answerable (Swift, 2001). In other words, each one of the actors involved in the activity is accountable, explains or justifies actions to the authorities, whenever they demand.

Accountability calls for tracing the different anomalies and accidents in the supply chain, and to link them to the actors who is interacting with the product at the moment when the anomaly occurs. An anomaly tracking mechanism can help enhancing accountability in supply chains. However, supply chain is multi-actors environment, which raises security and privacy challenges. Actors aim at tracking the anomalies in the supply chain, which implies the tracking of the involved actors, and the operations performed on the product, yet they are reluctant to leak information about their internal processes. Hence, preserving the privacy of supply chain actors in this multi-partner environment has to be taken into account.

In that context, product safety and supply chain actors are a mandatory requirement in today's supply chains. However, managing all the supply chain processes, to have a trace of all the anomalies all along the supply chain, can be difficult task thanks to the complexity of the actors' networks.

The purpose of this paper is to introduce an anomaly tracking solution using wireless sensor networks. This mechanism allows the traceability of the operations performed on the product, the actors that interact with the latter, and the anomalies that were detected by sensors. At each anomaly detection, sensor raises an alert. Moreover, the mechanism preserves the privacy of the involved actors, by allowing to only authorized entities the ability of tracing. To this effect, sensors are used as secure storage device to store a mark that represents the product path. More precisely, each product type will be attached to a specific sensor. At each new interaction with an actor, the sensor's memory is updated to add the current ac-

tivity identity. The set of identities collected by the sensor identifies the path trace that the product went through. Moreover, if the sensor detects an anomaly such as raising temperature, it will add it directly to the path. At the end of the supply chain, a verifier extracts the trace and verifies its validity, and what the exact alerts that got generated. The verifier is often a supply chain manager that wants to have a global overview of its supply chain in order to mitigate any potential threats.

In order to preserve actors' privacy, actors' identities have to be kept secret. Encryption is a straightforward solution preventing eavesdroppers from stealing identities and impersonating any legitimate supply chain actor. However, any technical solution that addresses secure and privacy preserving product tracking should take into account the limitations of sensor nodes. These are constrained devices in terms of computation power, memory, and energy. Due to the limited memory featured by sensors, straightforward storage of the collected encrypted steps' identities cannot be afforded. Thus, a compression mechanism to reduce the size of the path traces is a mandatory requirement in order to overcome the memory limitation of sensors. Also, the scarce energy resources of sensor nodes, make the implementation of complex functions such as public key encryption algorithms difficult (Gennaro et al., 1997). Therefore, our solution places all the complex computations on the supply chain actor's system in order to suit the scarcity of the computational resources.

The main features of the suggested tracking scheme are as follows:

- It allows the supply chain manager to verify the legitimacy of the path taken by a product. More precisely, it allows the supply chain manager to verify which set of actors, a product has visited.
- It tracks all the anomalies that got raised by the sensor nodes along the supply chain. For each anomaly, our solution links it to the corresponding supply chain actor.
- It guarantees the privacy of products and therewith actors in the supply chain. Only the supply chain manager is able to verify the path taken by a product.

Moreover the scheme is suitable for low capacity sensors. It only requires a few Kbytes storage. The protocol execution for each supply chain step requires only two modular multiplication.

1.1 Scenario: Importation of Chemicals from China to Europe

Figure 1 illustrates typical supply chain example from RESCUEIT project (Gomez et al., 2011). Aerosols are imported from a Chinese harbour toward the harbour of Le Havre, in France. Shipped goods are household and gardening chemicals: Garlon brush cluttering 350 ml, DECAP four express 500 ml and Ronstar 2kg. These products are meant to be shipped by boat from a Chinese harbour. When received at the Le Havre harbour, the merchandise is checked by customs against REACH (Williams et al., 2009)regulations. REACH is the European Community Regulation on chemicals and their safe use (EC 1907/2006) (Williams et al., 2009). It deals with the registration, evaluation, authorization and restriction of chemical substances. The aim of REACH is to provide an additional layer of protection for humans and the environment through the better and earlier identification of the intrinsic properties of chemical substances. To that extend, REACH introduces specific constraints on chemicals along the supply chain. They include the flash point, incompatibilities between products, and humidity conditions for chemicals. At the Le Havre harbour, french customs with the support of a Approved Economic Operator proceed to a merchandise integrity check. After a check of administrative document describing the content of the cargo, customs verify the quantity and quality of the goods received. Once quality checks have been performed at Le Havre harbour, and customs have verified that the merchandise is compliant with safety regulations, products are shipped by pickup trucks toward the warehouse located close to Savigny le Temple. This warehouse belonging to K+N (KNa, 2013) is classified SEVESO II. This classification defines a set of safety management systems, emergency and land-use planning. In addition, it reinforces provisions on inspections to be carried out by classified sites. In this case specific safety measures are implemented on site, such as storage rules (e.g. limited quantity of chemical stored at the same place). Finally, household and gardening products are distributed by retailers (e.g. supermarket) such as the Casino group (Cas,).

In case of anomalies along the supply chain such as temperature raising, the impact on population safety, and on the environment can be disastrous. For instance, an explosion might take place at the shipment process, which depending on its intensity, fire can have more or less serious impact on individual health (e.g., slightly burning to death). Therefore, anomalies detection mechanism should be in place to

mitigate potential accidents.

Figure 1: Import of Chemical Substance from outside of EU

2 Related work

Anomalies detection in supply chains has been around as an active topic in research for many years. However, managing those anomalies and transiting it to the right people has not been so deeply investigated by research institutions. Often, supply chain actors don't disclose the detection of an anomaly, or they simply ignore it for economical reasons (Trejo-Pech et al., 2009). Therefore, any mitigation plan by the supply chain manager is not complete since no communication about potential threats was made. Wireless sensor networks with their sensing capabilities allow the detection of anomalies related to the physical environment in real time. Accordingly, they are good candidate to raise alerts if an anomaly occurs.

The usage of wireless sensor networks in supply chains started with the need of real time product monitoring. Many projects (Szewczyk et al., 2004; Biagioni and Bridges, 2002; Werner-Allen et al., 2005; Burrell et al., 2004; Gibbons et al., 2003) use wireless sensor networks capabilities of sensing the physical parameters of the environment and detecting at early stage any potential threat. Nevertheless, those project often neglect the multi-actors aspect of today's supply chains, and the need of supply chain actors to protect their sensitive data, such as storage time, cost, their clients, or even their suppliers. For instance Gomez et al. (Gomez et al., 2012) suggested a mechanism to delegate anomalies detection to sensor nodes. The mechanisms is based on bytecode representation of alerts to deal with the limited memory issues. However, the solution doesn't take into account the privacy of the involved supply chain, neither the threat of fake alerts injection.

Cryptographic mechanisms preserving privacy of supply chain actors are suggested by many researchers. Khalfaoui et al. (Khalfaoui et al., 2012) proposed a mechanism based on AND-ACC collision and paillier cryptosystem in order to trace the steps and the activities that a product went through, while preserving the privacy of the actors that interacted with the product. Only the supply chain manager, which is the verifier can trace the product path. Elkhiyaoui et al. (Blass et al., 2011) proposed an cryptographic mechanism on RFIDs to deal with similar tracing problem. Nevertheless, there is no approach that comprehensively addresses the problem of anomaly detection in supply chains using sensors and in a privacy-preserving manner.

3 Background

A supply chain in this paper simply denotes a set of sites that a product goes through. These sites perform a specific activity involving the product such as packaging, shipment, or storage. The main concern taken into account by our solution is the privacy of the partners in the supply chain against the other business partners. Only, an authorized actor is able to recognize the sites that a product visited, and the activity that were performed. Formally, a supply chain is represented by a digraph $G = (V, E)$ whereby each vertex v represents one step in the supply chain. A step v in the supply chain is uniquely associated with an entity. Each directed edge e, which links vertex v_i to vertex v_j, express that v_j is a possible next step to step v_i in the supply chain. This simply means that according to the organization of the supply chain, a product might proceed to step v_j after the completion of step v_i. Whenever a product in the supply chain proceeds to step v_i , the entity interacts first with the sensor. This latter may detect a potential threat. Whenever the case, S added an alert $alert_k$ the path trace. A path P is defined as finite sequence of steps $P = (v_0, alert_0), ..., (v_l, alert_l)$, where l is the length of the path P. A Path P is deemed valid if it is part of a legitimate supply chain networks. The proposed solution involves the following entities:

- **Sensors** S_i: Each sensor is attached to a product in the supply chain. A sensor S_i features a re-writeable memory that stores the trace $T_{(S_i,j)}$. $T_{(S_i,j)}$ represents the trace of the path that the sensor took in the supply chain until site j. Sensors can also compute a cryptographic function f to

authenticate the partners' systems in the supply chain.

- **Issuer** I: the issuer I attaches S_i to a product and writes into S_i an initial trace $T_{(S_i,0)}$.
- **Manager** M: M wants to identify the sequence of steps that S went through and the sequence of alerts that S raised it all along the product path. M therefore, reads the current trace $T_{(S,M)}$ of S, and decides whether S visited legitimate sequence of steps or not. Then, M extracts the different alerts. We assume that M knows which paths in the supply chain are valid or not. In other words, M has a database DB_{valid} of valid path traces.
- **Actors** Ac_i: supply chain actor is partner that belong to the supply chain, and may or may not interact with the product. Each actor is allowed to perform a set of supply chain activities on the product. Without loss of generality, we assume that each supply chain actor's system Ac_i. Ac_i uses some function f_{Ac_i} to generate $T_{(S,step_{j+1})}$ from $T_{(S,step_j)}$, i.e., $f_{Ac_i}(s_{(S_i,j)}) = T_{(S,j+1)}$.
- **Alert** $alert_i$: an $alert_i$ is raised when an anomaly is detected by S, such as increasing in temperature or pressure above a defined threshold.
- **Step** $step_i$: supply chain step $step_i$ step is an activity that has been performed by a supply chain actor. $step_i$ represents the i^{th} activity performed on the product.

Thus, an AlertSec system is:

- a Supply Chain $G = (V,A)$
- a Sensor S
- a set of possible traces $\mathcal{T}$
- a set of different steps $Steps$
- a set of different alerts $Alerts$
- a supply chain manager M
- a set of valid paths $\mathcal{P}$

4 Protocol description

4.1 Approach

In our scheme, the path trace combines the sequence of the supply chain steps visited by S, and the alerts that are generated during the product transition. The main concept is to represent different paths in the supply chain using different polynomials. More precisely, at the end of a supply chain's valid path P_{valid}, a path's trace is the evaluation of specific polynomial at a fixed value x_0. Therefore, a path in the supply chain is represented by polynomial $P(x_0)$ that consists of digest of the path.

However, the polynomial based path representation is vulnerable to prevent path cloning, i.e., copying the path of a valid sensor into a fake one to be injected into the supply chain. To tackle this problem, sensors will store $P(x_0)$ multiplied by a keyed-HMAC of their unique IDs. Keyed-HMAC serves the following purpose: It proves that sensors are issued by a legitimate authority and prevents an adversary from injecting its own sensors. A path's trace, therefore, consists of three elements that are: a unique ID, $HMAC_k(ID)$ and $HMAC_k(\mathrm{ID})*P(x_0)$. Our scheme can be structured into four parts:

- Issuer I writes an initial trace into a sensor S.
- Supply chain actors successively compute the evaluation of a polynomial to achieve the evaluation of the final polynomial $P(x_0)$ at the end of a valid path. Each actor visited by sensor updates a new trace by applying specific arithmetic operations.
- sensor S raises alert and add it to the path trace.
- M extracts the trace from S, and checks if it is valid one or not.

4.2 Preliminaries

4.2.1 Path Encoding technique

The polynomial path encoding is used in (Blass et al., 2011). It is based on techniques for software fault detection. Noubir et al. (Noubir et al., 1998) propose to encode a software's state machine using polynomials such that the exact sequence of states visited during run-time generates a unique *"mark"*. Therewith, run-time faults can be detected. By considering the step instead of state machine, the path encoding used by Noubir et al. (Noubir et al., 1998) can be applied in our case.

For each step $step_i$ in the supply chain, $step_i$ is associated with a unique random identifier $step_i \in \mathbb{F}_q$, where q is a large prime.

As mentioned above, a path in the supply chain is represented as a polynomial $\in \mathbb{F}_q$. The polynomial corresponding to a path $\mathcal{P} = \overrightarrow{(step_0, alert_0)(step_1, alert_1)\ldots(step_l, alert_l)}$ is defined in Equation (1). All operations are in $\mathbb{F}_q$.

$$Q_{\mathcal{P}}(x) = \sum_{i=0}^{l} c_i * x^{l-i} \tag{1}$$

where c_i corresponds to the identity of either a $step_j$ or $alert_l$.

To have a more compact representation of paths, a path $\mathcal{P}$ is represented as the evaluation of $Q_{\mathcal{P}}$ at x_0, where x_0 is a generator of $\mathbb{F}_q^*$. We denote $\phi(\mathcal{P}) = Q_{\mathcal{P}}(x_0)$. The desired property of anti-collision, .i.e $\forall \mathcal{P} \neq \mathcal{P}\prime, Pr(\phi(\mathcal{P}) = \phi(\mathcal{P}\prime)) = \frac{1}{q}$ (Noubir et al., 1998), ensures the uniqueness of the path mark with high probability.

4.3 Paillier Cryptosystem

The following is description to the Paillier cryptosystem (Paillier, 1999) that we use in order to achieve both privacy and security of our mechanism:

Key Generation Let k be the security parameter. Choose uniformly and at random two k-bit primes p and q, set $N = pq$, and set $\lambda(N) = lcm(p-1, q-1)$. Choose a random base $g \in \mathbb{Z}_N^*$.

Encryption To encrypt message $m \in \mathbb{Z}_N$, one chooses a random value $r \in \mathcal{Z}_N^*$ and computes the ciphertext as

$$c = \mathcal{E}(m,r) = g^m r^N mod N^2 \tag{2}$$

Decryption When receiving a ciphertext c, check that $c < N^2$. If yes, retrieve the message m as

$$m = \mathcal{D}(c) = \frac{L(c^{\lambda(N)} mod N^2)}{L(g^{\lambda(N)} mod N^2)} mod N \tag{3}$$

$$\text{Where } \forall u \in \{u < N^2 / u \equiv 1 mod N\} L(u) = \frac{u-1}{N}$$

Additive Homomorphic property Paillier cryptosystem has the property to be additively homomorphic:

$$\mathcal{E}(m_1, r_1) * \mathcal{E}(m_2, r_2) = \mathcal{E}(m_1 + m_2, r_1 r_2) \tag{4}$$

This property allows the execution of arithmetic operations on encrypted data. Therefore, it supports the evaluation of the polynomial mark at each step of the supply chain without decryption.

Self Blinding Paillier cryptosystem has the property to be *Self-Blinding*, i.e the property by which any ciphertext can randomly be changed into another without affecting the plaintext. This property is achieved as follows:

$$\forall r \in \mathbb{Z}_N \mathcal{D}(\mathcal{E}(m,r)) = m \tag{5}$$

Therefore, the decryption of any message m is independent of the value of r.

4.4 Detailed Protocol description

Our protocol consists of several phases, which we describes as follows:

- Initialization: This phase is responsible for initializing the supply chain actors and the wireless sensor nodes.
- Authentication: This phase is responsible for verifying the legitimacy of the actors that interact with sensors.
- Collection: This phase is responsible for collecting the actors' identities, and alerts to have a unique path trace.
- Verification: This phase is responsible for verifying the correctness of the path trace and the extraction of the different alerts.

4.4.1 Initialization

In this phase, we assume that every actor in the supply chain has the required resources to perform the following actions:

- I generates randomly a list of steps identifiers for each actor. I distributes each step's identity to the corresponding actor.
- I generates a Paillier public key pk_M and private key sk_M. Then I sends $\mathcal{E}_M(step_i)$ to its actor in secure way.
- M shares with the supply chain actors a Rabin's private key (p_R, q_R). Rabin cryptosystem (Rabin, 1979) is used to achieve authentication of the supply chain actors. Rabin encryption is single square modular encryption, which makes it feasible for low capacity devices such as sensor nodes. Rabin's public key $N_R = p_R q_R$ is stored in the sensor to perform the authentication process.
- M generates the identifiers list S_{list} of sensors. For the sake of simplicity, we assimilate a sensor S_i and its identifier ID_i. I stores in each sensor S_i, the Paillier encryption of its ID_i, the Paillier encryption of $HMAC_k(S_i)$, where $HMAC_k$ is keyed hash function (Bellare et al., 1996), and k is its secret key.
- M generates the identifiers list $alert_{list}$ of alerts. M stores in each sensor S_i, the Paillier encryption of the all alerts needed, and the Paillier encryption of $HMAC_k(S_i)$, where $HMAC_k$ is keyed hash function (Bellare et al., 1996), and k is its secret key.

At the completion of this phase, each sensor is initialized with the value of the initial trace, that is, $E(HMAC_k(S_i))$.

4.4.2 Trace update

In this phase, the sensor S and the actor's system interact in the supply chain by executing the following actions: We assume that the sensor S has visited the steps $step_0$, ..., $step_l$. When, S visits the actor's system $step_{l+1}$, it is already stored the trace $\mathcal{P}_l = \overrightarrow{(step_1, alert_1)(step_2, alert_2)...(step_l, alert_3)}$ that encodes the path from the sites that belong to the steps $step_1, step_2, \ldots, step_l$, and raised the alerts $alert_1, alert_2, \ldots, alert_3$. Therefore, the current state of the sensor is $E(\mathcal{P}_l)$, which corresponds to the path trace after interacting with the l steps.

4.4.3 Authentication

S chooses a random value $r \in \mathbb{F}_{N_R}$ and sends $Rabin(r) = r^2 mod N_R$ to A, while storing the $hash(r)$ and $hash(N_R - r)$. A decrypts $Rabin(r)$ using its public key. The decryption gives exactly four solutions, $r, N_R - r, t, N_R - t$. As the actor does not know which is the real solution, he chooses to send back to S two hash values. The values are chosen in such a way that their sum is not null $mod N_R$. For example A chooses to send $hash(r)$, and $hash(N_R - t)$. Therefore, S considers the authentication as successful, if one of the received value matched one of the stored value. Then, S can start trace collection procedure.

4.4.4 Collection

After the authentication phase, S starts the collection phase. S sends its current state $s(S,l)$ to the actor's system A_{l+1}. For the sake of simplicity, we assume there is no alert has been raised at this point of time. Therefore, $actor_{l+1}$ updates the sensor's state as follows:

$$s(S,l+1) = s(S,l)^{x_0} * \mathcal{E}_M(c_{l+1}) \quad (6)$$

$$= s(S,l)^{x_0} * g^{c_{l+1}} r^N mod N^2 \quad (7)$$

Assuming that our products has to interact with n supply chain partner, the final sensor's state is:

$$\begin{aligned} T(S,n) &= T(S,n-1)^{x_0} * g^{step_n} r_1^N mod N^2 \\ &= T(S,0) * g^{\sum_{i=1}^{n} step_i x_0^{n-i}} r_2^N mod N^2 \\ &= g^{HMAC_k(S) x_0^n + \sum_{i=1}^{n} step_i x_0^{n-i}} r_3^N mod N^2 \\ &= \mathcal{E}_M(HMAC_k(S) {x_0}^n + \sum_{i=1}^{n} step_i {x_0}^{n-i}) \end{aligned} \quad (8)$$

$$where \quad r_1, r_2, \quad and r_3 \quad are \quad in \quad \mathbb{F}_{N_R}. \quad (9)$$

4.4.5 Alert Injection

Now Let's assume that an alert $step_m$ is raised just after the $step_k$. S updates its internal trace to take into account the current alert. As the last step is $step_k$, the current state of S is s(S,k). S adds the alert to the trace as follows:

$$T(S,k,alert_i) = T(S,k)^{x_0} * g^{alert_i * x_0^d} r_1^N \quad (10)$$

$mod N^2, where \quad r_1 \in \mathbb{F}_N$

$$= \mathcal{E}_M(HMAC_k(S) {x_0}^n + \quad (11)$$

$$\sum_{j=1}^{k} step_j * {x_0}^{n-j} + alert_i * x_0^d) \quad (12)$$

4.4.6 Path verification

In this phase, the supply chain manager M checks if the path recorded in the sensor is a valid one. M extracts the final trace from the sensor $T(S,n)$, and decrypts it, so he can extract the path trace $\phi(\mathcal{P})$.

$$\phi(\mathcal{P}) = \mathcal{D}_{TTP}(T(S,n)) = \sum_{i=1}^{n} a_i x^i$$

$$= HMAC(S) x_0^m + \sum_{i=1}^{m} step_i {x_0}^{m-i} + \quad (13)$$

$$\sum_{i=d+p}^{n} alert_{i-d} {x_0}^{d+p-i} \quad (14)$$

Using successive division operations, M extracts the coefficients $a_0, a_1, \ldots, a_n$ of the polynomial $\phi(\mathcal{P})$. Then, M computes $HMAC_k(S)$ and compare it with a_m. If $a_m = HMAC_k(S)$, M accepts the sensor. Otherwise, M rejects S.

4.4.7 Alert detection

Finally, M checks if the step identifiers $step_1$, $step_2$, ..., $step_m$ belongs to the list of valid steps' identifiers, and the path trace $\phi(\mathcal{P}) mod x_0^d$ proofs that the sequence of the steps is valid. if one of the identifiers, or the sequence is not valid, M rejects the sensor, and declares the product as not compliant. M retrieves the alerts $alert_0$, $alert_1$, ..., $alert_l$ as follows:

$$alert_i = a_{d+p-i} \quad (15)$$

$alert_i$ is detected at step $(m+d) - (d+p-i)$ which is $step_{m-p+i}$.

5 Security Analysis

In this section we prove the security of AlertSec system. The proof was inspired from (Ouafi and Vaudenay, 2009). Let's $\mathcal{A}$ an adversary that has the purpose of forging a valid path.

5.1 $\mathcal{A}$ is not a legitimate actor

if $\mathcal{A}$ authenticates successfully itself to S, she breaks the Rabin scheme security by definition. Therefore, only the legitimate supply actors can try to update the path trace maliciously.

if $\mathcal{A}$ took a blank sensor and try to inject a valid trace on it (from scratch), we use the security of keyed-HMAC to proof the infeasibility of this case.

Security of keyed-HMAC For our proof sketch, we are using the property indistinguishability of keyed hash function.

Indistinguishability property Let $O_{distinguish}$ be an oracle that when $\mathcal{A}$ provides it with a message m, $O^{distinguish}$ returns with the same probability a random number, or $HMAC_k(m)$. $\mathcal{A}$ cannot guess with no-negligible probability if the returned value is a random number, or $HMAC_k(m)$.

Lemma *Producing a new valid trace contradicts the indistinguishiability property of $HMAC_k$.*
Proof (Sketch). From $\mathcal{A}$, we can build an adversary $\mathcal{A}'$ that uses $\mathcal{A}$ to break the indistinguishability property of keyed-HMAC. we provide $\mathcal{A}$, with a sensor S and its s_{id}. $\mathcal{A}$ produces a new valid trace $\mathcal{E}(T_m)$ that corresponds to the sensor s_{id}. $\mathcal{A}$ provides s_{id} to $O_{distinguish}$. $O_{distinguish}$ returns value H to be tested. $\mathcal{A}$ decrypts $\mathcal{E}(T_m)$. She gets T_m, and computes $T_m \, mod \, H$. If $T_m \, mod \, H = 0$, H is the $HMAC_k(s_id)$, otherwise H is a random number.

5.2 $\mathcal{A}$ is a legitimate actor

If $\mathcal{A}$ is legitimate actor, the authentication process will succeed, and S accepts the received value from $\mathcal{A}$. In this case, the decisional composite residuosity assumption to prove the security of our protocol against forgery by a legitimate actor.

The decisional composite residuosity assumption (DCRA) The DCRA states that given a composite n and an integer z, it is hard to decide whether z is a n-residue mod n^2 or not. In other words, whether there exists y such that $z = y^n mod \, n^2$. This assumption is mainly used to proof the semantic security of Paillier cryptosystem (Paillier, 1999).
Cryptographic protocol is semantically secure if its indistinguishability against chosen plaintext attacks (IND-CPA) holds.

SPT is semantically secure if and only if DCRA and the indistinguishibilty of keyed-HMAC hold.

Proof. The main idea of this proof is to build an attacker $\mathcal{A}\prime$ from $\mathcal{A}$ whose advantage ε to forge a valid path, that is able to break DCRA. As shown in the previous *Lemma*, $\mathcal{A}$ cannot provide a new valid path trace from scratch. Now, Let's assume that $\mathcal{A}$ can update a valid path trace that she got from the learning phase to a new valid path trace. For the sake of simplicity, we consider that the AlertSec system has only one valid path.

Let O_{DCRA} be an oracle that, when it is queried with a parameter n, it flips a coin $b \in 0,1$. If $b = 1$ it takes a $y \in Z$ and returns $y^n mod \, n^2$ otherwise, it returns a random number C.

$\mathcal{A}\prime$ creates AlertSec system with a valid path, ($step_0$, ..., $step_m$). Then, she generates the AND-ACC identifier corresponds to each step.
Let $\mathcal{E}(T_{m-1})$ be the encrypted path trace until the $step_{m-1}$. First, $\mathcal{A}'$ sends a query to O_{DCRA} with N (Paillier modular) as parameter, and gets a challenge C. $\mathcal{A}'$ computes $\mathcal{E}(T_{m-1})$*C mod N^2, and writes the result in a sensor S.
If in the challenge phase, $\mathcal{A}$ is able to update the trace to a valid path trace $\mathcal{E}(T_m)$, then $\mathcal{E}(T_{m-1})$. C mod N^2 is a valid ciphertext of T_{m-1} (i.e $\mathcal{E}(T_{m-1})$. C mod N^2 is re-encryption to $\mathcal{E}(T_{m-1})$). Therefore, C is N-residue mod N^2, and $\mathcal{A}'$ breaks the DCRA assumption, with advantage of 1/2*ε, since she is wrong half of the time because of oracle's coin flip.
Table 1 illustrates the messages exchanged between the involved entities during the challenge game.

□

6 Privacy analysis

In this section we prove the privacy requirement of step unlinkability of AlertSec system. Step unlinkability is defined as preventing an adversary $\mathcal{A}$ from telling that two different sensors interacted with a common step.

theorem AlertSec provides step unlinkability under DCRA.

Table 1: Forgery challenge game

O_{DCRA}		$\mathcal{A}'$		$\mathcal{A}$
receive N	$\longleftarrow$	send N		
pick C	$\longrightarrow$	receive C		
		compute $\mathcal{E}(T_{m-1}).C \bmod N^2$	$\longrightarrow$	receive $\mathcal{E}(T_{m-1}).C \bmod N^2$
		receive $\mathcal{E}(T_m)$	$\longleftarrow$	update trace $\mathcal{E}(T_m)$
receive 1 or 0	$\longleftarrow$	if $\mathcal{E}(T_m)$ is valid, send 1 else send 0		

Proof. Assume there is an adversary A whose advantage ε to break the step unlinkability experiment is non-negligible. We now construct a new adversary $\mathcal{A}'$ that executes $\mathcal{A}$ and breaks the semantic security of Paillier.

Let O_{DCRA} be an oracle that, when it is queried with a parameter n, it flips a coin $b \in 0,1$. If $b = 1$ it takes a $y \in Z$ and returns $y^n mod\, n^2$ otherwise, it returns a random number C.

$\mathcal{A}\prime$ creates AlertSec system with multiple valid paths. Then, She generates the AND-ACC identifiers correspond to each step. First, $\mathcal{A}'$ sends a query to O_{DCRA} with N (Paillier modular) as parameter, and gets a challenge C. Then, $\mathcal{A}'$ builds two traces for two different path, with one step in common. Let $\mathcal{E}(T_m)$ the path trace for the path ($step_0$, ..., $step_m$), and $\mathcal{E}(T'_m)$ the path trace for the path ($step'_0$, ..., $step'_m$) with $step_i$ and $step'_i$ are the common step. However, the identifier is $\mathcal{E}(v_i)$ for $step_i$ and $\mathcal{E}(v_i)$*C mod N^2 for $step'_i$. $\mathcal{A}'$ provides the two traces to $\mathcal{A}'$ in the challenge phase.

If in the challenge phase, $\mathcal{A}$ is able to decide if both traces has a common step with an advantage ε, then $\mathcal{E}(v_i)$. C mod N^2 is a valid ciphertext of v_i). Therefore, C is N-residue mod N^2, and $\mathcal{A}'$ breaks DCRA, with advantage of 1/2*ε, since She is wrong the half of the time because of oracle's coin flip.
Table 2 illustrates the messages exchanged during the challenge game.

□

7 Performance analysis

This section is allotted to present the analytical performance evaluation of the proposed scheme. We only evaluate the performances related to the sensor itself. The performance evaluation criteria of the sensor, are the storage cost, the computation cost and the communication cost.

7.1 Storage cost

The storage cost is computed as the number of bytes that the sensor node has to store. Generally, this storage cost is introduced by the storage of different parameters and keys necessary to the function of the our scheme. The proposed privacy preserving product tracking scheme does not require much memory overhead.

- Initialization phase: in this phase, the sensor has to store Rabin's public key of size $sizeof(N_R)$, the path trace initialization $sizeof(\mathcal{E}(T))$, a sensor ID of size $sizeof(S_{id})$, and $k4$ encrypted alerts of $sizeof(\mathcal{E}(T))$ each. Therefore, the total storage needed by S at this phase is $(k+1) * sizeof(\mathcal{E}(T))$, + $sizeof(S_{id})$ + $sizeof(N_R)$.
- Collection phase: In this phase, S has to generate random number to start the authentication with the actor's system. The generated nonce has a size of $sizeof(N_R)$. Hash value of the generated nonce of size $sizeof(hash)$ has to be stored as well. Therefore, the the total storage needed by S at this phase is $sizeof(N_R)$ + $sizeof(hash)$. The update path process does not increase the size of the path trace, thus, no more memory capacity is required.
- Verification phase: In this phase, no storage by the sensor is required.

The total storage cost needed by S in our scheme is $(k+1) * sizeof(\mathcal{E}(T))$, + $sizeof(S_{id})$ + $2 * sizeof(N_R)$ + $sizeof(hash)$.

Table 2: step unlinkability challenge game

O_{DCRA}		$\mathcal{A}'$		$\mathcal{A}$
receive N	$\longleftarrow$	send N		
pick C	$\longrightarrow$	receive C		
		compute $\mathcal{E}(T_m)$ and $\mathcal{E}(T'_m)$	$\longrightarrow$	receive $\mathcal{E}(T_m)$ and $\mathcal{E}(T'_m)$
		receive 1 or 0	$\longleftarrow$	check if $\mathcal{E}(T_m)$ and $\mathcal{E}(T'_m)$ have a common step
receive 1 or 0	$\longleftarrow$	if 1 is received, send 1 otherwise send 0		

7.2 Energy cost

7.2.1 Computation cost

The computation cost can be measured in terms of time, use of CPU or energy dissipation. In fact, these parameters are related and each one can be deduced from the other. For instance, the energy dissipation can be deduced from the time as follows: Energy=Power*Time, where Power represents the CPU power when it is in its active state and Time represents the computing time. In the present analysis, the term cost is used in its general form without specifying the unit. The computation cost of our scheme during each phase can be computed as the sum of the computation cost of the main operations executed during this phase.

- Initialization phase: in this phase, the main operations are performed by the supply chain manager M himself. Therefore, No computation required by the sensor S in this phase.
- Collection phase: in this phase, S generates a random number, then S has to encrypt it using rabin scheme (i.e. single modular square). S computes a hash in order to check the validity of the actor's response. Then, S updates its trace by adding alert to it, which has a cost of single modular multiplication. In total, S consumes $c(rabin) + c(rand) + c(hash) + c(modularmultiplication)$ for each interaction with a single (step,alert) at maximum. Indeed, if there is no alert detected, no modular multiplication is required.
- Verification phase: In this phase, S has to authenticate the supply chain manager M, which is similar to authenticate a supply chain actor. therefore, S consumes in this phase, $c(rabin) + c(rand) + c(hash)$.

The total computation cost needed by S in our scheme is $(c(rabin) + crand + c_{hash})*(l+1) + +c(modularmultiplication)*m$, where m is the number of steps in the supply chain.

7.2.2 Communication cost

The main factor of the communication cost is the energy dissipation. The communication cost is computed using the same approach as TKH (Son et al., 2010). Actually, the communication cost in terms of energy dissipation is computed as the size of sent/received messages multiplied by the energy dissipated for the sent/receive of one bit. We denote e_r the energy consumed by S, when it receives one bit, and e_s when S sends one bit.

- Initialization phase: In this phase, S does not send any messages, however, it receives the initialization parameters. therefore, S consumes (sizeof(N_R)+ sizeof(S_{id})+sizeof($\mathcal{E}(T)$))*e_r.
- Collection phase: In this phase, S sends encrypted rabin value, which has the same size as the rabin's public key, then S receives a hashed value. Then, S receives the encrypted step's identity from the actor's site. Therefore, S consumes (sizeof(N_R))*e_s+ (sizeof($hash$) +sizeof($\mathcal{E}(T)$))* e_r for each interaction with a single step.
- Verification phase: In this phase, S exchanged the same messages with the supply chain manager as it performs with a supply chain actor's site, except of the received updated path trace. Therefore, S consumes (sizeof(N_R)+ sizeof($\mathcal{E}(T)$))*e_s+ (sizeof($hash$))* e_r

The total communication cost needed by S in our scheme is (sizeof(N_R)+ sizeof($hash$)* $(l+1)$ + sizeof(S_{id})+sizeof($\mathcal{E}(T)$)*l + 1)*e_r

+(sizeof(N_R)*($l+1$)+ sizeof($\mathcal{E}(T)$))*e_s where l is the number of the supply chain actors that interact with the product.

It is worth to mention that the storage cost, computation cost, and the communication cost can have a different result depending on the size of the security keys, and the algorithms that the supply chain manager may choose.

In this paper, a Rabin's public key has a size of 1024 bits. The hash function used is SHA1, which has an output size of 160 bits. Paillier encryption has an output's size of 2048 bits. For sensor identity, a size of 160 bits is chosen. Rabin's encryption is a single modular square which requires roughly 100μJ using ATmel128 microprocessor (http://www.atmel.com/Images/doc2467.pdf,) based on the result of Gaubatz et al. (Gaubatz et al., 2005). In our scheme, we use the TinyRNG (Francillon and Castelluccia, 2007) to generate random numbers. TiyRNG consumes around 58μJ at each random number generation. Hash function consumes roughly 1μJ (Hempstead et al., 2008). The communication cost are set to e_s =0.209μJ and e_r =0.226μJ from the characteristics of the CC2420 transceiver used in the Xbows MICA-Z and Telos B sensor nodes (http://www.xbow.com/,).

Table 3 shows that our protocol is implementable using today's sensors such as Crossbow motes (http://www.xbow.com/,) and phidgets (http://www.phidgets.com/,). It only requires sensors to store mainly the Rabin public key, which is 1024 bits, and the encrypted state, which is 2048 bits. Through the different steps of the supply chain, the amount of memory needed does not increase.

8 conclusion

In this paper, we presented a protocol to secure the tracking of products and alerts in supply chain. Our main idea is to encode the path of the products using polynomial path encoding. Partners in the supply chain update the path trace successively, such that the path has unique identifier. Whenever a sensor detect a potential threat, it updates its internal trace by adding the specific alert. Our protocol's security and privacy proprieties relies on the semantic security of Paillier and the security of keyed-HMAC. It requires only one modular multiplication in each step, and only 3Kb of storage, which ensures its feasibility in available sensors in the market.

In our supply chain scenario, we assume that we have a global supply chain manager. There is no notion of multiple managers. However in real world, that is might not be true. Supply chain can have a quality, security, and recall manger. Delivering the right information to the right manager is an issue, especially in big scale supply chains. However, this is left to future work

REFERENCES

Bellare, M., Canetti, R., and Krawczyk, H. (1996). Keying hash functions for message authentication. In *Advances in CryptologyCRYPTO96*, pages 1–15. Springer.

Biagioni, E. and Bridges, K. (2002). The application of remote sensor technology to assist the recovery of rare and endangered species. *International Journal of High Performance Computing Applications*, 16(3):315–324.

Blass, E., Elkhiyaoui, K., and Molva, R. (2011). Tracker : security and privacy for rfid-based supply chains. In *NDSS'11, 18th Annual Network and Distributed System Security Symposium, 6-9 February 2011, San Diego, California, USA, ISBN 1-891562-32-0*.

Burrell, J., Brooke, T., and Beckwith, R. (2004). Vineyard computing: Sensor networks in agricultural production. *Pervasive Computing, IEEE*, 3(1):38–45.

Casino (2013). Casino group.

Francillon, A. and Castelluccia, C. (2007). Tinyrng: A cryptographic random number generator for wireless sensors network nodes. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on*, pages 1–7. IEEE.

Gaubatz, G., Kaps, J., and Sunar, B. (2005). Public key cryptography in sensor networksrevisited. *Security in Ad-hoc and Sensor Networks*, pages 2–18.

Gennaro, R., Krawczyk, H., and Rabin, T. (1997). RSA-based undeniable signatures. *Advances in CryptologyCRYPTO'97*, pages 132–149.

Gibbons, P., Karp, B., Ke, Y., Nath, S., and Seshan, S. (2003). Irisnet: An architecture for a worldwide sensor web. *Pervasive Computing, IEEE*, 2(4):22–33.

Gomez, L., Gaci, O., Deutsch, J., and El-Khoury, E. (2012). Sensor based risk assessment for the supply of dangerous products. In *SENSORCOMM 2012, The Sixth International Conference on Sensor Technologies and Applications*, pages 342–348.

Gomez, L., Khalfaoui, M., El-Khoury, E., Ulmer, C., Deutsch, J., Chettouh, O., Gaci, O., Mathieu, H., El-Moustaine, E., Laurent, M., et al. (2011). Rescueit: securisation de la chaine logistique orientee service depuis le monde des objets jusqua lunivers informatique. In *Workshop Interdisciplinaire sur la Securite Globale*.

Hempstead, M., Lyons, M., Brooks, D., and Wei, G. (2008). Survey of hardware systems for wireless sensor networks. *Journal of Low Power Electronics*, pages 11–20.

Table 3: Performance analysis of AlertSec

Parameter	value
sizeof(S_{id})	160 bits
sizeof(N_R)	1024 bits
sizeof($hash$)	160 bits
sizeof($\mathcal{E}(T)$)	2048 bits
$c(rabin)$	$100\mu J$
$c(modularmultiplication)$	$100\mu J$
$c(rand)$	$58\mu J$
$c(hash)$	$1\mu J$
e_r	$0,209\mu J$
e_s	$0.226\mu J$
The storage cost	20714 bits for 8 alerts
the computation cost	$161,59mJ$ for 100 steps and 10 detected alerts
the communication cost	$12,34mJ$ for supply chain with 100 steps

http://www.atmel.com/Images/doc2467.pdf. Last access: 01/06/2012.

http://www.phidgets.com/. Last access: 12/06/2011.

http://www.xbow.com/. Last access: 01/06/2012.

Khalfaoui, M., Molva, R., and Gomez, L. (2012). Secure product tracking in supply chain. In *INSCRYPT 2012, 8th International Conference on Information Security and Cryptology, 28-30 November 2012, Pekin, China*, Pekin, CHINA.

Khune and Nagel (2013). Khune and nagel group.

Noubir, G., Vijayananda, K., and Nussbaumer, H. (1998). Signature-based method for run-time fault detection in communication protocols . *Computer Communications*, pages 405–421.

Ouafi, K. and Vaudenay, S. (2009). Pathchecker: An RFID Application for Tracing Products in Supply-Chains. In *International Conference on RFID Security*. Citeseer.

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology, EUROCRYPT99*, pages 223–238. Springer.

Rabin, M. (1979). Digitalized signatures and public-key functions as intractable as factorization.

Son, J.-H., Lee, J.-S., and Seo, S.-W. (2010). Topological key hierarchy for energy-efficient group key management in wireless sensor networks. *Wirel. Pers. Commun.*, 52(2):359–382.

Swift, T. (2001). Trust, reputation and corporate accountability to stakeholders. *Business Ethics: A European Review*, 10(1):16–26.

Szewczyk, R., Osterweil, E., Polastre, J., Hamilton, M., Mainwaring, A., and Estrin, D. (2004). Habitat monitoring with sensor networks. *Communications of the ACM*, 47(6):34–40.

Trejo-Pech, C. J., Weldon, R. N., House, L. A., and Gunderson, M. A. (2009). The accrual anomaly financial problem in the food supply chain. *Agribusiness*, 25(4):520–533.

Werner-Allen, G., Johnson, J., Ruiz, M., Lees, J., and Welsh, M. (2005). Monitoring volcanic eruptions with a wireless sensor network. In *Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on*, pages 108–120. IEEE.

Williams, E. S., Panko, J., and Paustenbach, D. J. (2009). The european union's reach regulation: a review of its history and requirements. *Critical reviews in toxicology*, 39(7):553–575.