Pathchecker: an RFID Application for Tracing Products in Suply-Chains[*]

Khaled Ouafi[**] and Serge Vaudenay

EPFL
CH-1015 Lausanne, Switzerland
http://lasecwww.epfl.ch

Abstract. In this paper, we present an application of RFIDs for supply-chain management. In our application, we consider two types of readers. On one part, we have readers that will mark tags at given points. After that, these tags can be checked by another type of readers to tell whether a tag has followed the correct path in the chain. We formalize this notion and define adequate adversaries. Morever, we derive requirements in order to meet security against counterfeiting, cloning, impersonation and denial of service attacks.

1 Introduction

Radio Frequency Identication (RFID) tags are being massively deployed in several application and business in order to ensure integrity and security. The deployment of this technology is mainly motivated by the gain in terms of time and cost due to the automation of previously labor-intensive control processes such as access control, authentication, shipment tracking, inventory and logistics, payment... In addition, RFID tags are extensively used to track and identify goods, supplies and equipment. Some of these deployments, like in the biometric identity cards and passports, are used to identify people or keep track of animals. In other applications, these tags are used as a countermeasure to cloning and counterfeiting (especially in the luxury and pharmaceutical industries) as it allows to authenticate the object they are associated with. Large companies are increasingly using RFIDs to extract intelligence from operations that can contribute to their competitiveness and efficiency. Finally, RFIDs are increasingly being considered for convenience and added-value applications for users like in access control where the automation of the process reduces waiting and processing time.

While much attention by researchers has focused on the efficiency, authentication, and privacy aspects (all fundamental concerns), the context in which

[*] The content of this paper is subject to a pending patent by ORIDAO. This work was partially funded by the European Commission through the ICT programme under Contract ICT-2007-216646 ECRYPT II.

[**] Supported by a grant of the Swiss National Science Foundation, 200021-119847/1.

such an application is deployed plays a vital role in its availability so that tags remain valid components for the duration of their projected life-time and forward-security. Strangely, this problem of using the RFID technology is not a popular research topic as only a limited number of papers, like [3], treat the subject. In this work, we focus on a proposed real-life application of RFID tags in supply-chain management.

We propose an application, that we call *pathchecker*, for using RFID tags in supply-chain managements. In our proposal, a supply-chain consists of a series of steps and a tag will be marked at each one of them by "marking readers". Another type of readers, "checking readers", can interact with the tags be able to tell whether, according to some data the tag transmits, went through the right path as it was supposeed to take. The goal of an adversary in such a scheme is to either produce a cloned tag that passes the verification of the checking readers or make a genuine tag which followed a parallel path be accepted in the supply-chain.

Our paper is structured as follows. In Section 2, we describe informally the pathchecker scheme and show examples of the marking and checking protocols between readers and tags. Then, we propose a formalization of this notion and the according security model in Section 3. Sections 4 and 5 deal with the security requirements on the underlying protocols and primitives it uses in order to achieve our desired notion of security. Finally, we propose a secure and private protocol for the authentication protocol in Section 6.

2 Description of the Pathchecker Scheme

Tag **Identity:** ID **State:** S	Update at step r	**Reader** **Identity:** ID_r **Secret:** $F^K_{\mathrm{ID}_r}$
$V_t = \mathsf{extract}(\mathrm{S})$	$\xrightarrow{\quad V_t \quad}$	
$S \leftarrow \mathsf{H}(S, e)$	$\xleftarrow{\quad e \quad}$	$e = F^K_{\mathrm{ID}_r}(\mathrm{ID}_r, V_t)$

Fig. 1. Protocol used to mark the tags at each step of the supply-chain.

We consider a RFID system in which every tag store a state S. This state is initialized with a value which depends on the identity of the tag ID, the application parameters, and a key K. Conversely, the system also consists of readers that possess either a secret function ID_r, unique for each one of them or a secret denoted K.

Depending on the information they hold, there are two kinds of readers. Some readers are used to authenticate the tag and possess K. Others are used to update the tag key and possess a secret $F^K_{\mathsf{ID}_r}$ derived from ID_r and K which represents a step identifier in the supply-chain. We also require that the RFID tags interact with the reader in a particular order known as the correct path. Since authenticating readers need to be able to recompute the internal state of the tags, the secret K has to contain all steps secret. Furthermore, we assume that authenticating readers also know the correct path with the identities of readers.

Tag **State:** S **Identity:** ID	$\mathsf{Identify}$	**Reader** **Secret:** K
$V_t = \mathsf{extract}(\mathrm{S})$	$\xrightarrow{\mathsf{ID}, V_t}$	check V_t

Fig. 2. Protocol used to check whether the tag has followed the right path in the supply-chain.

Concretely, we consider RFID tags that are initialized with a common initial state IV. RFID tags also come with a unique identity (generally known as the EPC number) denoted ID. The tags are attached to some products and then go through a specific path in a supply chain. The path consists of a list of *steps*. At each step, a reader updates the internal state of a tag by using some secret information. At the end of the path, an extra reader can verify that the internal state is consistent with the list of updates it should have received.

The security goal is to make the scheme such that it is impossible to create a tag which passes the verification phase without having run through the exact sequence of steps.

3 Formal Definition

Definition 1 (Pathchecker-scheme). *A* pathchecker-scheme *is a tuple consisting of*

- *a set $\mathcal{I}$ of possible ID's for the tags*
- *a state space $\mathcal{S}$ for the tags*
- *a set $\mathcal{V}$ of possible values*
- *an initial state $\mathsf{IV} \in \mathcal{S}$*
- *a path length n*
- *a function $H : \mathcal{S} \times \mathcal{V} \longrightarrow \mathcal{S}$ to be computed by the tag*
- *an extraction function* $\mathsf{extract} : \mathcal{S} \to \mathcal{U}$ *used by the tag*
- *a family of tuples, indexed by a key $K \in \mathcal{K}$, $\mathcal{F} = \left((F_1^K, \ldots, F_n^K)\right)_{K \in \mathcal{K}}$ consisting of n functions $F_1^K, \ldots, F_n^K : \mathcal{I} \times \mathcal{U} \longrightarrow \mathcal{V}$. By extension we define $\bar{F}_0^K, \ldots, \bar{F}_n^K$ by*

$$\bar{F}_0^K(\mathsf{ID}) = \mathsf{IV}$$
$$\bar{F}_i^K(\mathsf{ID}) = H\left(\bar{F}_{i-1}^K(\mathsf{ID}), F_i^K(\mathsf{ID}, \mathsf{extract}\left(\bar{F}_{i-1}^K(\mathsf{ID})\right)\right) \quad \text{for } i = 1, \ldots, n$$

and a function $V^K : \mathcal{I} \times \mathcal{V} \longrightarrow \{0,1\}$ by

$$V^K(\mathsf{ID}, x) = \begin{cases} 1 \text{ if } x = \bar{F}_n^K(\mathsf{ID}) \\ 0 \text{ otherwise} \end{cases}$$

Clearly, if (ID, x) follows the path, it is no surprise that $V^K(\mathsf{ID}, x) = 1$.

For simplicity, we assume $\mathcal{U} = \mathcal{S}$ and $\mathsf{extract}$ to be the identity function.

Recall that the aim of the pathchecker scheme is to allow readers to automatically check that a given tag has followed the path it was supposed to. The security argument of such an application is to guarantee that a tag accepted by a reader as having followed the path it was supposed to take.

Clearly, the goal of an adversary against this scheme is to produce a tag that did not follow the path in the supply-chain but is recognized by the "checking readers" as having followed it. We can imagine different scenarios and combine them:

- An adversary can introduce a counterfeit product in the middle of the supply-chain, such a product should be detectable by the end of the supply-chain.
- An adversary may take one tag from the middle of the supply-chain, interact with it and make it go through a parallel path. She can also put it back at any step of the supply-chain.
- Tags may be considered weak in the sense that an adversary can open them and get their internal states.

So we consider general adversaries that are able to request the creation of a new RFID tag or the corruption of an existing one (to get its internal state). We also assume that the adversary may have some control over the supply-chain in the sense that she can introduce a tag at any point in the supply-chain or take it at any point from the supply-chain. She is also assumed to be able to freely run protocols with the readers or the tags. Namely, she can query some F_i^K and V^K oracles.

We stress that we do not treat the notion of privacy here.

Definition 2 (Adversary). *A* (q,t)-adversary *against the pathchecker scheme is an algorithm playing the following game:*

```
pick $r$ at random and set $\mathsf{View} = r$
pick $K \in \mathcal{K}$ at random
for $j = 1$ to $q$ do
    $(\mathsf{ID}_j, x_j, i_j) = \mathcal{A}(\mathsf{View})$
    if $i_j > 0$ then
        $y_j = F_{i_j}^K(\mathsf{ID}_j, x_j)$
    else
        $y_j = V^K(\mathsf{ID}_j, x_j)$
    end if
    $\mathsf{View} \leftarrow \mathsf{View} \| y_j$
```

```
end for
$(\mathsf{ID}, x) = \mathcal{A}(\mathsf{View})$
output $(\mathsf{ID}, x)$
```

The total running time of the adversary $\mathcal{A}$ in this game must be at most t. We say that (ID, x) followed the path *if there exists a sequence $j_1, \ldots, j_n$ such that*

- $1 \leq j_1 < \cdots < j_n \leq q$
- *for all k we have $i_{j_k} = k$ and $\mathsf{ID}_{j_k} = \mathsf{ID}$*
- $x_{j_1} = \mathsf{IV}$
- $H(x_{j_n}, y_{j_n}) = x$
- *for all $k < n$ we have $x_{j_{k+1}} = H(x_{j_k}, y_{j_k})$*

We say that the adversary wins *if we have $V(\mathsf{ID}, x) = 1$ but (ID, x) did not follow the path.*

We say that the scheme is (q, t, ε)-secure *if for any (q, t)-adversary the probability to win is at most ε.*

In Section 5, we address a different threat related to cloning attacks in a scenario where genuine tags are tamper resistant.

Note that this definition is more general than the pathchecker scheme proposed in Section 2. In the proposed scheme, the tag identity ID is only used at the first step F_1: the personalization step. Other F_i functions do not use ID.

4 Parallel Path Detection

In a strong security model we assume that the adversary has entire control on the overall process (except by opening a reader). Such an adversary can get the full internal state of a tag (e.g. by physical attack, or, in the case of the described case, by simply getting x, sending a fake y and then getting a new x form which it is easy to recover the missing information by exhaustive search) and create fake tags with chosen identity and chosen state. Without loss of generality, this adversary can thus reduce to the scenario in the definition.

Theorem 1. *Let us consider a pathchecker scheme. With he above notations, there is a generic transform for a (q, t)-adversary $\mathcal{A}$ with winning probability ε into $(q, t + \mu)$-adversaries $\mathcal{A}_1, \ldots, \mathcal{A}_{q+1}$ with respective winning probabilities $\varepsilon_1, \ldots, \varepsilon_{q+1}$ which make no query to V^K and such that*

$$\varepsilon \leq \varepsilon_1 + \cdots + \varepsilon_{q+1}$$

Since the function H is only used in V^K in the game played by the adversaries, this result shows that we can reduce to adversaries in which H is never used.

Proof. First of all, we assume without loss of generality that adversaries do not make queries to V^K which trivially lead to the answer 1. Namely, if (ID, x) followed the path during the game, we assume that it is not queried to V^K. Next, we consider the final output (ID, x) as a *fictive* final query to V^K. Clearly,

$\mathcal{A}$ wins if and only if one among all queries to V^K (including the fictive one) answers 1. Then, we define $\mathcal{A}_i$ which simulates $\mathcal{A}$ except that the first i queries to V^K are not made and the answer 0 is simulated and the execution stops at the $i+1$th query. Clearly, $\mathcal{A}$ wins the game with the random tape r, K if and only if at least one of the $\mathcal{A}_i$ adversaries wins the game with the same random tape r, K. We conclude by defining μ to be the overhead complexity in the simulation of the adversaries. □

This leads us to the following statement:

> **In a strong adversarial model, the hash function H is irrelevant for security.**

We now introduce the definition of a pseudo-random function in order to provide a sufficient condition for security.

Definition 3 (Pseudorandom Function (PRF)). *A* pseudorandom function (PRF) *is a tuple consisting of*

- *a domain $\mathcal{D}$, a range $\mathcal{R}$, and a key space $\mathcal{K}$*
- *a family $\mathcal{F} = \left(F^K\right)_{K\in\mathcal{K}}$ of functions $F^K : \mathcal{D} \longrightarrow \mathcal{R}$*

*A (q,t)-*adversary *$\mathcal{A}$ is an algorithm playing the following game:*

```
pick a uniformly distributed random bit $b$
if $b = 0$ then
    pick a uniformly distributed random function $F$ form $\mathcal{D}$ to $\mathcal{R}$
else
    pick $K \in \mathcal{K}$ at random
    set $F$ to the $F^K$ function
end if
pick $r$ at random and set $\mathsf{View} = r$
for $j = 1$ to $q$ do
    $x_j = \mathcal{A}(\mathsf{View})$
    $y_j = F(x_j)$
    $\mathsf{View} \leftarrow \mathsf{View} \| y_j$
end for
$\tilde{b} = \mathcal{A}(\mathsf{View})$
output $\tilde{b} \oplus b$
```

The total running time of $\mathcal{A}$ in this game must be at most t. We say that the adversary wins *if the output is 0. We say that the PRF is (q,t,ε)-*secure *if for any (q,t)-adversary the probability to win is at most $\frac{1}{2}+\varepsilon$.*

Theorem 2. *Consider a pathchecker scheme with the previous notations and let $F^K(\mathsf{ID}, x, i) = F_i^K(\mathsf{ID}, x)$, $\mathcal{D} = \mathcal{I} \times \mathcal{U} \times \{1, \ldots, n\}$ and $\mathcal{R} = \mathcal{V}$. There exists a constant μ such that if $\left(F^K\right)_{K\in\mathcal{K}}$ is a $(q+n, t+n\mu, \varepsilon)$-secure PRF, then the pathchecker-scheme is (q,t,ε')-secure with*

$$\varepsilon' = (q+1)\left(2\varepsilon + \frac{nq}{\#\mathcal{V}}\right)$$

Proof. We assume that we have a PRF and we try to upper bound the winning probability of a (q, t)-adversary $\mathcal{A}$. Let μ_1 be the overhead defined by Th. 1. We construct $\mathcal{A}_1, \ldots, \mathcal{A}_{q+1}$ as before which never query V^K. We define $\mathcal{A}'_i$ as follows.

```
Input: View
simulate $q = \mathcal{A}_i(\mathsf{View})$ and look at what is scanned by $\mathcal{A}_i$ in View
if $q$ is an intermediate query then
  output ← $q$
else
  parse $q = (\mathsf{ID}, x)$
  parse the unscanned part of View into $y_1 \| \cdots \| y_j$ (with $0 \leq j \leq n$)
  state = IV
  for $i = 1$ to $j$ do
    state ⟵ $H(y_i, \mathsf{state})$
  end for
  if $j < n$ then
    output ← $(\mathsf{ID}, \mathsf{state}, j + 1)$
  else
    if state = $x$ then
      output ← 1 (final output)
    else
      output ← 0 (final output)
    end if
  end if
end if
yield output
```

We let $n\mu_2$ be the overhead in the simulation and $\mu = \max(\mu_1, \mu_2)$. Clearly, we obtain a $(q + n, t + n\mu)$-adversary against the PRF. When $b = 1$ in the PRF game, the $\mathcal{A}'_i$ adversary wins with random tape r, K if and only if $\mathcal{A}_i$ wins the pathchecker game with random tape r, K. When $b = 0$, there is at least one of the final n queries by $\mathcal{A}'_i$ which is new thus returns a random value. This value is different from all previous ones with high probability and the same for the forthcoming ones. More precisely, the wining probability is higher than $1 - \frac{nq}{\#\mathcal{V}}$. Thus, the overall wining probability of $\mathcal{A}'_i$ is

$$\Pr[\mathcal{A}'_i \text{ wins}] \geq \frac{1}{2} + \frac{1}{2}\Pr[\mathcal{A}_i \text{ wins}] - \frac{nq}{2\#\mathcal{V}}$$

Thanks to the PRF property we deduce

$$\Pr[\mathcal{A}_i \text{ wins}] \leq 2\varepsilon + \frac{nq}{\#\mathcal{V}}$$

We conclude by using the inequality of Th. 1. □

This leads us to the following conclusion:

> **It is enough that readers use a PRF to guaranty security in a strong adversarial model.**

5 Security against Genuine Tag Manipulation

In the previous section, we studied the pathchecker scheme to make sure that something goes sequentially to every step of a path to produce a final valid number. This means that the correct sequence of value must be obtained from each step but this does not mean that a tag shall receive it in the right order. In settings where genuine tags are tamper-resistant and there are no counterfeit tags, we study the problem of making a genuine tag end up in an acceptable state without following the path. Indeed, we could still imagine a tag cloning attack to clone genuine tags in the correct final state. These clones would pass the verification. To prevent from this without having to install a heavy audit tool mechanism to detect tag with same ID's, we must weaken the adversarial model.

We consider the problem of making a tag end up in a state which is accepted by the final verification without having received values produced by every step reader in exactly the right sequence. In this settings, we consider adversaries who cannot create genuine tags. Clearly, this model is weaker than the previous one.

The idea behind is that genuine tags with H embedded are legally protected and that it is easy to see if a tag is a counterfeit. Tags passing through the final V^K verification are not counterfeit but genuine tags are all released with the same initial state. We assume that the adversary cannot physically tamper the state of a genuine tag so the only interface with the tag is by sending values. More precisely, we consider the following definition:

Definition 4 (Weak adversary). *A (q, q', t)-weak adversary is a ploynomial-time algorithm playing the following game.*

```
pick $r$ at random and set $\mathsf{View} = r$
pick $K \in \mathcal{K}$ at random
for $j = 1$ to $q$ do
    $(\mathsf{ID}_j, x_j, i_j) = \mathcal{A}(\mathsf{View})$
    if $i_j > 0$ then
        $y_j = F^K_{i_j}(\mathsf{ID}_j, x_j)$
    else
        $y_j = V^K(\mathsf{ID}_j, x_j)$
    end if
    $\mathsf{View} \leftarrow \mathsf{View} \| y_j$
end for
$(\mathsf{ID}_1, \ldots, \mathsf{ID}_\ell, (i_1, y_1), \ldots, (i_{q'}, y_{q'})) = \mathcal{A}(\mathsf{View})$
order $\ell$ tags $T_1, \ldots, T_\ell$ in states $x_1 = \cdots = x_\ell = \mathsf{IV}$ and respective identities
    $\mathsf{ID}_1, \ldots, \mathsf{ID}_\ell$
for $j = 1$ to $q'$ do
    send $y_j$ to $T_{i_j}$
end for
```

The adversary wins if there exists one i such that tag ID_i end up in a state x such that $V^K(\mathsf{ID}_i, x) = 1$ but the list of values that he received is not the sequence $(\bar{F}_1^K(\mathsf{ID}_i), \ldots, \bar{F}_n^K(\mathsf{ID}_i))$.

Clearly, we can restrict without loss of generality to adversaries using a single tag, namely $\ell = i_1 = \cdots = i_q = 1$. The attack game consists of adaptively selecting ID based on training by querying the oracles and finding a sequence $y_1, \ldots, y_q$ such that the sequence defined by $x_0 = \mathsf{IV}$ and $x_i = H(x_{i-1}, y_i)$ verifies $x_q = \bar{F}^K(\mathsf{ID})$ but there is at least one $i \leq n$ such that $y_i \neq \bar{F}_i^K(\mathsf{ID})$.

If the pathchecker scheme is secure, the final ID and the final state of the tag must follow the path, meaning that the training phase has got the correct sequence $\bar{F}_i^K(\mathsf{ID})$. The next question is whether the tag can end up in the correct state if not fed by this correct sequence. Theorem 3 gives an answer to this question by showing that it is sufficient that H is a 2nd-preimage resistant hash function. The definition of such a hash function is given below:

Definition 5 (2nd-preimage resistant hash function). *A* hash function *is a tuple consisting of*

- *a domain $\mathcal{V}$, a range $\mathcal{R}$, and a key space $\mathcal{K}$*
- *a family $\mathcal{H} = \left(\mathsf{H}^K\right)_{K \in \mathcal{K}}$ of functions $\mathsf{H}^K : \mathcal{V} \longrightarrow \mathcal{R}$*

*A (q, t)-*adversary *$\mathcal{A}$ is an algorithm playing the following game:*

```
1: pick $K \in \mathcal{K}$ at random
2: set $\mathsf{H}$ to the $\mathsf{H}^K$ function
3: pick $x \in \mathcal{V}$ at random and set $\mathsf{View} = x$
4: for $j = 1$ to $q$ do
5:     $x_j = \mathcal{A}(\mathsf{View})$
6:     $y_j = \mathsf{H}(x_j)$
7:     $\mathsf{View} \leftarrow \mathsf{View} \| y_j$
8: end for
9: $\tilde{x} = \mathcal{A}(\mathsf{View})$
10: output 1 iff $\mathsf{H}(x) = \mathsf{H}(\tilde{x})$ and $x \neq \tilde{x}$
```

The total running time of $\mathcal{A}$ in this game must be at most t. We say that the adversary wins *if the output is 1.*

*We say that the hash function is (q, t, ε)-*2nd-preimage resistant *if for any (q, t)-adversary the probability to win is at most ε.*

Theorem 3. *Consider a pathchecker scheme. There exists a constant μ' such that if $(\mathsf{H}(\mathsf{IV}, \cdot))_{\mathsf{IV} \in \mathcal{S}}$ is a $(q + n, t + n\mu, \varepsilon)$-2nd-preimage resistant hash function and $\left(F^K\right)_{K \in \mathcal{K}}$ is a $(1, t + n\mu_2, \varepsilon_2)$-secure PRF , then the pathchecker-scheme is (q, t, ε')-weakly secure with*

$$\varepsilon' = \frac{1}{1 - \varepsilon_2} \left(\varepsilon + \frac{q}{\#\mathcal{S}} \right)$$

Proof. We can restrict without loss of generality to adversaries using a single tag and thus we consider the following adversary:

```
pick $r$ at random and set $\mathsf{View} = r$
pick $K \in \mathcal{K}$ at random
for $j = 1$ to $q$ do
    $(\mathsf{ID}, x_j) = \mathcal{A}(\mathsf{View})$
    $y_j = F_{i_j}^K(\mathsf{ID}_j, x_j)$
    $\mathsf{View} \leftarrow \mathsf{View} \| y_j$
end for
$(\mathsf{ID}, y_1, \ldots, y_{q'}) = \mathcal{A}(\mathsf{View})$
order 1 tag $T$ in state $\mathsf{IV} = x$ and identity $\mathsf{ID}$
for $j = 1$ to $q'$ do
    send $y_j$ to $T_{i_j}$
end for
```

The adversary wins if there exists one i such that tag ID_i end up in a state x such that $V^K(\mathsf{ID}_i, x) = 1$ but the list of values that he received is not the sequence $(\bar{F}_1^K(\mathsf{ID}_i), \ldots, \bar{F}_n^K(\mathsf{ID}_i))$. Recall that the scheme is weakly-secure against a (q, t, ε)-weak adversary that uses q queries, runs in time less than t, wins with probability of at most ϵ.

In the case where the function F^K is a $(1, t + \mu, \varepsilon_2)$-secure PRF, the adversary makes no distinction if it is replaced by a random function F except with probability ε_2. Clearly, this attack corresponds to a 2nd-preimage attack againt H.

The adversary $\mathcal{A}'$ wins with random tape r, K if and only if $\mathcal{A}$ wins with random tape r, K and does not distinguish F^K from F. Since the y_j produced by $\mathcal{A}$ is the output of a random function, this value is different from all previous ones with high probability and the same for the forthcoming ones. More precisely, the wining probability is higher than $1 - \frac{q}{\#\mathcal{S}}$ and we deduce the wining probability of $\mathcal{A}'$

$$\Pr[\mathcal{A}' \text{ wins}] \geq (1 - \varepsilon_2) \Pr[\mathcal{A} \text{ wins}] - \frac{q}{\#\mathcal{S}}$$

Assuming that H is 2nd-preimage resistant, we obtain

$$\Pr[\mathcal{A} \text{ wins}] \leq \frac{1}{1 - \varepsilon_2} \left(\epsilon + \frac{q}{\#\mathcal{S}} \right)$$

□

We deduce that if $y \mapsto H(\mathsf{IV}, y)$ is 2nd preimage resistant then no adversary can win the above game scenario with significant success probability. This leads us to the following conclusion:

> **In a weak attack model, 2nd preimage resistance of $y \mapsto H(\mathsf{IV}, y)$ and PRF properties for F are enough to guaranty that no genuine tag with acceptable final state can be created without receiving the expected sequence of values obtained at each corresponding step.**

6 Security against Tag Impersonation

In the previous sections, we have shown that, with the adequate security assumptions on F and H, an adversary cannot set a tag to a state that will allow this latter to be accepted by a "checking reader".

However, there exists a flaw in the protocol we described in Fig. 2. It is sufficient for an adversary to get the message V_t hat the tag sends to the reader and then forge a counterfeite tag that will send this value each time it is asked for checking. This is called a replay attack.

To thwart this attack, we introduce a challenge (each time different) that the reader sends at the begining of the protocol. The tag will then compute its answers by applying H with input its state S and the received challenge c. Upong receiving the response, the reader checks whether the tag followed the right path. A description of the protocol is shown in Fig. 3.

Tag **State:** S	Authenticate	**Reader** **Secret:** K
	$\xleftarrow{\quad c \quad}$	pick c at random
$r \leftarrow \mathsf{extract}(\mathsf{H}(S, c))$	$\xrightarrow{\quad r \quad}$	check r

Fig. 3. Challenge-reponse protocol to check whether the tag has followed the right path in the supply-chain.

Since our security model is weaker than the one proposed in [5]. We can reuse the results of [5, Theorem 13.] to prove the security of this protocol under the assumption that $(\mathsf{H}(\mathsf{IV}, \cdot))_{\mathsf{IV} \in \mathcal{S}}$ is a secure PRF. Furthermore, this protocol is weakly-private in under the same assumption.

Using these results, we conclude:

> **In a strong attack model, PRF property for $y \mapsto H(\mathsf{IV}, y)$ is enough to guaranty that no adversary can impersonate a tag to the reader.**

7 Protection from Denial of Service Attacks

In the original version of the pathchecker described in Section 2, any reader can try to run the update protocol with a tag of Fig. 1, making it desynchronize from honest readers. This results in a denial of service attack as the authenticating readers will not be able to verify the path a tag went through. To avoid this, we should have a reader authentication integrated in the update protocol.

We could have a part of V_c to be provided by the reader for reader authentication but this would impose increasing the minimal size of registers. We could avoid wasting bits by using the V_t window for reader authentication as well.

tag **Identity**: ID_t **State**: S	Update at step r (ver.2)	**reader** **Identity**: ID_r **Secret**: $F^K_{\mathsf{ID}_r}$
$V_t = \mathsf{extract}(S)$	$\xrightarrow{\mathsf{ID}_t, V_t}$	$e = F^K_{\mathsf{ID}_r}(\mathsf{ID}_r, V_t)$
$S' = \mathsf{H}(S, e)$	$\xleftarrow{e, V'_t}$	$V'_t \| V'_c = \mathsf{H}(S, e)$
if $V'_t \neq \mathsf{extract}(S')$ abort		
$S \leftarrow S'$		

Fig. 4. Update protocol with mutual authentication.

Following the update protocol, the tag sends its current V_t value extracted from its current state along with its identity. The reader uses it to figure out what is the current state S of the tag ID. Then, the reader computes $e = F^K_{\mathsf{ID}_r}(\mathsf{ID}_r, V_t)$ and $V'_t \| V'_c = \mathsf{H}(S, e)$ and sends e and V'_t to the tag. The tag then computes $S' = \mathsf{H}(S, e)$ and replaces S by this value if V'_t is equal to the ouput of $\mathsf{extract}(S')$. This protocol is depicted in Fig. 4.

However, this protocol assumes that an updating reader can authenticate a tag. This has an important drawback: The amount of computation needed for the protocol increases by the cost of searching ID_t in the database to get S or to reconstruct S using all secret by updating readers. Furthermore, this assumption may not be satisfied as such an operation requires that updating readers have access to the database or to all the updating readers' secrets.

tag **Identity**: ID_t **State**: $S, K_t = G_{S_u}(\mathsf{ID}_t)$	Update at step r (ver.3)	**reader** **Identity**: ID_r **Secret**: $F^K_{\mathrm{ID}_r}, S_u$
$a \leftarrow \mathsf{random}$, $V_t = \mathsf{extract}(S)$	$\xrightarrow{\mathsf{ID}_t, V_t, a}$	$e = F^K_{\mathrm{ID}_r}(\mathrm{ID}_r, V_t)$ $K_t = G_{S_u}(\mathsf{ID}_t)$
if $V'_t \neq \mathsf{extract}(\mathsf{H}(K_t, a \| e))$ abort	$\xleftarrow{e, V'_t}$	$V'_t \| V'_c = \mathsf{H}(K_t, a \| e)$
$S \leftarrow \mathsf{H}(S, e)$		

Fig. 5. Update protocol using an authentication key.

In order to relax this assumption, we introduce a secret key S_u, shared between updating readers, and used by these latter to authenticate themselves to the tags. In consequence, we assign to each tag a specific secret K that depends on its identity ID_t and only computable from S_u using a keyed one-way function G as $K_t = G_{S_u}(\mathsf{ID}_t)$.

In this variant, described in Fig. 5, the tag will send its identity ID_t along with V_t computed as before. Additionally, it sends a challenge a chosen at random. After deriving the key $K_t = G_{S_u}(\mathsf{ID}_t)$, the reader computes e as before and $V'_t \| V'_c = \mathsf{H}(K_t, a \| e)$ to authenticate itself to the tag. Upon receiving e and V'_t, the tag verifies that V_t' matches with its computed value $\mathsf{H}(K_t, a\|e)$ and replaces its state S by $\mathsf{H}(S, e)$ in case of success. Note that hashing e together with a has the effect of authenticating e at the same time. This is important to avoid desynchronization attacks of type man-in-the-middle.

Despite the fact that this variant needs more storage capacity on the tag side for K_t and more computational power, 2 hashes instead of 1, it presents the advantage of protoecting the system from denial of service atacks while keeping the complexity of the protocol relatively low.

8 Conclusion

In this paper, we have proposed a concrete application for RFID tags in supply-chain managements. The application addresses practical problems and risks that a company may have to confront by introducing this technology. We formalized our proposal, defined a security model and developped a number of attack scenarios.

In the first one, we have shown that using a pseudo-random function on the reader side prevents any adversary from creating a tag that will be accepted by readers as having passed through the supposed path. We also considered that adversaries may try to "inverse" the process and extract the secret information included in the tag and how such an attack can be avoided if the hash function used on the RFID chip is resistant to 2nd preimage attacks. At last, we modified the authentication protocol in order to be secure against cloning attacks. We also proposed a variant to the update protocol resistant to denial of service attacks.

Althrough early research on hash functions for RFID tags were not really optimistic [2], recent proposals have shown some promising results like the work of Bogdanov et al. [1] and Shamir [4]. Following on this area of research by developing hash functions addressing the needs and specificity RFID tags is fundamental for the upcoming introduction of RFID tags in large systems.

Acknowlegments

We would like to thank M. Nicolas Reffé from ORIDAO for proposing the Pathchecker application and initiating this work.

References

1. Andrey Bogdanov, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, and Yannick Seurin. Hash functions and RFID tags: Mind the gap. In Elisabeth Oswald and Pankaj Rohatgi, editors, *Cryptographic Hardware and Embedded Systems - CHES 2008, 10th International Workshop, Washington, D.C., USA, August 10-13, 2008. Proceedings*, volume 5154 of *Lecture Notes in Computer Science*, pages 283–299. Springer, 2008.

2. Martin Feldhofer and Christian Rechberger. A case against currently used hash functions in RFID protocols. In Robert Meersman, Zahir Tari, and Pilar Herrero, editors, *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops, OTM Confederated International Workshops and Posters, AWeSOMe, CAMS, COMINF, IS, KSinBIT, MIOS-CIAO, MONET, OnToContent, ORM, PerSys, OTM Academy Doctoral Consortium, RDDS, SWWS, and SeBGIS 2006, Montpellier, France, October 29 - November 3, 2006. Proceedings, Part I*, volume 4277 of *Lecture Notes in Computer Science*, pages 372–381. Springer, 2006.
3. Ari Juels, Ravikanth Pappu, and Bryan Parno. Unidirectional key distribution across time and space with applications to rfid security. In Paul C. van Oorschot, editor, *Proceedings of the 17th USENIX Security Symposium, July 28-August 1, 2008, San Jose, CA, USA*, pages 75–90. USENIX Association, 2008.
4. Adi Shamir. SQUASH - a new MAC with provable security properties for highly constrained devices such as RFID tags. In Kaisa Nyberg, editor, *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*, volume 5086 of *Lecture Notes in Computer Science*, pages 144–157. Springer, 2008.
5. Serge Vaudenay. On privacy models for RFID. In Kaoru Kurosawa, editor, *Advances in Cryptology - ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, December 2-6, 2007, Proceedings*, volume 4833 of *Lecture Notes in Computer Science*, pages 68–87. Springer, 2007.