RFID Path Authentication, Revisited

Mohammad Saiful Islam Mamun and Atsuko Miyaji
Japan Advanced Institute of Science and Technology (JAIST)
1-1 Asahidai, Nomi, Ishikawa, 923-1211, Japan.
{mamun, miyaji}@jaist.ac.jp

***Abstract*—In an RFID-enabled supply chain, where items are outfitted with RFID tags, path authentication based on tag enables the destination checkpoints to validate the route that a tag has already accessed. In this work, we propose a novel, efficient, privacy-preserving path authentication system for RFID-enabled supply chains. Compared to existing Elliptic curve Elgamal Re-encryption (ECElgamal) based solution, our Homomorphic Message authentication Code on arithmetic circuit (HomMAC) based solution offers less memory storage (with limited scalability) and no computational requirement on the reader. However, unlike previous schemes, we allow computational ability inside the tag that consents a new privacy direction to *path privacy* proposed by Cai et al. in ACNS'12. In addition, we customize a polynomial-based authentication scheme (to thwart potential tag impersonation and Denial of Service (DoS) attacks), so that it fits our new path authentication protocol.**

***Keywords*: Path Authentication, Arithmetic Circuit, Homomorphic MAC, Mutual Authentication.**

I. Introduction

A Supply Chain Management (SCM) controls and manages all of materials and information in the logistics process from acquisition of raw materials to product delivery to the end user. This yields convenience and efficiency, which leads to productivity gains. With the growing nature of SCM, it is crucial to construct protocols that enable the end user to verify the security and privacy not only of the tags but also the path that the tag passes through. Therefore, path authentication in RFID-enabled SCM is important, for it helps defend product genuineness by ensuring product derivation. Hence, the integrity of a supply chain. More clearly, when a tag reaches to the end of its supply chain, it would be desirable that, if the authentication results of several intermediate readers could be accompanied by a cryptographic proof guaranteeing their correctness from the authentication tag, *s.t.*, no intermediate reader was omitted (or selected wrongly) by the tag, either deliberately or not.

A number of tag authentication schemes, that have been proposed e.g., [1], [10]–[13], cannot be used directly for path authentication, either because they incur high computational overhead or they lack simultaneous online access to all the parties in the supply chain [3]. Nevertheless, after the first proposal by Blass et al. [2], a number of practical solutions have been followed e.g., [3]–[5], [14] to offer secure and privacy preserving path authentication in RFID-enabled SCM. There are two kinds of path authentication system: *static path*, where a valid *path* is predetermined and is shared with the destination checkpoint (e.g., [3]); and, *dynamic path*, where the path is generated dynamically and every node in the path can track the validation of the path (e.g., [4]). No prior *static path* based authentication schemes consider mutual authentication between the tag and intermediate readers, either because they assume that the communication channel between the reader and tag during path authentication is secure, or because they presume tag authentication implicitly. For instance, in [3], authors assume that the reader will update the tag's state only after successful authentication. However, this scheme does not include any tag authentication explicitly. Although some dynamic path authentication schemes incorporate mutual authentication (e.g., [6]) into their proposal.

In this paper, we stress on static path based path authentication by Cai et al. [3][1], where the authors define a new combined privacy notion and provide an efficient solution for path authentication without sharing any secrets among supply chain parties.

Main Contribution. Our contribution includes the following:

- We instantiate a new variant of path authentication scheme with arithmetic circuit based HomMAC. Note that building blocks of previous static path based authentication systems were mainly from expensive elliptic curve ElGamal re-encryption (ECElGamal) and security of the schemes were primarily either from Pseudo Random Function (PRF) or Homomorphic MAC (HMAC). Security of our scheme also stems from PRF.
- We propose state update operations to be held inside the tag. It offers more security and reasonable privacy since the intermediate readers obtain no knowledge about current state of the tag. However, it introduces a lightweight computation (polynomial operation) in the tag. Note that likewise other existing schemes, it is also manageable (even easier) to update state information into the readers (and hence no computation inside the tag).
- We consider a relaxed privacy assumption[2] that allows adversary to query the reader during Move oracle. Since the reader in our scheme coveys no information about the tag's current state during tag movement, disallowing adversary to query only the tag is sufficient enough for the path privacy experiment to fail. This assumption is more practical and formal. Thus, we redefine the generic

[1] an extended and more practical privacy variant of [2].

[2] Adversary in [3] is not allowed to query either the reader or tag during Move operation run by the game challenger.

privacy oracles of Cai et al. [3] in section 4.1.
- Unlike the scheme in [3], we propose two strategies (with or without path information) for checkpoint verification that conform to a more stringent protection of path privacy in the supply chain.
- Compare to [3], our scheme requires less storage but poses conditional scalability. However, it could be transformed into a fully scalable variant[3].
- We propose a polynomial based mutual authentication scheme from [9] that can optionally be integrated to our path authentication solution. We modify the protocol in order to conform secret and public parameters of our path authentication solution. In addition, we convert the existing *tag authentication* protocol to a *mutual authentication* protocol, significantly reduce communication, storage and computation overhead into the tag effectively.
- We purport how to accommodate a batch of tags that must follow the same path to the destination.

II. Background

A. Supply Chain Management

In a SCM network every product that reaches an end user represents the cumulative effort of multiple parties like *manufacturer, distributor, wholesaler*. These parties are referred to collectively as the supply chain. Parties in a supply chain are *linked* together through information flows that allow various supply chain partners to coordinate the day-to-day flow of products up and down through the supply chain path. It can be represented as a Directed Acyclic Graph (DAG) $G = (V, E)$, where V is a set of nodes and E is a set of edges. Each edge $e \in E$, $e = (v_i, v_{i+1})$ s.t., $(v_i, v_{i+1}) \in V$ represents a *step* in the supply chain path. An RFID tag attached onto every product in the supply chain contains a unique identification about the product. A valid supply chain is a path (or a set of paths) in the DAG. Supply chain authentication is about verifying that an item (or rather, a cryptographic token, supposed to be attached to the item) is forwarded along a valid supply chain.

A valid finite path $P = (v_0, \cdots, v_r)$ is a pre-defined path set by the coordinator e.g., the manufacturer that an RFID-enabled product requires to follow, where v_0 is the entrance of a product to supply chain and v_r is its final destination to arrive. An RFID-enabled SCM consists of an issuer (e.g., manufacturer) $\mathcal{M}$, a set of check points (e.g., retailers) $\mathcal{D}$, a set of ordinary readers (e.g., distributors, wholesalers etc.) $\mathcal{R}$, and a set of tags $\mathcal{T}$. $\mathcal{M}$ initializes the whole system by providing identifiers to the $\mathcal{R}, \mathcal{T}$ and storing necessary information into the tag and reader. Each reader in the path provides the contents to run status update operation inside the tag, while it moves through a supply chain. Once the tag arrives at any of the checkpoints $\mathcal{D}$, it can check the validity of the *tag* as well as the *path* it followed from the $\mathcal{M}$ to $\mathcal{D}$. More precisely, the system has the following functions:

- **Initialize**(λ): Given the security parameter λ, an SCM system defines a supply chain network G including an issuer $\mathcal{M}$, a set of d checkpoints $\mathcal{D}$, a set of n tags $\mathcal{T}$, a set of r ordinary readers $\mathcal{R}$, and a set of v valid paths $\mathcal{P}_v$.
- **Reader Authentication** ($\mathcal{R}_j$): This function transforms the identity information $ID_{\mathcal{R}_j}$ of the reader $\mathcal{R}_j$ to the tag. We assume that the tag along the path to be honest (without mutual authentication), that means, it accepts data from the reader only after successful authentication and updates its internal state st thereby.
- **Tag Evaluation**($\mathcal{T}_i$): A function that incorporates the new reader's information $ID_{\mathcal{R}_j}$ into the tag $\mathcal{T}_i$ in order to update the internal state $st_{\mathcal{T}_i}$ of the tag $\mathcal{T}_i$.
- **Verification** ($st_{\mathcal{T}_i}$): This function verifies whether a certain $\mathcal{T}_i$ has followed a valid path P_v and returns *True*. Otherwise it returns *False*.

B. Building Blocks

Labelled Program: The notion of labeled data or program was first introduced by Gennaro et al. [8]. Let an entity (e.g., checkpoint) want to authenticate some data $\tau := \{\tau_0, \tau_1, \cdots \tau_r\}$ (e.g., tag/reader's data) with respect to their corresponding labels $\mathcal{I} := \{\iota_0, \iota_1, \cdots \iota_r\}$ (e.g., tag/reader's unique identifier) where $\iota_i \in \{0,1\}^*$. A labeled program can be defined by $\mathcal{P} := (f, \mathcal{I})$ where $f : \{0,1\}^r \rightarrow \{0,1\}$ is a circuit on data τ. Output of a labeled program can be computed over data τ provided by different entities (e.g., Readers) at different times.

Arithmetic Circuit: An arithmetic circuit f over the variables or data $\tau = \tau_0, \tau_1, \cdots, \tau_r$ is a labelled directed acyclic graph G with its leaves labelled as $\mathcal{I} := \{\iota_0, \iota_1, \cdots \iota_r\}$ and internal nodes labelled as gate $\mathcal{O} := \{+, \times\}$ operations. The circuit has a designated output ρ.

In this paper, we consider an arithmetic circuit f over a field $\mathbb{Z}_p$ such that $f : \mathbb{Z}_p^r \rightarrow \mathbb{Z}_p$ for a prime p. The circuit f has bounded fan-in, that is, each of its internal nodes has at most two children. The size of a circuit, $\mathsf{size}(f)$ is the number of gates/vertices in underlying graph. The depth of the circuit, $\mathsf{depth}(f)$ is the length of the longest directed path in the circuit. Note that an arithmetic circuit can compute a polynomial in the natural way and every polynomial defines a unique function. An input gate of an arithmatic circuit can compute a polynomial it is tagged by the labels. A sum gate '$+$' computes the sum of *two* polynomials obtained from the incoming wire in the graph. Similarly, a product gate '$\times$' computes the product of *two* polynomials.

However, the *degree of a circuit* is delineated by the maximal degree of the gates in the circuit while the *degree of a gate* is defined by the total degree of the polynomial it computes. Note that all the polynomials belong to the class VP, the algebraic analog of class P. That is, all polynomials of polynomially bounded degree can be realized by an arithmetic circuit family with polynomially bounded size [16].

Homomorphic Authentication Scheme: In a homomorphic message authenticator scheme, an entity can authenticate data

[3]postponed to the full version of the paper.

System parameter	$(\lambda, \mathcal{F}, h, K, s, p, f, \mathfrak{f}, b)$ p is a prime of λ bits Polynomial $\mathfrak{f} := \{\mathfrak{f}_1(\alpha,\beta), \mathfrak{f}_2(\alpha,\beta), \cdots, \mathfrak{f}_m(\alpha,\beta)\}$ Hash $h : \{0,1\}^\lambda \to \mathbb{Z}_p$ Pseudo Random Function $\mathcal{F}_K : \{0,1\}^\lambda \to \mathbb{Z}_p$ Arithmetic circuit f: $\mathbb{Z}_p^r - \mathbb{Z}_p$, where $\|f\| = r$
Initialize Tag $\mathcal{T}_i$ Without `Auth` (y_0, y_1, f) With `Auth` $(y_0, y_1, Q, b, f, \mathcal{T}_i, \mathfrak{f})$	$y(z) = y_0 + y_1 z$ **s.t,** $\sigma = (y_0, y_1)$ $y_0 = h(\texttt{tag\_data})$ $\mathcal{T}_i = \mathcal{F}_K(\iota_0)$ s.t., $\iota_0 =$ `Tag ID` $y_1 = (\mathcal{T}_i - y_0)/s \bmod p$ $Q \leftarrow \mathsf{max}(\ \|f\|,\ (b-1)m^2 + m)$
Initialize Readers $(\mathcal{R}_1, \cdots, \mathcal{R}_r)$ Without `Auth`$(y_0^j, y_1^j, K, s, p, \mathfrak{f}, h\mathcal{F})$ With `Auth`$(y_0^j, y_1^j, \mathcal{T}_i, y_0^{\mathcal{T}_i}, K, s, p, \mathfrak{f}, h, \mathcal{F})$	$y^j(z) = y_0^j + y_1^j z$ s.t, $\sigma_j = (y_0^j, y_1^j)$ $y_0^j = h(\texttt{reader\_data})$ $y_1^j = (\mathcal{F}_K(\iota_j) - y_0^j)/s \bmod p$ s.t., $\iota_j =$ `Reader ID` $y_1^{\mathcal{T}_i} = y_0$ of $\mathcal{T}_i$ (`Tag ID` ι_i)
Initialize Checkpoint $\mathcal{D}_k$ Without `Path`-info $(s, \tau, \Lambda, \sigma)$ With `Path`-info $(s, p, f, K, \tau, \mathcal{F}, \{\iota_0, \cdots, \iota_r\}, \sigma)$	$\tau = y_0$ where $\sigma_{i,r} = (y_0, \cdots, y_d)$ ($\sigma_{i,r}$ is evaluated by $\mathcal{T}_i$ with $\mathcal{R}_r$) $\Lambda = f(\eta_0, \cdots, \eta_r)$ where $\eta_i = \mathcal{F}_K(\iota_i)$ and $\iota_i \leftarrow\ P = \{\iota_0, \cdots, \iota_r\}$

Fig. 1. Path Authentication Initialization

τ with its secret key $\mathbf{sk}$. Later evaluators can homomorphically execute an arbitrary program $\mathcal{P}$ over τ and subsequently generate an authentication tag σ without knowing sk. Note that σ certifies $\mathcal{P}(\tau)$. Finally a verifier that knows sk can assert whether σ is indeed the output of the $\mathcal{P}(\tau)$ without knowing τ. A Homomorphic Message Authentication scheme consists of the following *four* algorithms:

- $\mathsf{KGen}(1^\lambda)$: On input of the security parameter λ, it generates a key pair $(\mathsf{sk}, \mathsf{ek})$ where $\mathbf{sk}$ is the secret key and ek is the public evaluation key.
- $\mathsf{Authentication}(\mathsf{sk}, \iota, \tau)$: Given the secret key sk, a label ι and a message data τ, it outputs a succinct tag σ.
- $\mathsf{Evaluate}(\mathsf{ek}, f, \sigma)$: On input of the evaluation key ek, a circuit $f : \mathbb{Z}_p^r \to \mathbb{Z}_p$ and a set of authenticating tags $(\sigma_0, \cdots, \sigma_r)$, this algorithm outputs a new tag σ.
- $\mathsf{Verify}(\mathsf{sk}, \tau, \mathcal{P}, \sigma)$: On input of the secret key sk, a program $\mathcal{P} := (f, \mathcal{I})$ where $\mathcal{I} := \{\iota_0, \iota_1, \cdots \iota_r\}$, a message data τ (computed on f), and an authentication tag σ, the verification algorithm outputs 0 (reject) or 1 (accept).

III. Protocol Construction

We propose a privacy preserving path authentication protocol. We assume the supply chain path of a certain product is pre-determined (static) by the manufacturer. Each tag $\mathcal{T}_i$ conveys its identity information (a 1-degree polynomial), a path code f (gate sequence of the arithmetic circuit). We employ a homomorphic message authentication code (HomMAC) with labelled program and a one-way PRF scheme as building blocks of the protocol.

A. Path Authentication Protocol

Consider a real-life scenario where a tag-enabled product traverses an automated supply chain, the tag is scanned at multiple locations: the manufacturer, logistics carrier, distribution centers, wholesalers and retailers etc. Assume a supply chain path authentication system consists of a manufacturer $\mathcal{M}$, a set of n tags $\mathcal{T}$, a set of d checkpoints $\mathcal{D}$, and a set of r intermediate readers $\mathcal{R}$. Readers in the supply chain are *semi-honest*, independent and have no knowledge about the path P. More clearly, a reader $\mathcal{R}_i$ in a valid path P_v follows protocol transaction correctly on tags. For building construction, we adapt the practical HomMAC described in [7] but customized to work with our path authentication scheme. Security of the scheme relies on the security of one-way function (PRF).

We divide our path authentication protocol in three steps: Initial setup, Tag evaluation, Verification. Initially, $\mathcal{M}$ sets up the whole system and stores the necessary protocol data into the tag, checkpoints and intermediate readers. Tags then get into the supply chain system and proceed towards the intended path. However, a tag would update its status as it comes across a new reader during its journey towards the destination checkpoint. Finally, the tag's evaluated data would be justified by the checkpoint in order to validate a certain path.

Initial Setup: $\mathcal{M}$ first chooses a PRF $\mathcal{F}_K : \{0,1\}^* \to \mathbb{Z}_p$ where K is the seed of $\mathcal{F}$ and p, a λ-bit prime number. Then $\mathcal{M}$ runs $\mathsf{KGen}(1^\lambda)$ and outputs $(\mathsf{sk}, \mathsf{ek}) = (\{K, s\}, p)$ where $s \in \mathbb{Z}_p$. $\mathcal{M}$ stores sk to the readers and checkpoint and ek to the tag.

We consider all the entities (e.g., $\mathcal{T}$, $\mathcal{R}$) possess unique ID or label $\iota_i \in \{0,1\}^\lambda$. The supply chain path from the manufacturer to the checkpoint is defined by $(\iota_0, \cdots \iota_r)$ where ι_0 is the tag's ID and $(\iota_1, \cdots \iota_r)$ are the IDs of intermediate readers $(1, \cdots, r)$ and an arithmetic circuit $f : \mathbb{Z}_p^r \to \mathbb{Z}_p$.

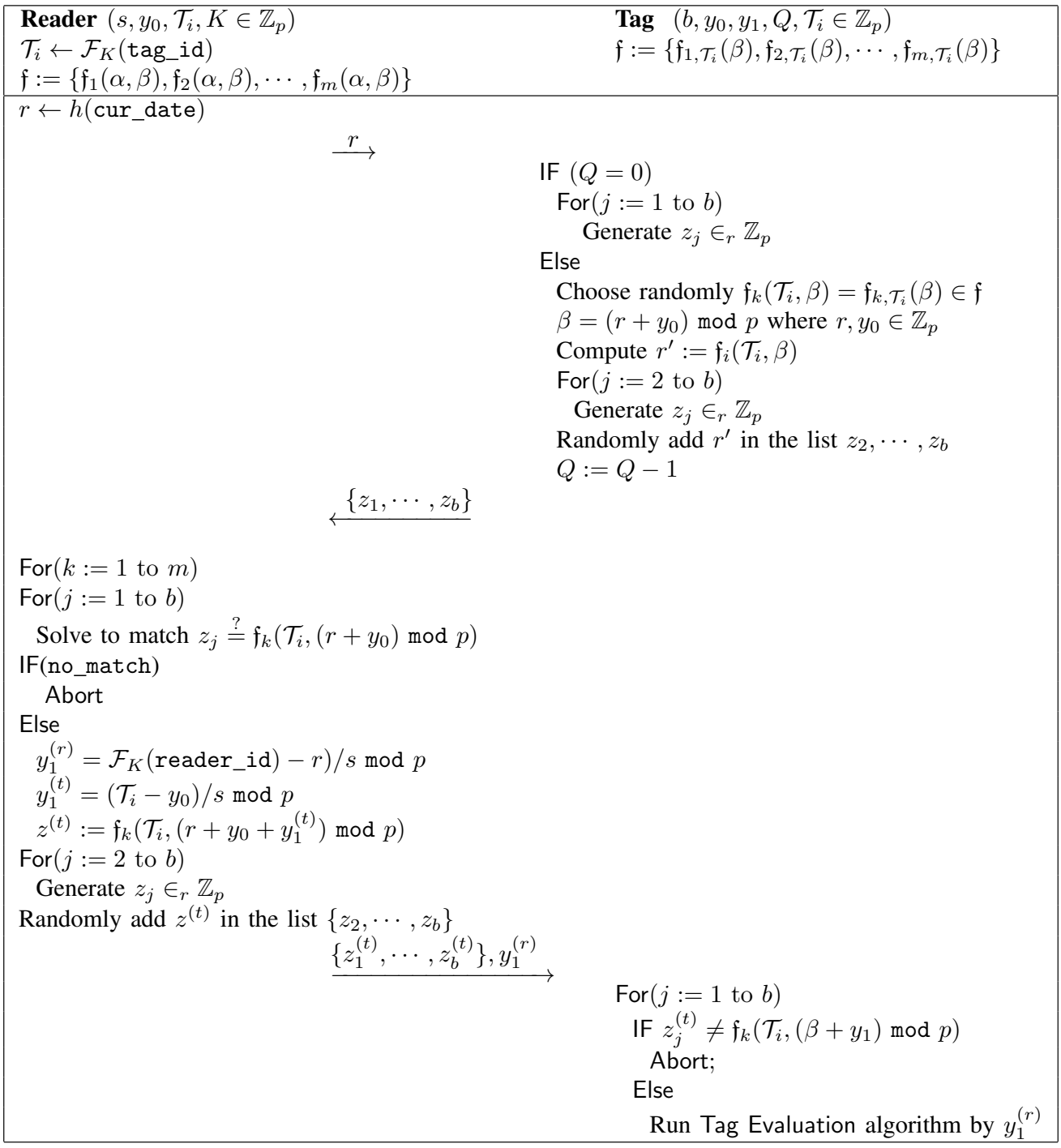

Fig. 2. Integrating Tag Authentication

Modern efficient inventory control policy includes exact knowledge of the flow of products: the amount of inventory at each location, predicted arrival date of an item etc. We address the issues in our solution. Let `reader_data`, `tag_data` be a certain reader and tag's meta data respectively. For instance, `reader_data` may include information about the expected arrival date, location etc., while tag data includes manufacture date, description of the product etc.

$\mathcal{M}$ defines a secure hash function $h : \{0,1\}^* \rightarrow \mathbb{Z}_p$ that converts any meta data to $\mathbb{Z}_p$. $\mathcal{M}$ provides h to $\mathcal{R}$ and store $h(\texttt{tag\_data})$ in $\mathcal{T}$. However, $h(\texttt{reader\_data})$ will be used as a *nonce* (during mutual authentication). In addition, both $h(\texttt{reader\_data})$ and $h(\texttt{tag\_data})$ will be used as a constant part (y_0) of the polynomial $y(z)$.

Every entity in the system (tags, readers) will be represented by a 1-degree polynomial $y(z) = y_0 + y_1 z, \;\; y \in \mathbb{Z}_p[z]$ where $y_0 = h(\texttt{tag\_data})$ or $h(\texttt{reader\_data})$, $y_1 = (\mathcal{F}_K(\iota) - y_0)/s$ mod p and outputs coefficients of the polynomial $y(z)$, that is, $\sigma = (y_0, y_1)$.

For each $\mathcal{T}_i$, $\mathcal{M}$ generates $y^{i0}(z) = \sum_j y_j^{i0} z^j$ where $\sigma_{i0} = (y_0^{i0}, y_1^{i0})$ and sets initial state $st_0 := \sigma_{i0}$ of the path. $\mathcal{M}$ sets secrets an evaluation key $\mathbf{ek}$ and a path code f where

$\mathsf{size}(f) = |P|$. $\mathcal{M}$ computes $\tau \leftarrow f(y_0^{i0}, \cdots, y_0^{ir})$ and shares τ with the checkpoint $\mathcal{D}$.

Tag Evaluation: As the product moves through the path P, $\mathcal{T}_i$ updates the path state st_j. When a tag $\mathcal{T}_i$ reaches $\mathcal{R}_j$, $\mathcal{R}_j$ runs $\mathsf{Authentication}(\mathsf{sk}, \iota, \tau)$ algorithm to compute $\sigma_{ij} = (y_0^{ij}, y_1^{ij})$ and forwards σ_{ij} to the tag $\mathcal{T}_i$ to update current state st_j.

Upon receiving σ_{ij} from $\mathcal{R}_j$, $\mathcal{T}_i$ runs the $\mathsf{Evaluate}(\mathsf{ek}, f, \sigma)$ algorithm and evaluates the existing circuit f on $\{\sigma_{i(j-1)}, \sigma_{ij}\}$ according to the current secret gate:

- If current gate is '+': $\mathcal{T}_i$ evaluates the new polynomial $y(z) = y^{j-1}(z) + y^j(z)$. Let d^j be the maximum degree of a polynomial $y^{j-1}(z)$, then coefficient of $y(z)$ will be $\sigma_{ij} \leftarrow (y_0^j, \cdots, y_d^j)$ where $d = \mathtt{max}(d^j, d^{j-1})$. Since $y^j(z)$ is always 1-degree polynomial, it is obvious that $d^{j-1} \geq d^j$. Note that the degree of $y(z)$ remain fixed after evaluating addition gate.
- If the current gate is '$\times$': $\mathcal{T}_i$ evaluates new polynomial $y(z) = y^{j-1}(z) \times y^j(z)$ and determines the coefficients of $y(z)$ as $\sigma_{ij} \leftarrow (y_0^j, \cdots, y_d^j)$ where $d = d^j + d^{j-1}$. Note that the degree of $y(z)$ increases by 1 after evaluating multiplication gate.

Finally, $\mathcal{T}_i$ stores $\sigma_{ij} := (y_0^j, \cdots, y_d^j)$ as the current state st_j.

Verification at the Checkpoint: $\mathcal{T}_i$ arrives at the destination checkpoint $\mathcal{D}$ with $st_r = \sigma_{i,r} = (y_0, \cdots, y_d)$. Now $\mathcal{D}$ verifies whether $\mathcal{T}_i$ has followed a valid path P_v by using $\mathsf{Verify}(\mathsf{sk}, \tau, \mathcal{P}, \sigma)$ algorithm. We consider two variants of verification process. First where $\mathcal{D}$ knows the path traversed $(\iota_1, \cdots, \iota_r)$ by a tag $\mathcal{T}_i$. Alternatively, where $\mathcal{D}$ has no knowledge of the path P_v (due to strict privacy).

Case-1: When $\mathcal{D}$ knows the valid path P_v of a tag $\mathcal{T}_i$:

- Check $y_0 \stackrel{?}{=} \tau$. If it outputs 1 (success), go to the next step.
- For every $\iota_i \in \mathcal{I}$, compute $\eta_i = \mathcal{F}_K(\iota_i)$
- Evaluate the circuit $f = \{o_1, o_2, \cdots, o_r\}$ on $\eta_0, \cdots, \eta_r$ s.t., $\Lambda = f(\eta_0, \cdots, \eta_r)$
- Evaluate the equation on $\sigma_{i,r}$ and check whether the following holds:

$$\Lambda \stackrel{?}{=} \sum_{\ell=0}^{d} \quad y_\ell s^\ell$$

Output 1 (accept) if true, else output 0 (reject).

Case-2: If $\mathcal{D}$ has no knowledge of the path P_v of a tag $\mathcal{T}_i$, $\mathcal{M}$ does not need to share $\mathcal{P} \leftarrow (f, \mathcal{I})$, PRF $\mathcal{F}$, and K, instead it shares Λ with $\mathcal{D}$. Then the verification algorithm will look like $\mathsf{Verify}(\mathbf{sk}, \tau, \Lambda, \sigma)$ where $\mathsf{sk} = \{s\}$. and proceeds as follows:

- Check $y_0 \stackrel{?}{=} \tau$. If it outputs 1 (success), go to the next step.
- Evaluate the equation on $\sigma_{i,r}$ and check whether the following holds:

$$\Lambda \stackrel{?}{=} \sum_{\ell=0}^{d} \quad y_\ell s^\ell$$

Output 1 (accept) if true, else output 0 (reject).

B. An Example

We illustrate our homomorphic path authentication system with a small and simple example. Suppose the manufacturer $\mathcal{M}$ initializes a tag T and a path with 3 intermediate readers $R = (R_1, R_2, R_3)$ with system parameters $p = 23$, secret $x = 4$. For simplicity, let $h(\mathtt{tag\_data})$ be 1, $h(\mathtt{reader\_data})$ of (R_1, R_2, R_3) be $(2, 3, 4)$, unique identifier labels of (T,R) and corresponding PRF output be $(\iota_0, \iota_1, \iota_2, \iota_3)$ and $(5, 10, 19, 12)$ respectively. Now we can construct 1-degree polynomials with coefficient $\sigma \leftarrow (y_0, y_1)$ for (T,R) according to the following:

- Tag T: $y^0(z) = 1 + ((5-1)/4 \bmod 23)\ z = 1 + z$ s.t., $\sigma_0 = (1, 1)$
- Reader R_1: $y^1(z) = 2 + ((10-2)/4 \bmod 23)\ z = 2 + 2z$ s.t., $\sigma_1 = (2, 2)$
- Reader R_2: $y^2(z) = 3 + ((19-3)/4 \bmod 23)\ z = 3 + 4z$ s.t., $\sigma_2 = (3, 4)$
- Reader R_3: $y^3(z) = 4 + ((12-4)/4 \bmod 23)\ z = 4 + 2z$ s.t., $\sigma_3 = (4, 2)$

Let T possess a secret path code $f :=$ ' $\times$ ++' or '100'. $\mathcal{M}$ computes $\tau (= 1 \times 2 + 3 + 4 = 9)$ by using the circuit f and shares τ with checkpoint D. As T moves through the valid path, it executes evaluation algorithm on f. Evaluation proceeds *gate-by-gate* as follows.

- On arrival R_1, for gate '$\times$': $y^{01}(z) = y^0(z) \times y^1(z) = 2 + 4z + 2z^2$
- On arrival R_2, for gate '+': $y^{012}(z) = y^{01}(z) + y^2(z) = 5 + 8z + 2z^2$
- On arrival R_3, for gate '+': $y^{0123}(z) = y^{012}(z) + y^3(z) = 9 + 10z + 2z^2$

As T arrives at checkpoint D, it first checks $y_0 \stackrel{?}{=} \tau (= 9)$. Then it computes $\rho = f(5, 10, 19, 12) = 5 \times 10 + 19 + 12 = 81$ (by using $\mathcal{F}$ and identifiers $(\iota_0, \iota_1, \iota_2, \iota_3)$) and checks whether the following equation holds:

$$\rho \stackrel{?}{=} \sum_{k=0}^{2} y_k x^k = 9 + 10.4 + 2.4^2 = 81 \text{ (for } 9 + 10z + 2z^2)$$

C. Integrating Mutual Authentication

Path authentication protocol cannot resist desynchronization, tag impersonation, or replay attack without mutual authentication. For instance, it is sufficient for an adversary to capture a protocol message from an honest reader and later replay it to the tag with counterfeit message to update current path state st. To address the above-mentioned attacks, we propose to extend our path protocol with a mutual authentication protocol in Fig. 2. We adopt polynomial-based authentication protocol described in [9] with major modifications (e.g., mutual authentication).

Unlike [9], we use two tag parameters $(\mathcal{T}_i, y_0)$ as secret, 1-degree bivariate set of polynomials $\mathfrak{f}$, no hash function in the tag, only b random numbers between the tag and reader. Reader initiates the protocol with $h(\mathtt{reader\_data})$, the tag follows the protocol transcripts in [9]. Upon receiving the feedback, the reader authenticates the tag and forwards $y_1^{(r)}$ (to update path status) and $z_i^{(t)}$ (to authenticate the reader) to the tag. Note that the tag would update current status st_j only if it can authenticate the reader successfully.

D. Batch Initialization

In [4], authors introduce path verification of a *batch of tags* that share the same path. However, we can accommodate the same construct in our protocol. Let a supply chain enrol a batch of n tags where each tag $\mathcal{T}_i$ is represented by a 1-degree polynomial $y^{(i)}(z) = y_0 + y_1^{(1)}z$. Since all the tags convey same meta data, they share same y_0. After initializing the batch of tags, $\mathcal{M}$ evaluates the circuit on polynomials $y^{(i)}(z),\ 1 \leq i \leq n$ by using $\mathsf{Evaluate}(\mathsf{ek}, f_b, \sigma_b)$ algorithm, where $|f_b| = n-1$ and $\sigma_b = \{\sigma_1, \cdots, \sigma_n\}$. Then it initializes the batch of $\mathcal{T}_i$ with the evaluated polynomial and releases it into the system. Meanwhile $\mathcal{M}$ shares necessary information (f_b, σ_b) with the checkpoints $\mathcal{D}$ for verification.

IV. Security Analysis

Correctness: An authentication tag σ can correctly authenticate a message τ under a set of label identifiers ι if

$$\Pr\Big[\mathsf{Verify}(\mathbf{sk}, \tau, \mathcal{P}, \sigma) = \mathsf{accept} \ \mid (\mathbf{ek}, \mathbf{sk}) \leftarrow \mathsf{KGen}(1^\lambda), \sigma \leftarrow \mathsf{Authentication}(\mathbf{sk}, \iota, \tau)\Big] = 1$$

where $\mathcal{P}$ is the identity program on a label $\iota \in \mathcal{I}$ with circuit f.

Our scheme consider a special 1-degree polynomial for a certain tag $\mathcal{T}_i$ s.t., $y^0(z) = y_0^0 + y_1^0 z$ where $y^0(0) = \tau$ and $y^0(x) = \eta_0 = \mathcal{F}_K(\iota_0)$. To preserve homomorphic property this is also followed by the intermediate readers $\mathcal{R}_j$ for evaluating the circuit $f : \mathbb{Z}_p^r \to \mathbb{Z}_p$ over $y^1, \cdots, y^r$. If a set of r triples $\{\tau_i, \mathcal{P}_i, \sigma_i\}$ such that $\mathsf{Verify}(\mathbf{sk}, \{\tau_i, \mathcal{P}_i, \sigma_i\}) = \mathsf{accept}$ then

$$\Pr\Big[\mathsf{Verify}(\mathbf{sk}, \tau^*, \mathcal{P}^*, \sigma^*) = \mathsf{accept} \ \mid \tau^* = f(\tau_1, \cdots, \tau_r), \mathcal{P}^* = f(\mathcal{P}_1, \cdots, \mathcal{P}_r), \sigma^* = \mathsf{Evaluate}(\mathbf{ek}, f, (\sigma_1, \cdots, \sigma_r))\Big] = 1$$

This definition briefly explains the correctness of the evaluation over the data.

Succinctness: The size of authentication tag $\mathtt{size}(\sigma)$ is bounded by a fixed $\mathtt{poly}(\lambda)$, where λ is a security parameter, irrespective of the input size of the arithmetic circuit f.

Polynomial Reconstruction Problem: Security of the authentication scheme described in [9] is based on the hardness of the well-known *Noisy Polynomial Interpolation Problem*(NPI) [15]. Authors consider *query and recovery* attack where the adversary queries the tag in order to recover the polynomial assigned to the tag. Because of the difficulty of *query and recovery* attack can be realized by the difficulty of the NPI problem. We refer to [9] for necessary definitions. Note that we slightly modify the existing protocol to reduce communication and computational overhead of the protocol. Moreover, our modified version is more secure, but requires more parameters to share between the reader and tag.

In order to respond to the challenge r, the tag evaluates a univariate polynomial $r' = \mathfrak{f}_{\mathcal{T}_i}(r + y_0)$. Since y_0 is a shared secret between the tag and reader, $y_0 + r$ can be considered as random to the adversary. In addition, using *secure hash* causes $h(\mathtt{reader\_data})$ to be considered as random even if the adversary knows $\mathtt{reader\_data}$. This r' is forwarded along with extra $b-1$ random elements. In every consecutive m queries $(m < Q)$ by the adversary, the tag employs all of its polynomial $\mathfrak{f}_i, 1 \leq i \leq m$ one after another, but in random manner.

Theorem 1. *Let $\mathcal{A}$ be a (Q, t, ϵ)-PPT adversary that can query a tag Q times $(m < Q$. Then the probability that $\mathcal{A}$ can successfully recover any polynomial of a tag in time t is*

$$\Pr[Adv_{\mathcal{A}}^{NPI(Q/m, mb-m+1, 1)}] \leq \epsilon.$$

Maximum number of queries allowed for a tag Q_{max} is

$$Q_{max} \approx ((b-1)m^2 + m).$$

Proof: We assume the maximum degree of α and β in a polynomial $\mathfrak{f}_i$ is 1 $(k = 1)$. We refer to the polynomial based authentication paper in [9] for detail proof.

Theorem 2. *The homomorphic authentication based protocol described in this paper is secure if and only if the Pseudo Random Function (PRF) $\mathcal{F}$ is secure.*
Proof: Proof of this will be given in the full version of the paper.

Proposition 1. *Let a PPT adversary $\mathcal{A}$ has compromised n tags targeting to recover $\mathfrak{f} := \mathfrak{f}_1, \cdots, \mathfrak{f}_m$. Hence the probability that $\mathcal{A}$ can compute $\mathfrak{f}_i$ is*:

$$\Pr[Adv_{\mathcal{A}}] \leq m^{-k-2}.$$

where k is the maximum degree of $\mathfrak{f}_i$ and $n \geq (k+1)^2/k$
Proof: deferred to the appendix part.

A. Privacy

In order to define privacy, we analyzed our protocol according to the *path-privacy* framework in [3] where the privacy of *tag identity* (tag unlinkability) and *path information*(step unlinkability) are formulated together in a single game. Our privacy notion is quite similar to the one proposed in [3], except for some minor modifications. First, we explicitly allow the adversary to query readers during Move operation. Second, unlike path authentication scheme in [3], our state update operation takes place inside the tag with some secrets, such as, coefficient of the tag's polynomial σ_0 and circuit information f.

Let $\mathcal{A}$ be a PPT adversary against RFID path authentication that takes as input the system public parameters, a set of readers $\mathcal{R}$, a set of tags $\mathcal{T}$, and a set of checkpoints $\mathcal{D}$. $\mathcal{A}$ has access to the following oracles $\mathtt{Read\_frm\_R}(\mathcal{R}_i)$, $\mathtt{Eval\_to\_T}(\mathcal{R}_i, \mathcal{T}_j), \mathtt{Path\_Verify}(st_{\mathcal{T}_j})$, $\mathtt{Move}(\mathcal{T}_j, k, \mathcal{K}, b)$, where $1 \leq k < |\mathcal{P}|$ for a certain path $\mathcal{P},\ \mathcal{K} \in \{\mathcal{P}, G\}, b \in \{0, 1\}$.

Let $\mathbf{Exp}_{\mathcal{A}}^{\mathtt{Path-Privacy}}[\lambda]$ be a path-privacy experiment that initializes the system $(\mathcal{M}, \mathcal{R}, \mathcal{D}, \mathcal{T})$ through $\mathsf{Setup}(\lambda)$. Adversary $\mathcal{A}$ consists of two algorithms, namely $\mathcal{A}_1$ and $\mathcal{A}_2$. We redefine generic oracles according to the following:

- Read_frm_R($\mathcal{R}_i$): This oracle returns identity information of a reader $\mathcal{R}_i$ to a tag $\mathcal{T}_j$. We assume that the readers along the path are honest, that is, they will send protocol transcript only if the tag is authenticated.
- Eval_to_T($\mathcal{T}_j,$): On input tag-reader references $\mathcal{T}_j, \mathcal{R}_i$, this oracle evaluates the internal state $st_{\mathcal{T}_j}$ of a tag $\mathcal{T}_j$. We assume the tags to be honest, i.e., they follow protocol transcripts.
- Path_Verify($st_{\mathcal{T}_j}$): On input state information st, this oracle verifies whether $\mathcal{T}_j$ has followed the valid path $\mathcal{P}_v$ and outputs 1 (successful). Otherwise it returns ϕ (fail).
- Move($\mathcal{T}_j, k, \mathcal{K}, b$): If $\mathcal{K} = G$, $\mathcal{T}_j$ evaluates the current state st, k times as it moves arbitrarily in the directed acyclic graph G irrespective of the value of b. However, if ($\mathcal{K} = \mathcal{P}$ and $b = 1$), it evaluates the current st along the valid path $\mathcal{P}_v$ in the supply chain k steps that outputs a new state $st_{\mathcal{T}_j}$. However, if $b = 0$, move the tag k steps arbitrarily to any path $\mathcal{P}'$ such that $\mathcal{P}' \cap \mathcal{P} = \varnothing$. The tag $\mathcal{T}_j$'s state is evaluated in each step of the path. Consequently, it returns the state transcript $st_{\mathcal{T}_j}$.

On input of public parameters as mentioned in Fig.1, a probabilistic polynomial time (PPT) algorithm $\mathcal{A}$, denoted by $\mathcal{A}^{\texttt{Read\_frm\_R},\ \texttt{Eval\_to\_T},\ \texttt{Path\_Verify},\ \texttt{Move}}(\lambda)$, runs a supply chain system via the above-mentioned oracles.

In the learning phase, a PPT adversary $\mathcal{A}_1$ queries the four oracles at certain times and outputs two tags $\mathcal{T}_0, \mathcal{T}_1$, a path $\mathcal{P}$ that has at least k readers to reach a checkpoint $\mathcal{D}$ for both tags, and the tag's internal state information st. In the challenge phase, after tossing a coin, $\mathbf{Exp}_{\mathcal{A}}^{\texttt{Path-Privacy}}$ chooses either $\mathcal{T}_0$ or $\mathcal{T}_1$ and moves through k readers remaining along the path and updates the internal state st. Let $\mathcal{T}_0$ reach its last state st_0 by following valid path. Alternatively, $\mathcal{T}_1$ reaches its last state st_1 without following the path. Although $\mathcal{A}_1$ has access to the readers, it has no access to the tag during the Move operations. In the challenge phase, the experiment $\mathbf{Exp}_{\mathcal{A}}^{\texttt{Path-Privacy}}$ provides $\mathcal{A}_2$ with last state of $\mathcal{T}_i$, that is, st_i and previous state information st. Then, $\mathcal{A}_2$ guesses $\mathcal{T}_i$. The experiment outputs 1, and hence, the adversary wins the game if $\mathcal{A}_2$ can guess $\mathcal{T}_i$ correctly with a probability more than $1/2$.

Experiment $\mathbf{Exp}_{\mathcal{A}}^{\texttt{Path-Privacy}}[\lambda]$

1) Run Setup(λ) to set $\mathcal{M}, \mathcal{R}, \mathcal{T}, \mathcal{D}$.
2) $\{\mathcal{T}_0, \mathcal{T}_1, \mathcal{P}, k, st\} \leftarrow \mathcal{A}_1^{\texttt{Read\_frm\_R},\ \texttt{Eval\_to\_T},\ \texttt{Path\_Verify},\ \texttt{Move}}$ where $|\mathcal{P}| \geq k \geq 1$ and st is current state information.
3) $b \leftarrow \{0, 1\}$.
4) $st_{\mathcal{T}_b} \leftarrow$ Move($\mathcal{T}_b, k, \mathcal{P}, b$). $st_{\mathcal{T}_b}$ represents the state of $\mathcal{T}_b$.
5) $b' \leftarrow \mathcal{A}_2^{\texttt{Read\_frm\_R},\ \texttt{Eval\_to\_T},\ \texttt{Path\_Verify},\ \texttt{Move}}(st_{\mathcal{T}_b}, st)$.
6) Output 1 if $b' = b$, or 0 otherwise.

Advantage of $\mathcal{A}$, denoted $Adv_{\mathcal{A}}^{\texttt{Path-Privacy}}(\lambda)$, in the path privacy experiment is $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\texttt{Path-Privacy}}[\lambda] = 1] - \frac{1}{2}|$

Theorem 3. *If PRF is secure and pseudorandom, then our path authentication protocol is private under the semantic security of Homomorphic MAC scheme.*

Proof. Proof is deferred to the full version of the paper.

TABLE I
COMPARISON AMONG PATH AUTHENTICATION PROTOCOLS

	Ours	ACISP'09
Authentication	Mutual	Tag
Shared secret	2	1
Tag storage	$2m$	$m \cdot (k+1)$
Random numbers generated by Tag	$2b-1$	$b-1$
Tag Computation	$2\mathfrak{f}(.)$	$1\mathsf{H}, \mathfrak{f}(.)$
Communication cost	$b+3$	$2b+1$

V. PERFORMANCE EVALUATION

Our scheme is secure and highly efficient, especially if we ignore system initialization process and the computations that can be pre-processed offline. Since this scheme offers a more practical and rigorous security assumption (e.g., adversary having access to the Readers during the Move operation, a polynomial based tag authentication scheme with the same parameter used in path authentication), we consider the tag to perform some lightweight computation at each step. All the major computations are performed by the manufacturer and Checkpoint verifier.

The cost of Tag evaluation depends on the size and gate types of the circuit f. Nevertheless, the evaluated polynomial inside the tag grows with a degree d, finally yields an overhead of $O(d)$ for addition gate and $O(d \log d)$ (using FFT)for multiplication gate. Yet the succinctness of the evaluated polynomial can be assured while $d < |f|$.

In each step of the path, a tag evaluates either an addition or a multiplication with a 1-degree polynomial (received from the Reader). *Addition* operation can be done simply by adding two vectors of coefficient. A polynomial of degree d has $d+1$ coefficients. So simple addition (with a 1-degree polynomial) requires only $d \geq 1$ addition, while *Multiplication* operation use the convolution operator $'*'$ that requires $2(d+1)$ multiplication and $d-1$ addition operation, resulting in, $3d$ operations in total. However, for very large d the number of operations can be reduced to $O(d \log d)$ using FFT. Initially, the manufacturer stores 2 secret items in $\mathbb{Z}_p$ (2 coefficient of a 1-degree polynomial) into the tag. Subsequently the tag evaluates the existing polynomial recursively as it moves. Note that the addition gates will not increase the value of d while each multiplication gate increases the value of d by 1. If we consider a 32-bit long prime, then initially a tag requires 64-bits, that grows upto $32d+1$ bits (tag requires to store $d+1$ items for a polynomial of degree d). For simplicity, we allow a maximum of 8 multiplication gates arbitrarily in f, which yields a 257-bit tag storage.

On the other hand, the maximum cost of verification in the checkpoint includes the cost of computing $\Lambda = f(\eta_0, \cdots, \eta_r)$, clearly $O(|f|)$ and $\sum_{l=0}^{d} y_l s^l$, that is $O(d)$. Note that in Case-2 (without Path − info) of the verification algorithm does not require the calculation of Λ since it is pre-shared between the manufacturer and Checkpoint.

We propose a polynomial-based protocol described in [9] with some modifications for authenticating the tag (optional) at each step of the supply chain. At each step, a tag needs

TABLE II
COMPARISON AMONG PATH AUTHENTICATION PROTOCOLS

	Ours	ACNS′12 [3]	NDSS′11 [2]	SEC′12 [4]	RFIDSec′09 [6]
Path generation	static	static	static	dynamic	dynamic
Building blocks	HomMAC	ECElGamal + PRF	ECElGamal	OMS	PRF
Privacy	PUlink$^{\ddagger\ \dagger}$	PUlink†	TUlink + SUlink†	TUlink + PUlink	NG
Path evaluation	Tag	Reader	Reader	Reader	Tag
Mutual authentication	Yes	No	No	No	Yes
Tag storage	257* bits	480 bits	960 bits	720 bits	NG
Reader storage	$O(1)$	$O(1)$	$O(1)$	$O(n)$	O(N)
Checkpoint storage	$O(N)$	$O(N)$	$O(N + vP)$	-	-
Tag computation	PolyA or PolyM	–	–	3H	3H
Reader computation	1H	2ECM, 2ECA, 1PRF	10ECM, 3ECA	1DEC, 3P, 4EX, 6M	1PRF, 1OWF, 1H

HomMAC: Homomorphic MAC on Arithmetic Circuit, ECElGamal: Elliptic Curve ElGamal re-encryption, OMS: Ordered Multi-Signature scheme, PRF: Pseudo Random Function, NG: Not Given, TUlink: Tag Unlinkability, SUlink: Step Unlinkability, PUlink: Path unilinkability, N: Number of total tags, n:Size of batch of tags, vP: Number of valid paths, ECM: Elliptic Curve multiplication, ECA: Elliptic Curve addition, H: Hash function, OWF: Keyed One-Way Function, P: Pairing, EX: Exponentiation, M: Multiplication, PolyA: 1-degree Polynomial addition, PolyM: 1-degree Polynomial multiplication

‡Path unilinkability where adversary has access to the reader during `Move` operation.
†Privacy proof included.
*Considering $32d + 1$ s.t., max degree of a polynomial $d = 8$ with prime p (32-bit).

to generate $b - 1$ random numbers in $\mathbb{Z}_p$, evaluate a 1-degree polynomial over $\mathbb{Z}_p$. Moreover, the tag is required to store m 1-degree univariate polynomials randomly, that is, the tag needs to store $2m$ items in $\mathbb{Z}_p$. In addition, it takes only 1 modular addition and 1 modular multiplication (Horner's rule) over $\mathbb{Z}_p$ to evaluate the polynomial. It is fairly certain that the value of m ($m = 16$ in [9]) must be larger in our scheme to reach same security settings as that of [9]. We carefully observe that incrementing the value of m comparatively demands more space and computational cost in the reader, instead of the tag. However, if $N(= 2^{\lambda}$, s.t., $2^{\lambda}+1 \leq p)$ be the maximum number of tags in a supply chain, then a tag requires λ-bits ROM, corresponding to λ gates in hardware for each $\mathbb{Z}_p$ element. In addition, modular multiplier takes several hundred more hardware gates. Meanwhile, each reader needs to store m 1-degree bivariate polynomials, that is, it needs $4m$ items in $\mathbb{Z}_p$ to store. In the worst case, it needs to solve mb 1-degree polynomial over $\mathbb{Z}_p$.

VI. CONCLUSION

In this paper, we studied the existing RFID-enabled path authentication schemes for a supply chain management. We present a new direction for using an arithmetic circuit based Homomorphic MAC. In addition, we introduce a refined privacy notion, an appropriate but optional mutual authentication scheme, a potential batch initialization of the tags.

REFERENCES

[1] RFID Security and Privacy Lounge, www.avoine.net/rfid/
[2] E.O. Blass, K. Elkhiyaoui, and R. Molva. Tracker: Security and privacy for RFID-based supply chains. in NDSS, pp. 455472, 2011.
[3] Cai, Shaoying, Robert Deng, Yingjiu, Li, Yunlei, Zhao. A new framework for privacy of RFID path authentication. Applied Cryptography and Network Security. Springer Berlin Heidelberg, 2012.
[4] Cai, Shaoying, Yingjiu Li, Yunlei Zhao. Distributed Path Authentication for Dynamic RFID-Enabled Supply Chains. Information Security and Privacy Research. Springer Berlin Heidelberg, pp501-512, 2012.
[5] Wang, Hongbing, et al. Two-level path authentication in EPCglobal Network. RFID (RFID), 2012 IEEE International Conference on. IEEE, 2012.
[6] Ouafi, Khaled, and Serge Vaudenay. Pathchecker: an RFID Application for Tracing Products in Suply-Chains. Workshop on RFID SecurityRFIDSec. Vol. 9. 2009.
[7] Catalano, Dario, and Dario Fiore. Practical Homomorphic MACs for Arithmetic Circuits. Advances in CryptologyEUROCRYPT. Springer Berlin Heidelberg, 336-352, 2013.
[8] Gennaro, R., Wichs, D. Fully homomorphic message authenticators. Cryptology ePrint Archive, Report 2012/290 (2012), http://eprint.iacr.org
[9] Wu, Jiang, and Douglas R. Stinson. A highly scalable RFID authentication protocol. ACISP 2009. Springer Berlin Heidelberg, 2009.
[10] MSI Mamun, A. Miyaji, M. Rahman. A Secure and Private RFID Authentication Protocol under SLPN Problem. NSS 2012, LNCS 7645, pp. 476-489, 2012.
[11] Katz, Jonathan, Ji Sun Shin, and Adam Smith. Parallel and concurrent security of the HB and HB+ protocols. Journal of cryptology 23, no. 3 (2010): 402-421.
[12] MSI Mamun, A. Miyaji. A fully-secure RFID authentication protocol from exact LPN assumption, IEEE TrustCom'13, page 102-109, DOI: 10.1109/TrustCom.2013.17
[13] MSI Mamun, A. Miyaji. A privacy-preserving efficient RFID authentication protocol from SLPN assumption. International Journal of Computational Science and Engineering (IJCSE), Inderscience Publishers, Vol. 9, 2014.
[14] Cai, Shaoying, et al. Protecting and restraining the third party in RFID-enabled 3PL supply chains. Information Systems Security. LNCS, 246-260, 2011.
[15] Naor, Moni, and Benny Pinkas. Oblivious polynomial evaluation. SIAM Journal on Computing 35.5: 1254-1281, 2006.
[16] Shpilka, A., Yehudayo, A. Arithmetic circuits: A survey of recent results and open questions. Foundations and Trends in Theoretical Computer Science 5(3-4), 207388, 2010.