Computational Intelligence, Volume 59, Number 000, 2010

SupAUTH: A New Approach to Supply Chain Authentication for the IoT

MOHAMMAD SAIFUL ISLAM MAMUN & ALI A. GHORBANI
Canadian Institute of Cybersecurity, University of New Brunswick, Fredericton, NB, Canada

ATSUKO MIYAJI
Cyber Security Engineering, Graduate School of Engineering, Osaka University, Japan

UYEN TRANG NGUYEN
Department of Computer Science, York University, Toronto, Ontario, Canada

Recent advances of the Internet of Things (IoT) technologies have enhanced the use of RFID based tracking system to be widely deployed in supply chain management covering every steps involved in the flow of merchandise from the supplier to the customer in order to ensure a trustworthy delivery environment. Such authentication system (aka path authentication) not only guarantees the merchandise to be available in the right destination with no discrepancies and errors, but also ensures the route of the merchandise progress to be valid. This paper outlines the current state-of-the-art cryptographic solutions for path authentication, highlights their properties and weakness, and propose a novel, privacy-preserving, and efficient solution.

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. Moreover, we allow computational ability inside the tag that articulates a new privacy direction to the state-of-the-art *path privacy*. This privacy notion helps support confidentiality of the tag movement in the context of IoT-enabled cross-organizational tracking environment where the stakeholders can be from different organizations associated together with the merchandise being delivered. As a potential extension to the path authentication protocol, we further propose a polynomial-based mutual authentication as a security extension and batch initialization as an efficiency extension.

Besides our brief security and privacy analysis, our evaluation shows that the proposed solution can significantly reduce memory requirements on tags with marginal computational overhead to ensure transmission path confidentiality. We observe that SupAUTH requires maximum 513-bit tag memory and 57.3 $m.s$ processing time during evaluation which is not only practical but also suitable for any suitable low-cost RFID deployment in IoT.

Key words: Supply Chain Management, Path Authentication, Arithmetic Circuit, Homomorphic MAC.

1. 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, efficiency, and gain in productivity. With the growing nature of SCM, it is crucial to construct protocols that enable end-users to verify the security and privacy not only of the RFID tags but also the path they go through. This makes the supply chain to be substantially more accurate and improves the reliability of the entire chain.

Path authentication in SCM ensures genuine authorization to Tag-Reader interaction and hence the integrity of the supply chain life-cycle. More concisely, when a tag reaches the end of its supply chain, its *authentication tag* guarantees that no intermediate Reader was omitted or selected wrongly by the tag, either deliberately or not. The objective of this study is to review the current state-of-the art cryptographic techniques for tag path authentication and propose a new solution to improve efficiency and privacy.

A number of *tag authentication* schemes that have been proposed so far (*e.g.*, Katz and Shin (2006); Mamun et al. (2012); Mamun and Miyaji (2013, 2015)) cannot be used directly for path authentication, either because they incur high computational cost which is unsupportable in practice or they lack simultaneous online access to all parties in the supply chain (Cai et al. (2012)). That is in order for an existing solution to deploy in a supply chain, the destination *i.e.*, verification checkpoint must have access to all the database entities in the SCM. This may be unrealistic in real-life practice where the initiator *i.e.*, Manufacturer has only access to the underlying entities.

Moreover, security requirements of an RFID *ownership transfer* protocol (Ouafi and Vaudenay (2009); Mamun and Miyaji (2014); Moriyama (2013)) where ownership of a tag need to be transferred securely are very close to that of *path authentication* protocols (Blass and Elkhiyaoui (2011); Cai et al. (2012); Wang et al. (2016, 2012)). But, in comparison to ownership transfer protocol, path authentication protocol demands additional privacy requirements such as *forward privacy* where former executions must not be traced by the current owner and *backward privacy* where the succeeding transaction should not be traced by the former owner Cai et al. (2012).

Besides that, some recent works have been inspired by path authentication protocol Han et al. (2015); Chang et al. (2016), but they are not exactly path authentication scheme or are different from the objective of path authentication scheme. For example, authors in Chang et al. (2016) propose a key update scheme for secure train communication at Train stations. Although their work seems to be similar to path authentication in the sense that the relaying stations are stationary (resembles RFID Readers in path authentication) and interact with the moving trains (resembles RFID tags in the path authentication) for authentication, this scheme is fundamentally different. Because trains are routed through the railway tracks pre-established by the Operational Control Center(OCC). Unlike path authentication in a supply-chain, the travel path of a train has actually served as a source of assurance in their scheme and it is not easy to diverge a train from the railway tracks. Moreover, in Han et al. (2015), authors highlight confidentiality problem of transmission path (business data) in IoT-enabled tracking systems and propose a prototype for a fine grained confidentiality control mechanism.

There are two kinds of path authentication system, namely, *dynamic* path authentication, where the path is generated dynamically and every node in the path can track the validation of the path (*e.g.*, Cai et al. (2012)); and, *static* path authentication, where a valid *path* is predetermined and shared with the destination checkpoint (*e.g.*, Cai et al. (2012)).

Dynamic path authentication protocol was first proposed in Wang et al. (2012) based on Hierarchical Identity-Based Encryption (HIBE) using Boneh-Lynn-Shacham (BLS) digital signature scheme (Boneh et al. (2004)) where tag identity is encrypted by the HIBE and the path is generated by hashing all the *past identities* of the Reader with digital signature. Every next node throughout the path can validate the past node as well as the path a certain tag has passed. Some other solutions for the dynamic path authentication includes the schemes in Cai et al. (2012); Ouafi and Vaudenay (2009) based on Ordered-Multi-signature and Pseudo-Random-Function, respectively.

After the first proposal on *static* path authentication by Blass and Elkhiyaoui (2011), protocol construction and priveacy has been significantly improved by Cai et al. (2012) and later by Wang et al. (2016). All of them use EIGamal Re-encryption with symmetric key to encrypt the tag identity.

Meanwhile although some dynamic path authentication schemes (*e.g.*, Ouafi and Vaude-

nay (2009); Cai et al. (2012)) incorporate mutual authentication into their proposal, no prior *static* path based authentication schemes consider mutual authentication between the tag and intermediate Readers. Either they assume the communication channel between the Reader and tag during path authentication is *secure*, or they presume tag authentication implicitly. For instance, in Cai et al. (2012) assume that the Reader will update the tag's state only after successful authentication.

SupAUTH presents a new variant of static path authentication scheme aiming to modify the privacy model of the path authentication scheme in Cai et al. (2012) which is an extended and more practical privacy variant of Blass and Elkhiyaoui (2011). In comparison to this scheme, our contribution includes the following:

- We instantiate a new variant of path authentication scheme with arithmetic circuit based HomMAC. Note that building blocks of the previous static path based authentication systems were mainly from expensive elliptic curve ElGamal re-encryption (ECElGamal) and the security of the schemes was primarily either from the Pseudo Random Function (PRF) or the Homomorphic MAC (HomMAC). Security of SupAUTH also stems from the PRF.

- We propose that the state update operations to be held inside the tag. It offers more security with a 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 similar to other existing schemes, it is also manageable and potentially easier to update state information into the Readers, and hence no need for any computation inside the tag.

- By modifying privacy model proposed in Cai et al. (2012), our scheme achieves confidentiality requirement of sensitive business data. Note that Cai et al. (2012) introduces a new oracle Move that models a tag's movement along a designed or an arbitrary path in a supply chain. Adversary however is not allowed to query either the Reader or the tag during Move operation run by the game challenger. However, we consider a relaxed privacy assumption that allows adversary to query the Move oracle by redefining the generic privacy oracles of Cai et al. (2012) (in Section 4.1). That is why disallowing adversary to query only the tag is sufficient enough for the path privacy experiment to fail in our case. This assumption is more practical and formal. More clearly, in our scheme, an RFID tag collects Readers' information, but not vice versa and update its status along the path. Therefore, the Readers convey no information about the tag's current state during tag movement. This techniques helps support confidentiality of the tag movement from different stakeholders associated together with the merchandise being shipped.

- Unlike the static path authentication scheme in Cai et al. (2012), 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 Cai et al. (2012), our scheme requires less storage but poses conditional scalability such as maximum number of tags, arithmetic operation gates etc. (detail in Section 5).

- We propose a polynomial based mutual authentication scheme from Wu and Stinson (2009) 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 reducing communication, storage and computation overhead into the tag effectively.

- We show how to accommodate a batch of tags that must follow the same path to the destination.

The rest of this paper is organized as follows. Related works on supply chain management and cryptographic tools are briefly discussed in Section 2. Section 3 presents the main protocol employed. Section 4 describes the security and privacy analysis of the protocol. Section 5 discusses the potential extension to the proposed protocol SupAUTH with the corresponding security proof. Section 6 details the performance evaluation and comparison with the related works, while conclusions are given in Section 7.

2. BACKGROUND

2.1. Supply Chain Management & Path Authentication

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 chain are *linked* together through information flows that allow various supply chain partners to coordinate the day-to-day flow of products within a supply chain path. It can be represented as a directed acyclic graph (DAG). An RFID tag attached on every product in the supply chain contains a unique identifier about the product.

Directed Acyclic Graph: 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.

Path: A valid finite path $P = (v_0, \cdots, v_r)$ where v_0 is the entrance of a product to the supply chain and v_r is its final destination to arrive, is a licit sequence of readers/steps a valid tag needs to pass. A path is usually determined by the coordinator (*e.g.*, the manufacturer) of the supply chain.

Unlike Tag-only authentication, path authentication requires a valid destination/reader to accept only those tags that have been delivered to it through a valid *path*. Therefore, Tag-only authentication or mutual authentication could be a part of path authentication, not a replacement, since they cannot verify the path a tag would have passed.

RFID-enabled SCM: 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 $\mathcal{T}$. Manufacturer $\mathcal{M}$ 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 $\mathcal{M}$ to $\mathcal{D}$. Note that checkpoint is a special Reader or node where the validation of a tag and its path is finally verified. More precisely, the system has the following functionalities:

- **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 if this is the case. Otherwise it returns *false*.

2.2. **Cryptographic Tools**

Labelled program: The notion of labeled data or program was first introduced by Gennaro and Wichs (2013). Let an entity (*e.g.*, checkpoint) want to authenticate some data $\tau := \{\tau_0, \tau_1, \ldots, \tau_r\}$ (*e.g.*, tag/Reader's data) with respect to their corresponding labels $\mathcal{I} := \{\iota_0, \iota_1, \ldots, \iota_r\}$ (*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 $f : \{0,1\}^r \to \{0,1\}$ 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 $\tau := \tau_0, \tau_1, \ldots, \tau_r$ is a labelled directed acyclic graph G with its leaves labelled as $\mathcal{I} := \{\iota_0, \iota_1, \ldots \iota_r\}$ and internal nodes labelled as gate $\mathcal{O} := \{+, \times\}$ operations. The circuit has a designated output ρ.

In SupAUTH, we consider an arithmetic circuit f over a field $\mathbb{Z}_p$ such that $f : \mathbb{Z}_p^r \to \mathbb{Z}_p$ 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 the 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 a natural way and every polynomial defines a unique function. An input gate of an arithmetic circuit can compute a polynomial that is tagged by the labels. A sum gate '$+$' computes the summation of *two* polynomials obtained from the incoming wire in the graph. Similarly, a product gate '$\times$' computes the multiplication of *two* polynomials. (See Figure 1.)

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 the class P. That is, all polynomials of polynomially bounded degree can be realized by an arithmetic circuit family with polynomially bounded size Shpilka and Yehudayoff (2010).

Pseudorandom Function (PRF): A function $\mathcal{F} : \{0,1\}^* \to \mathbb{Z}_p$ is called a (t, ϵ, q)-PRF if for a given key K with a security parameter λ and an input $X \in \{0,1\}^*$ there is an *efficient* polynomial-time algorithm to compute $\mathcal{F}_K(X) = \mathcal{F}(X, K)$. A PPT adversary $\mathcal{A}(t, \epsilon)$ can break the PRF $\mathcal{F}$ if,

$$\Pr[\mathcal{A}^{\mathcal{F}_K}(\lambda)] = 1 \text{ - } \Pr[\mathcal{A}^{\mathcal{U}}(\lambda)] = 1 \geqslant \epsilon$$

where $\mathcal{U}$ is uniformly distributed over $\mathbb{Z}_p$ and $\mathcal{A}(\lambda)$ can make at most q query to the oracle

in running time t.

Homomorphic Authentication Scheme: In a homomorphic message authenticator scheme, an entity can authenticate data τ with its secret key sk. Later evaluators can homomorphically execute an arbitrary program $\mathcal{P}$ over τ and subsequently generate an authentication tag σ with the knowledge of another key ek (without knowing sk). Note that σ certifies $\mathcal{P}(\tau)$. Finally a verifier that knows sk can verify whether σ is indeed the output of the $\mathcal{P}(\tau)$ without knowing τ. A Homomorphic Message Authentication scheme consists of the following algorithms:

- $\mathsf{KGen}(1^\lambda)$: on input of the security parameter λ, it generates a key pair $(\mathsf{sk}, \mathsf{ek})$ where 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).

Moreover, a homomorphic message authenticator (HomMAC) scheme (Catalano and Fiore (2013)) must have the following properties:

- Succinctness: Authenticity of a labeled program $\mathcal{P}$ can be ensured with minimum communication cost. That is, the cost should be much less than that of sending the original inputs.
- Composability: If a tag σ is generated (as an output of $\mathcal{P}$) for authenticating former computations, it must be used as an input to authenticate further new computations.
- Security: An adversary must not be able to create any valid tag σ for messages τ that are not produced as the output of $\mathcal{P}$, provided that it can adaptively observe polynomially many tag-messages pairs of its own choice.

3. PROTOCOL CONSTRUCTION

We propose a privacy preserving path authentication protocol assuming the path to be 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.

3.1. 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

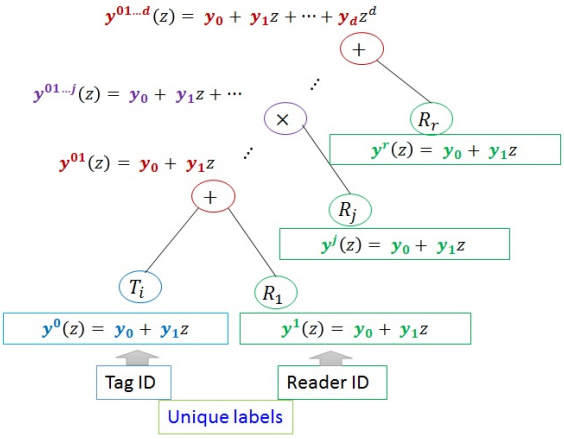

FIGURE 1. Arithmetic circuit & polynomial replacement

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 protocol construction, we adapt the practical HomMAC described in Catalano and Fiore (2013) but customized to work with our SupAUTH. Security of the scheme relies on the security of one-way function (PRF).

We divide our 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 (see Figure 4).

Initial setup: $\mathcal{M}$ first chooses a PRF $\mathcal{F}_K : \{0,1\}^* \rightarrow \mathbb{Z}_p$ 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 $s \in \mathbb{Z}_p$. Manufacturer $\mathcal{M}$ stores sk to the Readers and checkpoints 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 $f : \mathbb{Z}_p^r \rightarrow \mathbb{Z}_p$.

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 $y(z)$.

System parameter	$(\lambda, \mathcal{F}, h, K, s, p, f, \mathfrak{f}, b)$ p is a prime of λ bits, $s \in \mathbb{Z}_p$, $b, m \in \mathbb{N}(b, m > 1)$ Polynomial $\mathfrak{f} := \{\mathfrak{f}_1(\alpha, \beta), \mathfrak{f}_2(\alpha, \beta), \cdots, \mathfrak{f}_m(\alpha, \beta)\}$ Hash $h : \{0,1\}^\lambda \in \mathbb{Z}_p$ Pseudo Random Function $\mathcal{F}_K : \{0,1\}^\lambda \in \mathbb{Z}_p$ Arithmetic circuit f: $\mathbb{Z}_p^r \to \mathbb{Z}_p$, 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` $y_1 = (\mathcal{T}_i - y_0)/s \bmod p$ $Q :=$ `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 $\mathcal{T}_i$
Initialize Checkpoint $\mathcal{D}_k$ Without `Path-info` $(s, \tau, \Lambda, \sigma)$ With `Path-info` $(s, p, f, K, \tau, \mathcal{F}, P_v, \sigma)$	$\tau = f(y_0^0, \cdots, y_0^r)$ $\sigma = (y_0^{i,r}, \cdots, y_d^{i,r})$ (evaluated by $\mathcal{T}_i$ with Reader $\mathcal{R}_r$) $\Lambda = f(\eta_0, \cdots, \eta_r)$ s.t., $\eta_i = \mathcal{F}_K(\iota_i)$ where $\iota_0 =$ `Tag ID`, $\{\iota_1, \cdots, \iota_r\} =$ `Readers ID` Path $P_v \leftarrow \{\iota_0, \cdots, \iota_r\}$

FIGURE 2. SupAUTH Initialization and Public Parameters

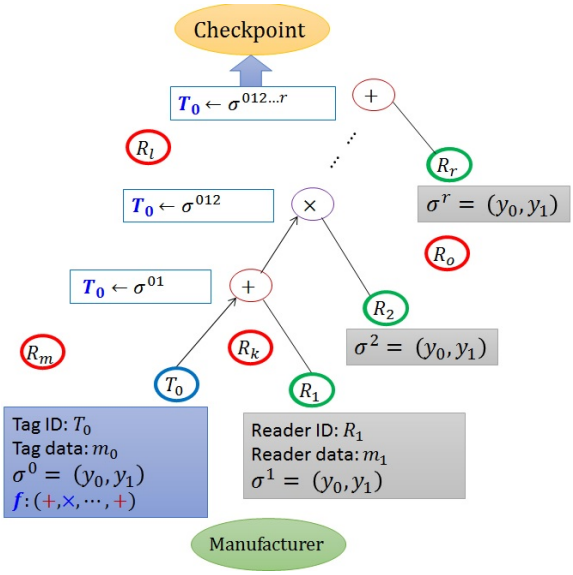

FIGURE 3. Path Evaluation in the Tag

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 $h(\texttt{reader\_data})$, $y_1 = (\mathcal{F}_K(\iota) - y_0)/s \bmod p$ 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 as secrets an evaluation key ek and a path code f where $\mathsf{size}(f) = |P|$. $\mathcal{M}$ computes $\tau \leftarrow f(y_0^0, \cdots, y_0^r)$ 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 $\mathcal{R}_j$, it 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 $\mathcal{R}_j$, tag $\mathcal{T}_i$ 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 $d^{j-1} \geqslant d^j$. Note that the degree of $y(z)$ remain fixed after evaluating an addition gate.

- If the current gate is '×': $\mathcal{T}_i$ 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 a multiplication gate.

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

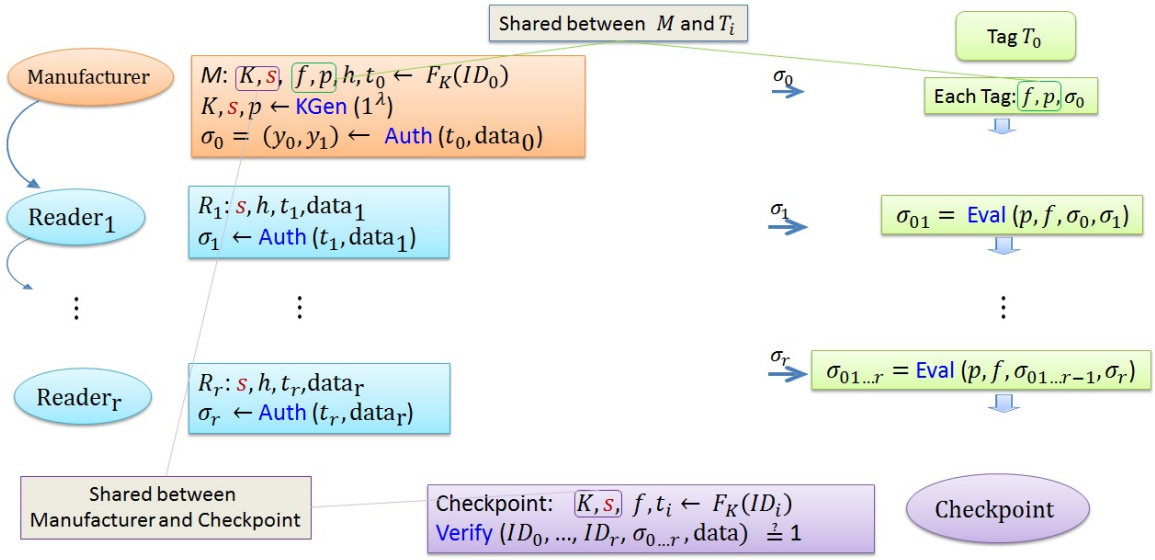

FIGURE 4. SupAUTH Construction with Building Blocks

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:

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

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 $\mathsf{Verify}(\mathsf{sk}, \tau, \Lambda, \sigma)$ where $\mathsf{sk} = \{s\}$.

- 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:

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

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

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 $h(\texttt{tag\_data})$ be 1, $h(\texttt{reader\_data})$ 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 $\tau(= 1\times2+3+4 = 9)$ 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:

$$\rho \stackrel{?}{=} \sum_{k=0}^{2} y_k x^k = 9 + 10 \times 4 + 2 \times 4^2 = 81 \text{ (for } 9 + 10z + 2z^2)$$

4. SECURITY ANALYSIS

Security of Path authentication: Security of SupAUTH scheme relies on the homomorphic message authenticator (HomMAC) in Catalano and Fiore (2013) where the security of HomMAC has been proven equivalent to unforgeability against chosen-message attacks(UF-CMA) in the random oracle model (ROM).

Let there exist $L := (\iota_1^*, \tau_1), \cdots, (\iota_r^*, \tau_r)$. A labeled program $\mathcal{P}^* = (f^*, \iota_1^*, ..., \iota_r^*)$ is *well defined* on L, if there exist $(\iota_i^*, *) \notin L$, there is $f^*(\tau_j \leftarrow (\iota_j, \tau_j) \in L \cup \hat{\tau}_j \leftarrow (\iota_j, *) \notin L)$ that provides same output irrespective of choosing $\hat{\tau}_j$.

Our HomMAC system is secure, such that, a probabilistic polynomial time adversary $\mathcal{A}$ who can adaptively observe the authentication tag σ corresponding to any arbitrary messages of his own choice, cannot generate a valid σ which was not really generated from the valid labelled program $\mathcal{P}$.

Consider an experiment $\mathsf{Exp}_{\mathcal{A},\mathsf{HomMAC}}(\lambda)$ between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

- Setup. The challenger generates $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$ and gives ek to $\mathcal{A}$. It also initializes a list $L = \{\emptyset\}$.

- Authentication queries. $\mathcal{A}$ can adaptively ask for label-message pair (ι, τ) s.t., $\tau \leftarrow h(\texttt{date})$ of its choice. If $\mathcal{C}$ receives any query (ι, τ) that is available in the list (s.t., $(\iota, *) \in L$), it simply ignores the query and feedback with the (ι, τ) as it received before. Else, it runs $\sigma \leftarrow \mathsf{Authentication}(\mathsf{sk}, \iota, \tau)$ algorithm, forwards σ to $\mathcal{A}$, and updates $L = L \cup (\iota, \tau)$.

- Forgery. Alike verification query $(\tau^*, \mathcal{P}^*, \sigma^*)$, adversary $\mathcal{A}$ is allowed to output a forgery $(\tau^*, \mathcal{P}^* = (f^*, \iota_1^*, ..., \iota_r^*), \sigma^*)$.

- Verification queries. Given a query $(\tau, \mathcal{P}, \sigma)$ by $\mathcal{A}$, $\mathcal{C}$ replies with either 1 (accept) or 0 (reject) by using algorithm $\mathsf{Verify}(\mathsf{sk}, \tau, \mathcal{P}, \sigma)$.

The experiment $\mathsf{Exp}_{\mathcal{A},\mathsf{HomMAC}}(\lambda)$ outputs 1 if and only if $\mathsf{Verify}(\mathsf{sk}, \tau, \mathcal{P}, \sigma) = 1$ and one of the following conditions holds:

- *Type 1 Forgery:* $\mathcal{P}^*$ is not well-defined on L.
- *Type 2 Forgery:* $\mathcal{P}^*$ is well defined on L and τ^* is not the correct output of $\mathcal{P}^*$ s.t., $\tau^* \notin f^*(\tau_j \leftarrow (\iota_j, \tau_j) \in L)$.

Two major improvements of HomMAC by Catalano and Fiore (2013) that constitute our protocol are: allowing adversary to query verification oracle and adapting the definition of forgery. Since tag/Reader IDs are unique, HomMAC scheme does not allow re-using a label (ι) to authenticate input data $h(\texttt{date})$ in order to track authenticated inputs uniquely. The notion of well defined programs is to define an adversary generated tuple (ι_j, τ_j) as *forgery*. That is why, even if the adversary trivially modify the circuit f by adding dummy gates and inputs, it does not violate security requirements. Verifier ensures that either $\mathcal{P}$ is run on valid inputs $(\iota_j, \tau_j) \in L$, otherwise, $(\iota_j, \tau_j) \notin L$ do not affect computation process in anyway.

Notice that in **Case-2**, *Checkpoint verification* of our protocol, we disallow the manufacturer $\mathcal{M}$ to share $\mathcal{P}$ with checkpoints (to obtain more privacy), instead $\mathcal{M}$ shares $\Lambda \leftarrow f(\eta_0, \cdots, \eta_r)$. However, it constitutes an infringement to the security of the protocol. Because it will allow the adversary to modify $\mathcal{P}$ in such a way (*e.g.*, adding dummy gates and inputs so that output remains same) that the modified circuit will remain equivalent to the previous one semantically. More clearly, scheme in Catalano and Fiore (2013) defines *well-defined program*. Hence *forgeries* without considering any tuples $(\iota_j, *) \notin L$ for verification. Therefore, for **Case-2**, we consider slightly weaker assumption, that is, we will not allow the adversary $\mathcal{A}$ to modify the circuit anyway, that will best match with the *Type 2 Forgery*

definition of Gennaro and Wichs in Gennaro and Wichs (2013).

Correctness: An authentication tag $\sigma = \mathsf{Authentication}(\mathsf{sk}, \iota, \tau)$ can correctly authenticate a message τ under a set of label identifiers ι if

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

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 $f : \mathbb{Z}_p^r \rightarrow \mathbb{Z}_p$ over $y^1, \cdots, y^r$. If a set of r triples $\{\tau_i, \mathcal{P}_i, \sigma_i\}$ such that $\mathsf{Verify}(\mathsf{sk}, \{\tau_i, \mathcal{P}_i, \sigma_i\}) = \mathsf{accept}$ then

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

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

In case of *tag authentication*, let a Reader receive b elements $z_1, \cdots, z_b$ from a valid tag $\mathcal{T}_i$, then it is obvious that one of the elements r' must satisfy $z = \mathfrak{f}_i(\mathcal{T}_i, r + y_0)$ for a polynomial $\mathfrak{f}_k = \{\mathfrak{f}_1, \cdots, \mathfrak{f}_m\}$. Since this scheme *solves* instead of *searches* for a tag $\mathcal{T}_i$ as in Wu and Stinson (2009), Reader considers $\mathcal{T}_i$ as the solution to the equation $z = \mathfrak{f}_i(\mathcal{T}_i, r + y_0)$. We consider using block cipher based PRF that outputs uniformly at random in $\mathbb{Z}_p$. However, the Reader needs to solve maximum $mb - 1$ (considering 1-degree polynomial) extra equations for authentication. If there are N tags registered with the Reader, the probability of collision, that is, two or more solutions for a single RFID tag $\mathcal{T}_i$ will be

$$1 - (1 - N/p)^{mb-1}$$

Succinctness: Our HomMAC is succinct. That is, the authentication tag σ of the program $\mathcal{P}$ should be certifiable using less communication than what was required to send the original inputs. 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.

Theorem 1. *If the Pseudo Random Function (PRF) $\mathcal{F}$ holds security and pseudorandom, then SupAUTH has path privacy property under the semantic security of Homomorphic authenticator scheme (HomMAC).*

Proof: Let an adversary $\mathcal{A}$ against the privacy game (described in Section 4.1) that can break the path privacy of the scheme. Suppose another adversary $\mathcal{B}$ to break the semantic security of HomMAC . $\mathcal{B}$ simulates $\mathcal{A}$ as a subroutine and responds to any query generated by $\mathcal{A}$. Assume the public evaluation key of the HomMAC scheme is ek, and its corresponding private key is sk and adversary $\mathcal{B}$ wants to break the semantic security of the scheme under $\mathsf{Evaluation}$ algorithm. In order to simulate the path authentication scheme, $\mathcal{B}$ first initializes the system with public parameters apart from the keys $\{\mathsf{K}, \mathsf{ek}\}$ of the Intermediate Readers and Checkpoints. Note that $\mathcal{B}$ knows the secret key of all the Readers $\{\mathsf{sk}\}$, but has no knowledge about $\{\mathsf{K}, \mathsf{ek}\}$. Later, adversary $\mathcal{A}$ begins path-privacy experiments in 4.1. In the learning phase, $\mathcal{A}_1$ queries the oracle and $\mathcal{B}$ answers. However, since $\mathcal{B}$ has no access to $\{\mathsf{K}, \mathsf{ek}, \mathsf{f}\}$, it can answer all the queries directly except ($\mathtt{Eval\_to\_T}$, $\mathtt{Path\_Verify}$). In order to answer the queries ($\mathtt{Eval\_to\_T}$, $\mathtt{Path\_Verify}$), $\mathcal{B}$ maintains a database L which

keeps the records of all the oracles operations' history. Therefore, in case of $\mathcal{A}_1$ query (`Eval_to_T`, `Path_Verify`), $\mathcal{B}$ cannot evaluate $st_{\mathcal{T}}$ (does not know f and ek) and can not verify σ_r (does not know K) to get Λ in order to compare the recorded value in the database. To answer these queries $\mathcal{B}$ maintains a database list L with the historical oracle output. $\mathcal{B}$ maintains the list L for $(i = 1, \ldots, n)$ tags with the tuples $(T_i, \sigma_0^i, \sigma_r^i)$. $\mathcal{B}$ updates its database by maintaining links between the tag's states (old and new). If the state of the tag changes, $\mathcal{B}$ adds a new link for the specific tag T_i. In this way, even though $\mathcal{B}$ has no access to $\{\mathsf{K}, \mathsf{f}\}$, it can answer the queries `Eval_to_T`, `Path_Verify` and trace the tag T_i through the records of the state in the database. $\mathcal{A}_1$ outputs two tags T_0 and T_1 at the end of the learning phase, a path $\mathcal{P}$ of k steps. Assume the state of T_0 is (r_0, σ_0) and the state of T_1 is (r_1, σ_1) where r_b is the $\texttt{reader}_{\texttt{data}}$ of the state T_b. At first, $\mathcal{B}$ submits two $\texttt{reader}_{\texttt{data}}$ (r_0, r_1) to the authentication oracle `Auth`. `Auth` randomly chooses $b \in \{0, 1\}$ and sends $h(r_b)$ under the hash function h. $\mathcal{B}$ forwards $(h(r_b), r)$ to `Evaluate` oracle where r is any arbitrary random string with $|r| = |\sigma_b|$. $\mathcal{B}$ was supposed to provide $(h(r_b), \sigma_b)$ to `Evaluate`, where σ_b is the value of the k^{th} Reader in path P. After that $\mathcal{B}$ evaluates σ_b to σ_b' by using `Evaluate` oracle where σ_b' is the new value of σ_b that has been processed by tag after k^{th} Reader in the path P. $\mathcal{B}$ forwards $(h(r_b), r')$ to $\mathcal{A}_2$. In fact, $\mathcal{B}$ provides $(h(r_b), \sigma_b')$ to $\mathcal{A}_2$, where σ_b' is the value of the k^{th} tag along the path P.

We claim that $(h(r_b), r')$ and $(h(r_b), \sigma_b')$ contain same information to be used by $\mathcal{A}_2$. Since $\mathcal{B}$ has no access to the *gate* of circuit f and $\mathcal{F}_K$ is a pseudo random function, $\mathcal{A}_2$ cannot guess any information from $\sigma_b^p rime$. $\mathcal{A}_2$ assumes the value of b by analyzing $h(r_b), r'$. Hence, $\mathcal{B}$ produces the same output as $\mathcal{A}$. Assuming the pseudorandomness of $\mathcal{F}$, the advantage of $\mathcal{B}$ to break the semantic security of Homomorphic authenticator scheme (HomMAC) is the same as the advantage of $\mathcal{B}$ to break the path privacy of the scheme. Since the HomMAC is *semantic secure*, protocol transcripts do not leak any information about the tag's identity as well. Therefore, we conclude that SupAUTH our scheme satisfies path privacy with tag-privacy.

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:

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

where k is the maximum degree of $\mathfrak{f}_i$ and $n \geqslant (k+1)^2/k$

Proof: If $\mathcal{A}$ knows $\mathcal{T}_i$ of n tags, then using algorithm in Guruswami and Sudan (1998) could recover $\mathfrak{f}_1, \cdots, \mathfrak{f}_m$. However, if the tags are assumed to be *tamper-proof*, then no algorithm can recover $\mathfrak{f}$ efficiently.

Let n tags be compromised and the adversary $\mathcal{A}$ obtains the univariate polynomials $(\mathfrak{f}_{1,\mathcal{T}_i}(\beta), \cdots, \mathfrak{f}_{m,\mathcal{T}_i}(\beta))$. Target of $\mathcal{A}$ is to recover the bivariate polynomial $(\mathfrak{f}_1(\alpha, \beta), \cdots, \mathfrak{f}_m(\alpha, \beta))$. We can express polynomial assignment in matrix form. Let $X \in \mathbb{Z}_p^{n \times k+1}$ be a matrix with secret $\mathcal{T}_i$ of n tags and $S_1, \cdots, S_m$ where $S_i \in \mathbb{Z}_p^{(k+1)\times(k+1)}$ is the matrix representation of bivariate polynomial $\mathfrak{f}_i(\alpha, \beta)$ stored in the server. Let $Y_i = XS_i \in \mathbb{Z}_p^{n\times(k+1)}$, then univariate polynomials assigned to tag i are $Y_1[i], \cdots, Y_m[i]$. Let $\mathcal{A}$ know Y_i and intend to recover S_i. A necessary condition to solve the problem is $n \geqslant (k+1)^2/k$ Wu and Stinson (2009). Since a tag selects $\mathfrak{f}_i$ from randomly $\mathfrak{f}_1, \cdots, \mathfrak{f}_m$, probability of $\mathcal{A}$ to recover $\mathfrak{f}_i$ is

$$\Pr[Adv_{\mathcal{A}}] = 1/m^{n-1} \leqslant m^{-k-2} \text{ i.e., } n \geqslant (k+1)^2/k$$

Note that, unlike authentication scheme in Wu and Stinson (2009), we consider 1-degree

polynomial for $\mathfrak{f}_i$ in our scheme. Therefore, probability of $\mathcal{A}$ to recover $\mathfrak{f}_i$ is $\Pr[Adv_{\mathcal{A}}] \leqslant 1/m^3$ in our case. However, in order for the system remain secure, we could increase m and/or k (to lower $Adv_{\mathcal{A}}$). We carefully observe that increasing m is more effective than to increase k (when $k < m$). Therefore, we propose to increment the value of system parameter m so that $Adv_{\mathcal{A}}$ remains the same.

4.1. **Privacy**

In order to define privacy, we analyzed our protocol according to the *path-privacy* framework in Cai et al. (2012) 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 Cai et al. (2012), except for some minor modifications. First, we explicitly allow the adversary to query Readers during Move operation. Second, unlike path authentication scheme in Cai et al. (2012), our state update operation takes place inside the tag with some secrets, such as, the coefficient of the tag's polynomial σ_0 and the circuit information f.

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

Proof. Let $\mathcal{A}$ be a PPT adversary against RFID path authentication that takes the system's public parameters, a set of Readers $\mathcal{R}$, a set of tags $\mathcal{T}$, and a set of checkpoints $\mathcal{D}$ as input. Adversary $\mathcal{A}$ 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 $1 \leqslant k < |\mathcal{P}|$ 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:

- $\mathtt{Read\_frm\_R}(\mathcal{R}_i)$: 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.

- $\mathtt{Eval\_to\_T}(\mathcal{T}_j,)$: 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.

- $\mathtt{Path\_Verify}(st_{\mathcal{T}_j})$: On input state information st, this oracle verifies whether $\mathcal{T}_j$ has followed the valid path $\mathcal{P}_v$ and outputs 1 for *success* and 0 for fail.

- $\mathtt{Move}(\mathcal{T}_j, k, \mathcal{K}, b)$: 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 $st_{\mathcal{T}_j}$.

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

(1) Run $\mathsf{Setup}(\lambda)$ 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 $|\mathcal{P}| \geqslant k \geqslant 1$ and st is current state information.
(3) $b \leftarrow \{0, 1\}$.
(4) $st_{\mathcal{T}_b} \leftarrow \texttt{Move}(\mathcal{T}_b, k, \mathcal{P}, b)$. $st_{\mathcal{T}_b}$ 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 $b' = b$, and 0 otherwise.

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 (Steps 3-5), after tossing a coin, $\mathbf{Exp}_{\mathcal{A}}^{\texttt{Path}-\texttt{Privacy}}$ chooses either $\mathcal{T}_0$ or $\mathcal{T}_1$ and moves through k remaining Readers 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 $\mathbf{Exp}_{\mathcal{A}}^{\texttt{Path}-\texttt{Privacy}}$ provides $\mathcal{A}_2$ with last state of $\mathcal{T}_i$, that is, st_i and the 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 of more than $1/2$.

5. EXTENSION TO THE PROTOCOL (OPTIONAL)

5.1. Leverage protocol with 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 a 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 Wu and Stinson (2009) with major modifications (*e.g.*, mutual authentication) (see Table 1). The modified protocol is scalable and its security and privacy is also based on the hardness of reconstructing a polynomial with noisy data.

Unlike Wu and Stinson (2009), we use two tag parameters, $\mathcal{T}_i, y_0$, as secret, 1-degree bivariate set of polynomials $\mathfrak{f}$, no hash function employed in the tag, and only b random numbers between the tag and Reader. Reader initiates the protocol with the *hash value* on the Reader's meta data, $h(\texttt{reader\_data})$. The tag follows the protocol transcripts in Wu and Stinson (2009). Upon receiving the feedback, the Reader authenticates the tag and forwards Reader (r's) initialization tag $y_1^{(r)}$ to update path status and $z_i^{(t)}$ to authenticate the Reader to the tag. Note that the tag updates the current status st_j only if it can authenticate the Reader successfully (see Figure 5).

Security of this mutual authentication protocol can be realized from the hardness of *noisy polynomial interpolation* (NPI) problem. In Wu and Stinson (2009), authors consider query-and-recovery attack model where an attacker repeatedly queries and collects corresponding responses from a tag in order to recover the polynomial assigned to it and relate the attack

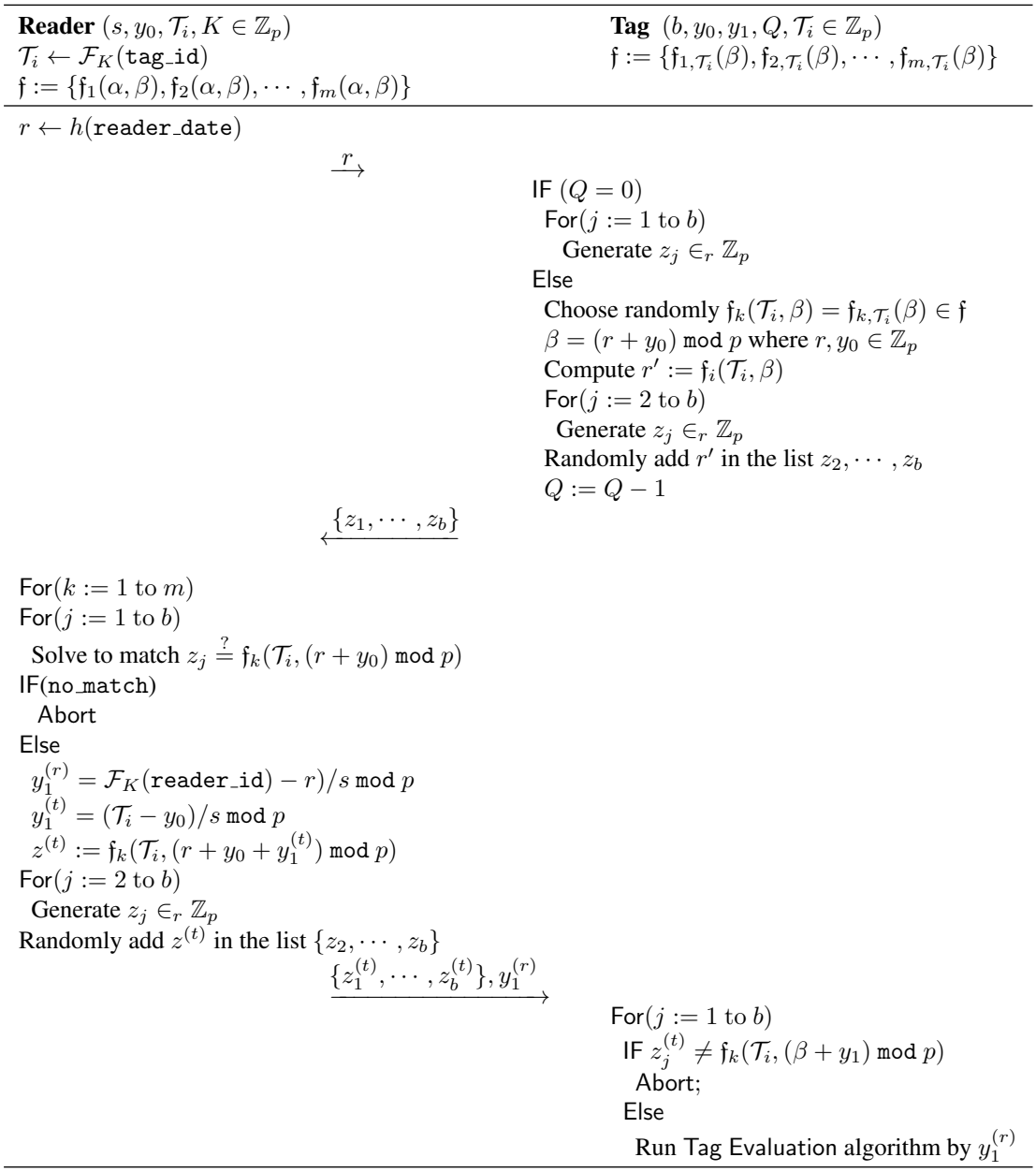

FIGURE 5. Update protocol with RFID Authentication

to the hardness of solving NPI problem. We refer the Readers to Wu and Stinson (2009) for detailed techniques and security proof. In the following we provide a sketch of the security proof followed by Wu and Stinson (2009).

Polynomial reconstruction Problem: Security of the authentication scheme described in Wu and Stinson (2009) is based on the hardness of the well-known *Noisy Polynomial Interpolation Problem*(NPI) Naor and Pinkas (2006). 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

TABLE 1. Comparison between Authentication Protocols

	SupAUTH	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	$b+3$	$2b+1$

the NPI problem. We refer to Wu and Stinson (2009) for necessary definitions. Note that we slightly modify the existing protocol to reduce communication and computational overhead of the protocol. Since we convert original tag-only authentication (Wu and Stinson (2009)) to *mutual* authentication, 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 $h(\texttt{reader\_data})$ to be considered as random even if the adversary knows `reader_data`. This r' is forwarded along with extra $b-1$ random elements. In every consecutive m queries (where $m < Q$) by the adversary, the tag employs all of its polynomial $\mathfrak{f}_i, 1 \leqslant i \leqslant m$ 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*

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

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 Wu and Stinson (2009) for the details of the proof.

5.2. **Batch initialization of the tags**

In Cai et al. (2012), 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 enroll 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 the same meta data, they share the same y_0. After initializing the batch of tags, $\mathcal{M}$ evaluates the circuit on polynomials $y^{(i)}(z),\ 1 \leqslant i \leqslant n$ 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.

6. PERFORMANCE EVALUATION

SupAUTH is secure and marginally efficient. Compared to the previous works, our scheme offers conditional memory reduction on tags with marginal computational overhead in order to achieve transmission path confidentiality. For performance evaluation, we ignore any computations related to system initialization described in Figure 2. that can be pre-processed off-line by the manufacturer or supplier. Besides that, major computations are performed by the Checkpoint verifier. In this section, we mainly discuss about the tag's memory requirement and computational overhead during interaction with the Readers along the supply chain path.

TABLE 2. Comparison among path authentication protocols

	SupAUTH Ours	ISPEC'16 Chang et al. (2016)	ACNS'12 Cai et al. (2012)	NDSS'11 Blass and Elkhiyaoui (2011)	SEC'12 Cai et al. (2012)	RFIDSec'09 Ouafi and Vaudenay (2009)
Path generation	static	static	static	static	dynamic	dynamic
Building blocks	HomMAC	ECElGamal	ECElGamal + PRF	ECElGamal	OMS	PRF
Privacy	PUlink$^{\ddagger\,\dagger}$	PUlink‡	PUlink†	TUlink + SUlink†	TUlink + PUlink	NG
Path evaluation	Tag	Reader	Reader	Reader	Reader	Tag
Mutual authentication	Yes	No	No	No	No	Yes
Tag storage	257* bits *or* 513* bits	800 bits	480 bits	960 bits	720 bits	NG
Reader storage	$O(1)$	$O(1)$	$O(1)$	$O(1)$	$O(n)$	O(N)
Checkpoint storage	$O(N)$	O(N)	$O(N)$	$O(N + vP)$	-	-
Tag computation	PolyA or PolyM	–	–	–	3H	3H
Reader computation	1H	6ECE, 4ECM	2ECM, 2ECA, 1PRF	8ECE, 3ECM, 2HMAC	1DEC, 3P, 4EX, 6M	1PRF, 1OWF, 1H

HomMAC: Homomorphic MAC on Arithmetic Circuit, HMAC: Homomorphic MAC, 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, ECE: Elliptic Curve exponentiation, 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 unlinkability where adversary has access to the Reader during `Move` operation.
†Privacy proof included.
*Considering $32d + 1$ or $64d + 1$ s.t., max degree of a polynomial $d = 8$ with prime p (32-bit or 64-bit).

Computation cost of the *tag evaluation* depends on the size and gate types of the circuit f. Let the polynomial to be evaluated inside the tag be grown with a degree of d (see Figure. 1). Then the succinctness of the evaluated polynomial can be ensured while $d < |f|$.

Horner's rule provides a computationally efficient method of forming a polynomial from a list of its coefficients. Let two univariate polynomials be $p = \sum_{k=0}^{d} a_k X^k$ and $q = \sum_{k=0}^{n} b_k X^k$ over $\mathbb{Z}_p[x]$. Calculating a similar representation of their product

$$p.q = \sum_{k=0}^{d+n} c_k X^k \qquad c_k = \sum_{i=0}^{k} a_i.b^{k-i} \tag{1}$$

This takes $O(dn)$ additions and multiplication in $\mathbb{Z}_p[x]$. Given 1-degree polynomial for each

Reader (see Figure 3.), $O(dn)$ will be $O(d.2) \approx O(d)$ in our case.

More clearly, following Horner's method for polynomial evaluation causes the number of coefficients grow by a factor of $d^n - 1$ for a multivariate polynomial of n Sunar and Cyganski (2005). In our case, it would be d (for univariate polynomial $n = 1$). That is, for any degree/level d, the maximum number of additions or multiplication is $d.2$ (for 1-degree polynomial $d = 2$). Therefore, the total number of additions and multiplications required at any stage is $3d$ ($2d$ multiplication + d addition).

Minimal tag evaluation cost, more precisely, minimizing the interaction time between the tag and reader is one of the major challenges of passive RFID tag deployment where tags must not exceed 500 ms in reading process Han et al. (2015). THreefore, several theoretical methods have been proposed so far to improve the costliest multiplication algorithm. For example, well-known Fast Fourier Transformation (FFT) method requires $O(n.\log n.\log\log n)$ time for an n-bit integer multiplication. In Wieser and Hutter (2014), authors evaluate various large-integer multiplication methods suitable for passive RFID. They synthesize 21 different well-known multipliers with different input-word sizes targeting a typical frequency of low-power RFID devices (1 MHz). Their results show that FFT-based multipliers are not suitable for resource constrained tiny devices like RFIDs. Digit-serial multiplier outperform others in all metric- low-power, time and low-area. Allowing several bits to be processed in parallel can also boost its performance. We evaluate SupAUTH based on the result of efficient hardware implementation of Digit-serial multiplication algorithm.

TABLE 3. Maximum power consumption and calculation time for the tag evaluation assuming max. degree of polynomial $d = 8$

Multiplication Type	Power consumption (in μw)		Calculation Time (in ms)	
	32-bit	64-bit	32-bit	64-bit
Bit-Serial	32.4	65	57.3	24.5
Digital-Serial	43.2	61.44	24.5	12.2

Table 3. shows the average power consumption and calculation time of our scheme based upon Bit-Serial and Digital-Serial multipliers. Our results show that using digital-serial multiplier the maximum computation time for the tag evaluation is 57.3 ms which is suitable for any low-cost passive RFID deployment e.g. EPCGlobal Class 1 Generation 2 tags.

Note that, *addition gates* of the circuit f will not increase the value of d as the tag moves, but each *multiplication gate* will cause the increment of the value of d by 1. Considering 32-bit long prime for $\mathbb{Z}_p$, initially a tag requires 64-bits (1-degree polynomial), that grows up to $32d+1$-bits. For simplicity, assuming the circuit f with maximum 8 multiplication gates, the amount of tag memory space required is approximately 257 bits (to achieve 32-bit security) and 513 bits (to achieve 64-bit security) (see Table 2.).

On the other hand, the maximum cost of verification in the checkpoint Reader is $O(|f| + d)$ from the computation cost of $\Lambda = f(\eta_0, \cdots, \eta_r)$, and $\sum_{l=0}^{d} y_l s^l$. However, in Case-2 of the verification algorithm where Λ is considered to be pre-shared between the manufacturer and Checkpoint, the verification cost is $O(d)$. Moreover, for additional *mutual authentication* protocol, at each step of the path, a tag needs to generate $b - 1$ random numbers, compute 1-degree polynomial (1 modular addition and multiplication), and store $2m$ items over $\mathbb{Z}_p$.

7. CONCLUSION

RFID-enabled path authentication is a research area in RFID based inventory control system. In this paper, we briefly studied existing state of the art solutions for RFID-enabled path authentication schemes for a supply chain management and present a new privacy direction in RFID-enabled path authentication using an arithmetic circuit based Homomorphic MAC in order to achieve confidentiality of the tag movement. In addition, we introduce a polynomial-based mutual authentication protocol and a batch initialization protocols that can be optionally used as a security and performance extension to the main protocol.
Compared to existing Elliptic curve Elgamal Re-encryption (ECElgamal) based solution, Homomorphic Message Authentication Code Arithmetic circuit based SupAUTH offers memory efficiency with a marginal computational overhead in the tag, computational efficiency on the Reader, and adds a new privacy direction to the state-of-the-art path privacy applicable to IoT technologies.

REFERENCES

BLASS, ERIK-OLIVER, and KAOUTAR ELKHIYAOUI. 2011. Tracker: security and privacy for rfid-based supply chains. *In* Proceedings of the 18th Annual Network and Distributed System Security Symposium, NDSS'11, Citeseer.

BONEH, DAN, BEN LYNN, and HOVAV SHACHAM. 2004. Short signatures from the weil pairing. Journal of cryptology, **17**(4):297–319.

CAI, SHAOYING, ROBERT H. DENG, YINGJIU LI, and YUNLEI ZHAO. 2012. A new framework for privacy of rfid path authentication. *In* Proceedings of the 10th International Conference on Applied Cryptography and Network Security, ACNS'12, Springer-Verlag, Berlin, Heidelberg, pp. 473–488.

CAI, SHAOYING, YINGJIU LI, and YUNLEI ZHAO. 2012. Distributed path authentication for dynamic rfid-enabled supply chains. *In* IFIP International Information Security Conference, Springer, pp. 501–512.

CATALANO, DARIO, and DARIO FIORE. 2013. Practical homomorphic macs for arithmetic circuits. *In* Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, pp. 336–352.

CHANG, SANG-YOON, SHAOYING CAI, HWAJEONG SEO, and YIH-CHUN HU. 2016. Key update at train stations: Two-layer dynamic key update scheme for secure train communications. *In* International Conference on Security and Privacy in Communication Systems, Springer, pp. 125–143.

GENNARO, ROSARIO, and DANIEL WICHS. 2013. Fully homomorphic message authenticators. *In* International Conference on the Theory and Application of Cryptology and Information Security, Springer, pp. 301–320.

GURUSWAMI, VENKATESAN, and MADHU SUDAN. 1998. Improved decoding of reed-solomon and algebraic-geometric codes. *In* Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on, IEEE, pp. 28–37.

HAN, WEILI, YIN ZHANG, ZEQING GUO, and ELISA BERTINO. 2015. Fine-grained business data confidentiality control in cross-organizational tracking. *In* Proceedings of the 20th ACM Symposium on Access Control Models and Technologies, ACM, pp. 135–145.

KATZ, JONATHAN, and JI SUN SHIN. 2006. Parallel and concurrent security of the hb and hb+ protocols. *In* Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, pp. 73–87.

MAMUN, MOHAMMAD SI, ATSUKO MIYAJI, and MOHAMMAD S RAHMAN. 2012. A secure and private rfid authentication protocol under slpn problem. *In* International Conference on Network and System Security, Springer, pp. 476–489.

MAMUN, MOHAMMAD SAIFUL ISLAM, and ATSUKO MIYAJI. 2013. A fully-secure rfid authentication protocol from exact lpn assumption. *In* 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, IEEE, pp. 102–109.

MAMUN, MOHAMMAD SAIFUL ISLAM, and ATSUKO MIYAJI. 2014. A scalable and secure rfid ownership transfer protocol. *In* 2014 IEEE 28th International Conference on Advanced Information Networking and Applications, IEEE, pp. 343–350.

MAMUN, MOHAMMAD SAIFUL ISLAM, and ATSUKO MIYAJI. 2015. A privacy-preserving efficient rfid

authentication protocol from slpn assumption. International Journal of Computational Science and Engineering, **10**(3):234–243.

MORIYAMA, DAISUKE. 2013. An rfid authentication protocol with flexible path authentication. *In* RFID-Technologies and Applications (RFID-TA), 2013 IEEE International Conference on, IEEE, pp. 1–6.

NAOR, MONI, and BENNY PINKAS. 2006. Oblivious polynomial evaluation. SIAM Journal on Computing, **35**(5):1254–1281.

OUAFI, KHALED, and SERGE VAUDENAY. 2009. Pathchecker: An rfid application for tracing products in supply-chains. *In* Proceedings of RFIDSec 2009, Number LASEC-CONF-2009-006.

SHPILKA, AMIR, and AMIR YEHUDAYOFF. 2010. Arithmetic circuits: A survey of recent results and open questions. Foundations and Trends® in Theoretical Computer Science, **5**(3–4):207–388.

SUNAR, BERK, and DAVID CYGANSKI. 2005. Comparison of bit and word level algorithms for evaluating unstructured functions over finite rings. Lecture notes in computer science, **3659**:237.

WANG, HONGBING, YINGJIU LI, ZONGYANG ZHANG, and ZHENFU CAO. 2012. Two-level path authentication in epcglobal network. *In* 2012 IEEE International Conference on RFID (RFID), IEEE, pp. 24–31.

WANG, HONGBING, YINGJIU LI, ZONGYANG ZHANG, and YUNLEI ZHAO. 2016. Efficient tag path authentication protocol with less tag memory. *In* International Conference on Information Security Practice and Experience, Springer, pp. 255–270.

WIESER, WOLFGANG, and MICHAEL HUTTER. 2014. Efficient multiplication on low-resource devices. *In* Digital System Design (DSD), 2014 17th Euromicro Conference on, IEEE, pp. 175–182.

WU, JIANG, and DOUGLAS R STINSON. 2009. A highly scalable rfid authentication protocol. *In* Australasian Conference on Information Security and Privacy, Springer, pp. 360–376.