Information **2015**, *6*, 258-274; doi:10.3390/info6020258

OPEN ACCESS

information

ISSN 2078-2489

www.mdpi.com/journal/information

Article

Efficiency and Privacy Enhancement for a Track and Trace System of RFID-Based Supply Chains

Xunjun Chen [1,2,*], Yuelong Zhu [1], Jiguo Li [1], Yamin Wen [3] and Zheng Gong [4]

[1] College of Computer and Information, Hohai University, Nanjing 210098, China; E-Mails: ylzhu@hhu.edu.cn (Y.Z.); ljg1688@163.com (J.L.)

[2] Department of Computer Science, Taizhou University, Linhai 317000, China

[3] School of Mathematics and Statistics, Guangdong University of Finance & Economics, Guangzhou 510320, China; E-Mail: yamin.wen@gmail.com

[4] School of Computer Science, South China Normal University; Guangzhou 510631, China, E-Mail: cis.gong@gmail.com

* Author to whom correspondence should be addressed; E-Mail: neojun@hhu.edu.cn; Tel./Fax: +86-576-88-800-356.

Academic Editors: Qiong Huang and Guomin Yang

Received: 2 April 2015 / Accepted: 4 June 2015 / Published: 11 June 2015

Abstract: One of the major applications of Radio Frequency Identification (RFID) technology is in supply chain management as it promises to provide real-time visibility based on the function of track and trace. However, such an RFID-based track and trace system raises new security and privacy challenges due to the restricted resource of tags. In this paper, we refine three privacy related models (*i.e.*, the privacy, path unlinkability, and tag unlinkability) of RFID-based track and trace systems, and clarify the relations among these privacy models. Specifically, we have proven that privacy is equivalent to path unlinkability and tag unlinkability implies privacy. Our results simplify the privacy concept and protocol design for RFID-based track and trace systems. Furthermore, we propose an efficient track and trace scheme, Tracker+, which allows for authentic and private identification of RFID-tagged objects in supply chains. In the Tracker+, no computational ability is required for tags, but only a few bytes of storage (such as EPC Class 1 Gen 2 tags) are needed to store the tag state. Indeed, Tracker+ reduces the memory requirements for each tag by one group element compared to the Tracker presented in other literature. Moreover, Tracker+ provides privacy against supply chain inside attacks.

Keywords: Radio Frequency Identification (RFID); provable privacy; track and trace; unlinkability; supply chain

1. Introduction

Today, Radio Frequency Identification (RFID) tags are extensively used to track and identify goods, supplies, and equipments. In these applications, the tags are physically attached to objects, providing a convenient management of supply chains. Such convenience depends on the track and trace function of RFID-based supply chains, while such a track and trace system provides real-time visibility for supply chains. Thus, this may allow hackers to breach privacy by tracing and observing the tag through time and space. Since RFID tags are equipped with limited computational ability and storage, the design of track and trace system for RFID-based supply chains may bring new privacy and security challenges.

Recently, Blass *et al.* presented three kinds of privacy-related models [1] for RFID-based track and trace systems: privacy, path unlinkability, and tag unlinkability. Unfortunately, the definitions of privacy and path unlinkability in [1] are incomplete since they depend on the impractical assumption that each tag goes through each step (or each path) in supply chains with the same probability. Moreover, the above three kinds of privacy models are too complicated to understand the privacy of RFID-based track and trace systems. Can these privacy requirements be simplified? In other words, what are the relations among privacy, path unlinkability, and tag unlinkability? These problems have not yet been addressed in the literature.

In addition, RFID tags are resource-restricted devices, especially the EPC Class 1 Gen 2 tags [2], which have very limited memory and support only simple operations such as XOR, CRC, and the 16-bit random number generator. Moreover, the tag is passive and not tamperproof. Therefore, it cannot provide secure access control and authentication to readers. During the life cycle of a tag in the RFID-based supply chain, how to prepare the tag data in a way to enable secure and private track and trace becomes a substantial challenge. The existing track and trace scheme Tracker [1] aims to address this problem. However, it cannot guarantee the claimed privacy since the signature part of the internal state of each tag is unchanged for each path. Hence, an adversary can trace the tag by comparing the signature part of its current internal state with the previous one. Therefore, it is of vital importance to develop an efficient and secure track and trace scheme for RFID-based supply chains.

1.1. Our Contributions

In this paper, we address the abovementioned track and trace problems of RFID-based supply chains. The main contributions are as follows.

(1) We refine three privacy-related models reported in [1], the definitions of which rely on the impractical assumption that each tag goes through each step (or each path) in supply chains with the same probability. Our refined and improved models do not depend on such an assumption and capture the privacy requirements and the essences of RFID-based supply chains intuitively.

(2) We clarify the relations among privacy, path unlinkability, and tag unlinkability. Specifically, it has been proven that privacy is equivalent to path unlinkability and tag unlinkability implies privacy.

Our results simplify the privacy requirements for a track and trace system of RFID-based supply chains, and promise to design efficient and simple privacy-preserving track and trace schemes for RFID-based supply chains.

(3) We propose an efficient track and trace scheme, Tracker+. The Tracker+ allows for authentic and private identification of RFID-tagged objects in supply chains. In Tracker+ , only a few bytes of storage (such as EPC Class 1 Gen 2 tags) is needed to store the tag state, while no computational ability is required for tags. Indeed, Tracker+ improves Tracker [1] by reducing the memory requirement of one group element for each tag and by providing privacy against supply chain inside attacks. The efficiency and privacy enhancement of Tracker+ is attributed to the randomness reuse technique and the randomized HMAC [3] method.

1.2. Related Work

RFID-related security and privacy issues have been widely studied in the literature, such as a survey [4] and a more up-to-date bibliography [5]. Most of this research focused on tag-reader interactions [6–13]; however, only a few reported the secure and privacy-preserving supply chain management, especially the RFID-based track and trace systems. For example, Ouafi and Vaudenay [14] addressed verification of the genuineness of products using strong cryptographically RFID tags. In their solution, tags authenticate readers at every step in the supply chain. The tags will update their internal state if the readers are successfully authenticated. The evaluation of authentication relies on two hash functions, one of which is for authentication of readers and the other is for tags' state update. Li and Ding [15] proposed a similar approach with tags evaluating cryptographic hash functions.

1.3. Organization

The rest of the paper is organized as follows. In Section 2, we provide the technical precedents of the track and trace system. In Section 3, we introduce the security requirements for the track and trace system. In Section 4, we clarify the relations among privacy models of track and trace system. In Section 5, we propose an efficient track and trace system, Tracker+. In Section 6, we prove the security of Tracker+ and analyze its efficiency. Finally, Section 7 concludes this paper.

2. Preliminaries

In this section, we describe the mathematical conventions, the definition of supply chain, and the model of track and trace system. We use terms and expressions similar to the ones used by Ma *et al.* [16] and Blass *et al.* [1].

Mathematical Preliminaries: If $A(\cdot,\cdot,...)$ is a randomized algorithm, then $y \leftarrow A(x_1, x_2, ...; cn)$ means $y(x_1, x_2, ...)$ that y is assigned the unique output of the algorithm A on inputs $x_1, x_2, ...$ and coins cn, while $y \xleftarrow{\$} A(x_1, x_2, ...; cn)$ is shorthand for first picking cn at random and then setting $y \leftarrow A(x_1, x_2, ...; cn)$. Let $y \leftarrow A^{O_1,...,O_n}(x_1, x_2, , ...)$ denote that y is assigned the output of the algorithm A, which takes $x_1, x_2, ...$ as inputs and has oracle accesses to $O_1, ..., O_n$. If $x_1, x_2, ...$ are strings, then $x_1 \| x_2 \| \cdots$ denotes the concatenation of them. If x is a string then $|x|$ denotes its bit length in binary

code. If S is a set then $s \in_R S$ indicates that s is chosen uniformly at random from S and $|S|$ denotes its cardinality (*i.e.*, the number of elements of S). Let $\Pr[E]$ denote the probability that an event E occurs.

2.1. Supply Chain

As described in [1], there are four kinds of entities in a track and trace system of RFID-based supply chains: the tags, the issuer I, the readers, and the manager M. At first, the issuer I prepares the initial state of the tag that will enter the supply chain. Then, the products go through the supply chain and the reader interacts with their tags at each supply chain step. Finally, the manager M verifies the validity of a tag at the end of its trip.

Throughout this paper we denote a supply chain as a series of consecutive steps that a product has to pass through. Formally, a supply chain SC is represented by a digraph $G = (V, E)$ comprising of vertices V and edges E. A vertex $v_i \in V = \{v_0, v_1, \cdots, v_{\eta-1}\}$ is equivalent to one step in the supply chain SC. Each vertex/step v_i in the supply chain is uniquely equipped with a reader R_i. Every directed edge $e \in E$, $e := \overrightarrow{v_i v_j}$, from vertex v_i to vertex v_j, indicates that v_j is a possible next step to step v_i in SC. If products must not pass from step v_i to v_j, then $\overrightarrow{v_i v_j} \notin E$. Whenever a product in the supply chain proceeds from step v_i to step v_j, reader R_j interacts with the product's tag.

Issuer I is represented in G by the unique vertex without incoming edges v_0. A path P is a finite sequence of steps $P = \{v_0, \cdots, v_\ell\}$, where $\forall i \in \{0, \cdots, \ell\} : \overrightarrow{v_i v_{i+1}} \in E$ and ℓ is the length of path P. A valid path P_{valid} represents a particular legitimate sequence of steps in the supply chain. We assume there are ν multiple different valid paths in a supply chain. The manager M will check for T_i's path validity in the checkpoint, which is the last step v_ℓ of a valid path $P_{valid_i} = \{v_0, \cdots, v_\ell\}$.

2.2. Track and Trace System

Formally, a track and trace system $\mathrm{TK} = \{G, R, T, I, M, P_{valid}, S_{valid}\}$ consists of the following components:

Initialize(κ): Upon the security parameter κ, the system prepares a supply chain G, an issuer I and a manager M, a set of n tags T, a set of η readers R, a set of ν valid paths P_{valid}, and a set of valid state S_{valid}.

Read(T_i): A function that reads out tag T_i and returns its current state $s_{T_i}^j$.

Write(T_i): A function that writes a new state $s_{T_i}^{j+1}$ into tag T_i.

GoNext(T_i): The tag position transition function, which transports the tag T_i from its current step to its next step. Let its current state be $s_{T_i}^j$. After this transportation, its state has been transformed to $s_{T_i}^{j+1}$ through the above Read and Write operations, where $s_{T_i}^{j+1} = f(s_{T_i}^i)$ and $f : S \to S$ is a state transition function.

Check($s_{T_i}^j$): A function that verifies whether tag T_i has been through a valid path P_{valid_i}. If is the case then return the valid path P_{valid_i}, $\varnothing$ otherwise.

3. Security Requirements

In this section, we introduce the security model of the track and trace system based on the following assumptions. One is that the readers in the supply chain are independent and the other is that a reader R_i

at step v_i behaves correctly. For instance, a reader R_i at step v_i, which corresponds to quality control, does not update the state of T_j unless the product attached to T_j satisfies the quality requirements. Basically, security requirements of track and trace system consist of authenticity, privacy, and unlinkability, which are defined in the following subsections.

3.1. Authenticity

The main security goal of the track and trace system is to prevent an adversary from forging a tag's internal state with a valid path that was not actually taken by the tag in the supply chain. It is formalized by the following experiment Exp_A^{aut} (*cf.*, Experiment 1), where the adversary $A=(A_1, A_2)$ runs in two phases. Let O_{CP} denote the operation (or oracle) that corrupts the internal party v_i of supply chains. It returns the secret information of party v_i. Also let O_W, O_G, O_C, O_R, and O_W denote **GoNext**, the **Check**, **Read**, and **Write** functions, respectively. First, in the learning phase, A can query the five oracles in any order to learn useful information, with the restriction that it cannot query $O_{CP}(v^*)$. Then, in the challenge phase, A is asked to output a tag T_i. The total number of A's oracle queries does not exceed ρ.

Experiment 1. The authenticity experiment.

Experiment $\mathrm{Exp}_A^{aut}[\kappa,\rho]$
(1) initialize the Tracker system through **Initialize(** κ **)**;
(2) choose an honest party v^*;
(3) $st \leftarrow A_1^{O_G,O_R,O_W,O_C,O_{CP}}(\mathrm{TK})$; //*learning stage*
(4) $T_i \leftarrow A_2(\mathrm{TK}, st)$; //*challenge stage*
(5) $s_{T_i}^j \leftarrow$ **Read** (T_i)
(6) if **Check(** $s_{T_i}^j$ **)** $= P_{valid_k}$ and tag T_i has not been through the step v^* and $v^* \in P_{valid_k}$ then output 1; 0 otherwise.

Definition 1. The advantage of adversary in the experiment $\mathrm{Exp}_A^{aut}[\kappa,\rho]$ is defined as:

$$\mathrm{Adv}_A^{aut}(\kappa,\rho) = \left| \Pr[\mathrm{Exp}_A^{aut}[\kappa,\rho]=1] - \frac{1}{2} \right|$$

where the probability is taken over the choice of the track and trace system TK and the coin tosses of the adversary A.

Definition 2. An adversary A (ε,t,ρ)-breaks the authenticity of the track and trace system, if the advantage $\mathrm{Adv}_A^{aut}(k,\rho)$ of A in the experiment Exp_A^{aut} is at least ε and the running time of A is at most t.

Definition 3 Authenticity. A track and trace system is said to be (ε,t,ρ)-authenticated if there exists no adversary which can (ε,t,ρ)-break its authenticity.

3.2. Privacy

Informally, privacy means that an adversary should not be able to tell if a tag goes through some step v in the supply chain based on the data stored on the tag.

More precisely, the privacy definition is based on the following privacy experiment EXP_A^{prv} (*cf.*, Experiment 2). Let $O_{T,v}$ denote the oracle that picks a tag that goes through the step v. In the learning phase, A chooses a step v from the supply chain and is allowed to query the six oracles O_G, O_R, O_W, O_C, O_{CP}, and $O_{T,v}$ in any order. Then, in the challenge phase, the system randomly selects an uncorrupted tag $T_{ch} \in T$ (*i.e.*, A did not write into T_{ch}) and performs the **GoNext(** T_{ch} **)** operation to change T_{ch}'s internal state. A is given the tag T_{ch} and is asked to guess if T_{ch} has been through step v by outputting a bit b. In this phase, A is also to launch the six oracle queries under the restriction that it can query O_C of tag T_{ch}'s internal state. The total number of A's oracle queries does not exceed ρ.

Experiment 2. The privacy experiment.

Experiment $\mathrm{EXP}_A^{prv}[\kappa,\rho]$

(1) initialize the Tracker system through **Initialize(** κ **)**;
(2) A_1 chooses a step v
(3) $st \leftarrow A_1^{O_G,O_R,O_W,O_C,O_{CP},O_{T,v}}(\mathrm{TK})$; //*learning stage*
(4) choose randomly bit $b \in \{0,1\}$
(5) if $b=0$ then choose a tag $T_{ch} \in_R T$ that does not go through v,
else choose a tag $T_{ch} \in_R T$ which goes through v
(6) operate **GoNext(** T_{ch} **)**;
(7) $b' \leftarrow A_2^{O_G,O_R,O_W,O_C,O_{CP},O_{T,v}}(\mathrm{TK}, st, T_{ch})$; //*challenge stage*
(8) if $b=b'$ then output 1, 0 otherwise.

Definition 4. The advantage of adversary in the experiment $\mathrm{EXP}_A^{prv}[\kappa,\rho]$ is defined as:

$$\mathrm{Adv}_A^{prv}(\kappa,\rho) = |\Pr[\mathrm{Exp}_A^{prv}[\kappa,\rho]=1] - \frac{1}{2}|.$$

The probability is taken over the choice of track and trace system TK and the coin tosses of the adversary A.

Definition 5. An adversary A (ε,t,ρ)-breaks the privacy of the track and trace system, if the advantage $\mathrm{Adv}_A^{prv}(k,\rho)$ of A in the experiment Exp_A^{prv} is at least ε and the running time of A is at most t.

Definition 6 Privacy. A track and trace system is said to be (ε,t,ρ)-private if there exists no adversary that can (ε,t,ρ)-break its privacy.

Remark 1. Our privacy model is different from that of Blass *et al.* [1] in the choice of the challenge tag T_{ch}. In our model, T_{ch} is selected through a toss coin to decide whether it goes through the target step v or not; instead, T_{ch} is chosen uniformly at random from the tag set in the model of Blass *et al.* [1]. The privacy definition of [1] relies on the assumption that each tag goes through each step in the supply chains with the same probability. Unfortunately, it is easy to see that this assumption does not hold true in the supply chains. Furthermore, our privacy model allows inside attacks by providing O_{CP} queries to the adversary.

3.3. Unlinkability

Another two privacy requirements of the track and trace system are path unlinkability and tag unlinkability to prevent the adversary A from binding the tag data to its path and behavior, respectively. We give the detailed descriptions of them in the following.

3.3.1. Path Unlinkability

The privacy model of path unlinkability is depicted in the following experiment Exp_A^{pul} (*cf.*, Experiment 3). Let O_{T,P_i} denote the oracle that picks a tag that goes through the path P_i. In the learning phase, A chooses a tag T_0 from the supply chain and is allowed to query the six oracles O_G, O_R, O_W, O_C, O_{CP}, and O_{T,P_0} in any order, where $P_0 = \mathbf{Check}(T_0)$. Then, in the challenge phase, the system first selects a random bit $b \in \{0,1\}$. If $b = 0$ then it selects an uncorrupted tag $T_{ch} \in_R T$ that does not go through the path P_0; otherwise, it selects the tag $T_{ch} \in_R T$ that goes through the path P_0. Then, it performs the **GoNext(**T_{ch}**)** operation to change T_{ch}'s internal state. A is given the tag T_{ch} and is asked to guess if T_{ch} has been through the path P_0 by outputting a bit b. In this phase, A is also to launch the six oracle queries under the restriction that it can query O_C of tag T_{ch}'s internal state. The total number of A's oracle queries does not exceed ρ.

Experiment 3. The path unlinkability experiment.

Experiment $\mathrm{Exp}_A^{pul}[\kappa,\rho]$
(1) initialize the Tracker system through **Initialize**(κ);
(2) A_1 chooses a tag $T_0 \in T$; Let P_0 denote the path T_0 took;
(3) $st \leftarrow A_1^{O_G,O_R,O_W,O_C,O_{CP},O_{T,P_0}}(\mathrm{TK})$; //*learning stage*
(4) choose randomly bit $b \in \{0,1\}$
(5) if $b = 0$ then choose a tag $T_{ch} \in_R T$ that does not go through P_0, else choose a tag $T_{ch} \in_R T$ which goes through P_0
(6) operate **GoNext(**T_{ch}**)**;
(7) $b' \leftarrow A_2^{O_G,O_R,O_W,O_C,O_{CP},O_{T,P_0}}(\mathrm{TK}, st, T_{ch})$; //*challenge stage*
(8) if $b = b'$ then output 1; 0 otherwise.

Definition 7. The advantage of adversary A in the experiment $\mathrm{Exp}_A^{pul}[\kappa,\rho]$ is defined as:

$$\mathrm{Adv}_A^{pul}(\kappa,\rho) = \left| \Pr[\mathrm{Exp}_A^{pul}[\kappa,\rho] = 1] - \frac{1}{2} \right|.$$

The probability is taken over the choice of track and trace system TK and the coin tosses of the adversary A.

Definition 8. An adversary A (ε, t, ρ)-breaks the path unlinkability of the track and trace system, if the advantage $\mathrm{Adv}_A^{pul}(k,\rho)$ of A in the experiment Exp_A^{pul} is at least ε and the running time of A is at most t.

Definition 9 Path Unlinkability. A track and trace system is said to be (ε, t, ρ)-path-unlinkable if there exists no adversary that can (ε, t, ρ)-break its path unlinkability.

Remark 2. Our path unlinkability model is different from that of [1] in the choice of the challenge tag T_{ch}. In the path unlinkability model of [1], T_{ch} is chosen uniformly at random from the tag set. Such a model relies on the assumption that each tag goes through the P_0 with the same probability. However, this kind of assumption is not always true since some tags may never go through the path P_0. Hence, the path unlinkability model of [1] is incomplete for RFID-based track and trace systems. In our model, T_{ch} is selected through a toss coin to decide whether it goes through the target path P_0 or not. Our model avoids the abovementioned impractical assumption. Furthermore, our path unlinkability model allows inside attacks by providing O_{CP} queries to the adversary.

3.3.2. Tag Unlinkability

The privacy model of tag unlinkability is depicted in the following experiment Exp_A^{tul} (*cf.*, Experiment 4). In the learning phase, A chooses a tag T_0 from the supply chain and is allowed to query the five oracles O_G, O_R, O_W, O_{CP}, and O_C in any order. At the end of this phase, A outputs two tags (w.l.o.g., T_0 and T_1). Then, in the challenge phase, the system tosses a coin b and performs the **GoNext**(T_b) operation to update T_b's internal state. A is given the challenge tag T_b and is asked to guess the random bit b by outputting a bit b'. In this phase, A is also allowed to launch the five oracle queries under the restriction that it cannot query O_C about tag T_b's internal state. The total number of A's oracle queries does not exceed ρ.

Experiment 4. The tag unlinkability experiment.

Experiment $\mathrm{Exp}_A^{tul}[\kappa, \rho]$
(1) initialize the Tracker system through **Initialize**(κ);
(2) $\{T_0, T_1, st\} \leftarrow A_1^{O_G, O_R, O_W, O_C, O_{CP}}(\mathrm{TK})$; //*learning stage*
(3) $b \in_R \{0,1\}$; $T = T - \{T_0, T_1\}$;
(4) **GoNext**(T_b);
(5) $b' \leftarrow A_2^{O_G, O_R, O_W, O_C, O_{CP}}(\mathrm{TK}, st, T_b)$; //*challenge stage*
(6) if $b = b'$ then output 1, 0 otherwise.

Definition 10. The advantage of adversary A in the experiment $\mathrm{Exp}_A^{tul}[\kappa, \rho]$ is defined as:

$$\mathrm{Adv}_A^{tul}(\kappa, \rho) = |\Pr[\mathrm{Exp}_A^{tul}[\kappa, \rho] = 1] - \frac{1}{2}|,$$

where the probability is taken over the choice of track and trace system TK and the coin tosses of the adversary A.

Definition 11. An adversary A (ε, t, ρ)-breaks the tag unlinkability of the track and trace system if the advantage $\mathrm{Adv}_A^{tul}(k, \rho)$ of A in the experiment Exp_A^{tul} is at least ε and the running time of A is at most t.

Definition 12 Tag Unlinkability. A track and trace system is said to be (ε, t, ρ)-tag-unlinkable if there exists no adversary that can (ε, t, ρ)-break its tag unlinkability.

4. Relations among Privacy Models

In this section, we investigate the relations between privacy, path unlinkability, and tag unlinkability. Our results illustrate that tag unlinkability implies privacy, which is equivalent to the path unlinkability. Therefore, with respect to the security of track and trace systems, we only need to consider the authenticity and tag unlinkability, which will lead to simple schemes. More detailed explanations are as follows.

Theorem 1. (privacy $\Longleftrightarrow$ path unlinkability) In the track and trace system TK, the privacy model is equivalent to the path unlinkability model.

Proof. (1) privacy $\Rightarrow$ path unlinkability. Assume that TK is not path-unlinkable, *i.e.*, there exists an adversary A that can (ε, t, ρ)-break its path unlinkability. Then, we can use A as a subroutine to construct an algorithm B that can break the privacy of TK. The algorithm B simulates the experiment Exp_A^{pul} for A and is constructed as follows.

At first, when A submits the target tag T_a, B obtains the path P_a through $\mathbf{Check}(s_{T_a})$, where s_{T_a} is the internal state of T_a. Next, B chooses a step $v \in P_a$ and submits it to the privacy experiment as the target step. Then, B prepares the answers for A's as below. B answers O_R, O_W, O_C, O_{CP}, and O_G directly by querying them in the privacy experiment. If A asks a query of O_{T,P_a}, B chooses a tag T_i with initial state written by the issuer I and operates T_i that goes through the path P_a via the oracle query of O_G to the privacy experiment. Then, B returns T_i to A. Finally, in the challenge phase, B is given a challenge tag T_{ch}, which is forwarded to A as the challenge tag of experiment Exp_A^{pul}. If A outputs 1, then B also outputs $b' = 1$; else B outputs a bit $b' \in_R \{0,1\}$.

It is easy to see that B provides a perfect simulation of experiment Exp_A^{pul} for A. Let the advantage of A be ε. Now, we analyze the advantage of B.

$$\begin{aligned}\Pr[B \text{ succeeds}] &= \Pr[B \text{ succeeds} \wedge T_{ch} \text{ goes through } P_a] \\ &\quad + \Pr[B \text{ succeeds} \wedge T_{ch} \text{ does not go through } P_a] \\ &>= \tfrac{n}{|P_{valid}|}\varepsilon + \tfrac{1}{2}\end{aligned}$$

Hence, $\mathrm{Adv}_B^{prv} = |\Pr[B \text{ succeeds}] - \frac{1}{2}| \geq \frac{n}{|P_{valid}|}\varepsilon$.

(2) path unlinkability $\Rightarrow$ privacy. This can be inferred similarly to the method described in the above.

We have finished the proof of Theorem 1. □

Theorem 2. (tag unlinkability $\Rightarrow$ privacy) If the track and trace system TK is tag unlinkable then it is also private.

Proof. Assuming that TK is not private, *i.e.*, there exists an adversary A that can (ε, t, ρ)-break its privacy. Then, we use A as a subroutine to construct an algorithm B, which breaks the tag unlinkability of TK. The algorithm B simulates the experiment Exp_A^{prv} for A and proceeds as follows.

At first, when A submits the target step v, B selects two tags T_0 and T_1 such that T_1 goes through the step v but T_0 did not. Then, B answers A's oracle queries as below. B answers O_R, O_W, O_C, O_{CP}, and O_G directly by querying them in the privacy experiment. If A asks a query of $O_{T,v}$, B chooses a tag T_i with initial state setup by the issuer I and operates T_i to go through the step v via the oracle query of O_G to the privacy experiment. Then, B returns T_i to A. After the learning phase, B submits

T_0 and T_1 to its tag unlinkability experiment, which will return the challenge tag T_b to B. Finally, in the challenge phase, B deliveries T_b to A as its challenge tag T_{ch} of experiment Exp_A^{prv}. If A outputs b, then B also outputs b.

It is easy to see that B provides a perfect simulation of experiment Exp_A^{prv} for A and the advantage of B is just the same as that of A.

We have finished the proof of Theorem 2. □

The above two theorems illustrate that the tag unlinkability implies the privacy as well as the path unlinkability. Hence, with respect to the security of track and trace system, we only need to consider the authenticity and tag unlinkability, which simplifies the security concepts for the track and trace system.

Definition 13. A track and trace system of RFID-based supply chains is said to be secure if it is authenticated and tag unlinkable.

5. The Tracker+

In this section, we propose an efficient track and trace scheme Tracker+ for RFID-based supply chains. Specifically, no computational ability is required for tags in Tracker+, which implies that Tracker+ is totally compatible with EPC Class 1 Gen 2 standards. Although Blass *et al.* presented the track and trace scheme Tracker [1], it indeed cannot guarantee the claimed privacy since the adversary can trace a tag by comparing the deterministic signature part of its internal state with the history records. However, Tracker+ provides provable privacy even against supply chain inside attacks and is more efficient than Tracker.

5.1. Path Encoding

We use the same method of [1] to encode a path in the supply chain. Specifically, each path is represented by a number $v_p \in Z_q^*$ (where q is a big prime number, e.g., $|q|=160$), which has been derived from a polynomial determined by all steps in the path. Concretely, we associate each step v_i with a random number $a_i \in Z_q^*$ such that the numbers of all steps in a path can be used as the coefficients to construct a polynomial. W.l.o.g., let the path be $P_i = \{v_0, v_1, \cdots, v_n\}$, then the polynomial is

$$Q_{P_i}(x) = a_0 x^n + a_1 x^{n-1} + \cdots + a_{n-1} x + a_n = x Q_{p_{i-1}}(x) + a_n . \quad (1)$$

5.2. Multiple ElGamal Encryption and HMAC

Multiple ElGamal. Multiple ElGamal encryption is a variant of ElGamal encryption [17], which encrypts multiple messages under multiple public keys with the same randomness. Concretely, a multiple ElGamal encryption system **MEG=(PKG,Encrypt,Decrypt)** is as follows.

PKG. The public and private key generation algorithm, which selects the private key $x \in_R Z_q^*$ and computes the public key $y = g^x$, where g is the generator of a abelian group whose order is a big prime q.

Encrypt. The encryption algorithm, which inputs a pair of messages (m_1, m_2) and a pair of public key (y_1, y_2), selects a random number $k \in_R Z_q^*$ and computes $c_0 = g^k$, $c_1 = m_1 y_1^k$ and $c_2 = m_2 y_2^k$. The ciphertext is $c = (c_0, c_1, c_2)$.

Decrypt. The decryption algorithm, which inputs the ciphertext (c_0, c_1, c_2), computes $m_1 = c_1 / (c_0)^{x_1}$ and $m_1 = c_2 / (c_0)^{x_2}$, and returns (m_1, m_2).

HMAC. HMAC is a hashed MAC algorithm that can be used to generate authentication code. An HMAC function σ is defined as $\sigma(k, m) = h(k \oplus opad \,||\, h(k \oplus ipad) \,||\, m)$, where k refers to key, m refers to a message, and h refers to a hash function. For more details about *opad* and *ipad* see Krawczyk *et al.* [3].

5.3. Detailed Description of Tracker+

Intuitively, Tracker+ should consist of an initial setup phase, the preparation of new tags entering the supply chain, interactions between readers and tags, and the path verification by the manager M. However, all of these functions can be achieved via the five components of the track and trace system TK described in Section 2.2. Therewith, we only need to design the five components for Tracker+. The detailed description of Tracker+ is as follows.

Initialize(κ): Upon the security parameter κ, the system first prepares a supply chain $G = \{V, E\}$, a set of n tags T, a set of η readers R, a set of ν valid paths P_{valid}, and a set of valid state S_{valid}, and then it does as follows.

(1) Set up a multiple ElGamal public key encryption system [17] and generate the private keys $x_1, x_2 \in_R Z_q^*$ and the public keys $y_1 = g^{x_1}$ and $y_2 = g^{x_2}$, where g is the generator of group G whose order is a big prime q ($|q| = poly(\kappa)$);

(2) Set up an HMAC algorithm $\sigma : K \times \{0,1\}^* \to \{0,1\}^{160}$ [3] and choose η different keys $k_0, k_1, \cdots, k_{\eta-1}$ from the key space K;

(3) Select a generator x_0 of Z_q^* and η random numbers $a_0, a_1, \cdots, a_{\eta-1}$ from Z_q^*;

(4) Provide the issuer I with the tuple $(x_0, a_0, k_0, y_1, y_2)$ and each reader R_i with the tuple $(x_0, a_i, k_i, y_1, y_2)$;

(5) Provide the manager with the set $\{(x_0, a_0, k_0), \cdots, (x_0, a_{\eta-1}, k_{\eta-1})\}$, the private key (x_1, x_2), and public key (y_1, y_2);

Finally, the issuer I initializes each tag $T_i \in T$ by writing the tuple $(e_{i0}^0, e_{i1}^0, e_{i2}^0, \sigma_i^0)$ into it, where $e_{i0}^0 = g^{r_0}$, $e_{i1}^0 = y_1^{r_0} ID_i$, $e_{i2}^0 = y_2^{r_0} g^{a_0}$, $\sigma_i^0 = \sigma(k_0, ID_i)$, $r_0 \in Z_q^*$ is a random number, and ID_i is the identity of tag T_i. The manager M computes the path mark pmk_i for the valid path $P_{valid_i} = \{v_0, v_1, \cdots, v_\ell\}$ as

$$pmk_i = g^{a_0 x_0^\ell + \cdots + a_{\ell-1} x_0 + a_\ell}.$$

Then, M stores all the valid path marks and their corresponding path information into its database.

Read(T_i): Let the internal state of tag T_i be $s_{T_i}^j = (e_{i0}^j, e_{i1}^j, e_{i2}^j, \sigma_i^j)$. Then, return $s_{T_i}^j$.

Write(T_i): Let the tuple will be written into tag T_i be $s_{T_i}^{j+1} = (e_{i0}^{j+1}, e_{i1}^{j+1}, e_{i2}^{j+1}, \sigma_i^{j+1})$. Then store $s_{T_i}^{j+1}$ into tag T_i.

GoNext(T_i): When a tag T_i arrived at step v_{j+1} from step v_j, the reader R_{j+1} first reads out the internal of T_i through the operation $(e_{i0}^j, e_{i1}^j, e_{i2}^j, \sigma_i^j) = s_{T_i}^j \longleftarrow \text{Read}(T_i)$. Then it generates the new state $s_{T_i}^{j+1} = f(s_{T_i}^j)$ defined as follows.

Function $f(s_{T_i}^j)$

(1) Parse $s_{T_i}^j$ as $(e_{i0}^j, e_{i1}^j, e_{i2}^j, \sigma_i^j)$;

(2) Choose random number $r_{j+1} \in G_q^*$

(3) Compute $e_{i0}^{j+1} = (e_{i0}^j)^{x_0} g^{r_{j+1}}$, $e_{i1}^{j+1} = (e_{i1}^j)^{x_0} y_1^{r_{j+1}}$ and $e_{i2}^{j+1} = (e_{i2}^j)^{x_0} g^{a_{j+1}} y_2^{r_{j+1}}$;

(4) Compute $\sigma_i^{j+1} = \sigma(k_{j+1}, \sigma_i^j \| e_{i0}^{j+1})$;

(5) Return $s_{T_i}^{j+1} = (e_{i0}^{j+1}, e_{i1}^{j+1}, e_{i2}^{j+1}, \sigma_i^{j+1})$

End Function

Finally, reader R_{j+1} writes the state information $s_{T_i}^{j+1}$ into tag T_i.

Check($s_{T_i}^j$): At first, M parses $s_{T_i}^j$ as $(e_{i0}^j, e_{i1}^j, e_{i2}^j, \sigma_i^j)$ and decrypts the path mark

$$pmk_i = e_{i2}^j / (e_{i0}^j)^{x_2}.$$

Then, it searches the database to find the path mark pmk_i and its corresponding path information $\{(a_0, k_0), (a_1, k_1), \cdots, (a_j, k_j)\}$. If it does not find it then output $\varnothing$; otherwise, continue to verify the validation of the path signature as follows. Compute $e_{i0}^k = (e_{i0}^{k+1})^{\frac{1}{x_0}}$ for $k = j-1, \cdots, 1$, $ID_i = (e_{i1}^j / (e_{i0}^j)^{x_1})^{(\frac{1}{x_0})^j}$ and verify

$$\sigma_i^j = \sigma(k_j, (\cdots \sigma(k_1, \sigma(k_0, ID_i) \| e_{i0}^1) \cdots \| e_{i0}^j). \quad (2)$$

If the verification Equation (2) holds, then return the path $P_{valid_i} = \{v_0, v_1, \cdots, v_j\}$, else return $\varnothing$.

Remark 3. The internal state of Tracker+ is three group elements plus a HMAC code, while that in the original Tracker [1] is four group elements plus a HMAC code. Moreover, the HMAC is randomized in Tracker+ for every path so that its privacy can be guaranteed even in the presence of replay attacks, whereas the HMAC is fixed for every path in Tracker. Hence, it is easy to trace a tag simply by comparing the HMAC values stored in its memory, which implies that the privacy of Tracker can be broken without any difficulty. More detailed efficiency and security analysis will be demonstrated in Section 6.

6. Analysis

In this section, we first review the security definitions of HMAC and multiple encryption. Then, we prove the security of Tracker+. Our proofs illustrate that Tracker+ is provably secure against inside attacks. Finally, we evaluate the efficiency of Tracker+ and compare it with Tracker [1].

6.1. HMAC Security

Let O_H be an HMAC oracle that when it is provided with a message m, returns HMAC(m). The security of HMAC consists of two aspects:

(1) Existential Unforgeability under Adaptively Chosen Message Attacks (EUF-CMA): An adversary can launch oracle query O_H of n messages $m_1,\cdots,m_n$ to get the corresponding $HMAC(m_1),\cdots,HMAC(m_n)$ adaptively. Still, there is an advantage to A coming up with a new pair $(m^*, HMAC(m^*))$ where $m^* \neq m_i$ for $i=1$ to n is negligible.

(2) Indistinguishability: even the message m is known; an adversary A cannot distinguish $HMAC(m)$ from a random number, *i.e.*, the advantage of A is negligible.

6.2. Semantic Security

The semantic security of Multiple ElGamal is defined as follows. In the learning phase, an adversary is given the public key y_1 and y_2. Then it selects two message pairs (m_1^0, m_2^0) and (m_1^1, m_2^1), which have been submitted to the semantic security experiment. In the challenge phase, the adversary is given a ciphertext c^* and asked to guess which message pair is the plain text of c^*. Multiple ElGamal is said to be semantic secure if the probability that the adversary wins is at most $\frac{1}{2} + negligible$.

6.3. Security of Tracker+

The security of Tracker+ is guaranteed by the following Theorems 3, 4, and 5.

Theorem 3. If the HMAC function σ is EUF-CMA secure, then Tracker+ is authenticated.

Proof. Assume that Tracker+ is not authenticated, *i.e.*, there exists an adversary A such that it can break the authenticity of Tracker+. Then, we can construct a forger B to break the EUF-CMA security of HMAC function σ (whose key is k which is unknown to B). B uses A as a subroutine and answers A's queries as follows.

At first, B initializes the Tracker+ system in the same way as the **Initialize** operation except that the HMAC key of the manager is set to be k. It is easy to see that B can answer the queries of O_R, O_W, O_{CP}, and O_G (the arrived step is not M) directly. Upon a query of O_G with an arrived step of M, B reads out the internal state $(e_{i1}^j, e_{i1}^j, e_{i2}^j, \sigma_i^j)$ of tag T_i, and updates the three former group elements accordingly. Then, it asks the query $O_H(\sigma_i^j \,||\, e_{i1}^{j+1})$ to get $\sigma_i^{j+1} = \sigma(k, \sigma_i^j \,||\, e_{i1}^{j+1})$. Then, B answers the tuple $(e_{i0}^{j+1}, e_{i1}^{j+1}, e_{i2}^{j+1}, \sigma_i^{j+1})$ to A. The oracle O_C can be simulated similarly.

Obviously, if A is successful, then B is also successful.

At last, A outputs a tag T_i with internal state $s_{T_i}^j$. B first gets the path $P_i = \{v_0, v_1, \cdots, v_j\}$ and the tag's identity ID_i through the Check $(s_{T_i}^j)$ operation. Then, B computes

$$\sigma_i^{j-1} = \sigma(k_{j-1}, (\cdots \sigma(k_1, \sigma(k_0, ID_i) \,||\, e_{i0}^1) \cdots \,||\, e_{i0}^{j-1}).$$

Finally, B outputs the pair $(\sigma_i^{j-1} \,||\, e_{i0}^j, \sigma_i^j)$ as a forge for the HMAC function σ.

We have finished the proof of Theorem 3. □

Theorem 4. If the HMAC function σ is indistinguishable, then Tracker+ is tag unlinkable.

Proof. Assume that Tracker+ is not tag unlinkable, *i.e.*, there exists an adversary A such that it can break the tag unlinkability of Tracker+. Then, we can construct an algorithm B to break the semantic security of Multiple ElGamal encryption system, which has been proven secure under the Decisional

Diffie-Hellman (DDH) assumption. To this end, B uses A as a subroutine and maintains a list L to answer A's queries as follows.

Let the public key of Multiple ElGamal encryption cryptosystem be (y_1, y_2), whose corresponding private key is (x_1, x_2). At first, B initializes the Tracker+ system in the same way as the **Initialize** operation except that the public and private key pairs of the manager are implicitly set to be (y_1, y_2) and (x_1, x_2), respectively, where (x_1, x_2) are unknown to B. Moreover, B inserts the tuple $(ID_i, s_{T_i}^0)$ for $i = 1$ to n into list L. It is easy to see that B can answer the queries of O_R, O_W, O_{CP}, and O_G directly. For each O_G query, B also updates list L by inserting the pair $(ID_i, s_{T_i}^j)$ into L, so that it is able to answer the O_C queries by searching the list L to find the tag identity and the path. At the end of the learning phase, A outputs two tags T_0 and T_1. Algorithm B finds their identity and path mark pairs (ID_0, pmk_0) and (ID_1, pmk_1) by searching list L (using tag internal state as index).

In the challenge phase, B first submits the two message pairs (ID_0, pmk_0) and (ID_1, pmk_1) to the semantic security experiment of multiple ElGamal encryption system and gets the challenge ciphertext $c^* = (c_1, c_2, c_3)$. Then, B prepares the challenge tag T_{ch} for A as below. Choose a random bit b and set the internal state of T_b to be $(c_1, c_2, c_3, \sigma_j)$, where $\sigma_j = \sigma(k_j, \sigma_{j-1} \| c_1)$ and σ_{j-1} is the last part of the previous internal state of T_b. Set $T_{ch} = T_b$ and submit T_{ch} to A.

At last, A outputs a bit b'. If $b = b'$ then B outputs b, else B outputs a random bit.

Let the advantage of A be ε, then the advantage of B is at least $\frac{\varepsilon}{2}$ since B provides a perfect simulation for A if c^* is an encryption of T_b's identity and its current path.

We have finished the proof of Theorem 4. □

Theorem 5. If the HMAC function σ is EUF-CMA secure and indistinguishable, then Tracker+ is secure against inside attacks.

Proof. The proof of Theorem 5 can be inferred directly from Theorems 3 and 4. □

6.4. Efficiency and Comparisons

Efficiency Consideration. Tracker+ requires a tag only to store data. For each tag, only three group elements and a HMAC are required to be stored. If we choose the elliptic curve based multiple ElGamal encryption (where each element of group G is 160 bits) for Tracker+ and the output of HMAC is 160 bits, then the total storage requirement for each tag is 640 bits, which is feasible for EPC Class 1 Gen 2 tags.

Each reader in Tracker+ is required to store a tuple (x_0, a_i, k_i) and the manager's public key (y_1, y_2). Thus the total storage per reader is 800 bits. Regarding the computation, for each interaction between a tag and a reader, the reader needs to compute a multiple ElGamal encryption and HMAC evaluation. This is feasible for modern readers, which are more powerful than tags.

The manager M is responsible for the verification of the path that each tag goes through. To this end, manager M is required to decrypt the ciphertext stored in the tag and to verify the validity of the HMAC, which involves $j+3$ exponentiations and j HMAC evaluations. We conjecture that this is feasible for a powerful manager.

Compared to Tracker[1]. (*cf.* Table 1.) The storage of each tag in Tracker+ is 160 bits less than that of Tracker, which implies that Tracker+ saves 20% storage for tags. The computation costs for readers

and managers in Tracker+ are almost the same as those of Tracker. Secondly, Tracker+ has been proven to satisfy the privacy requirements of track and trace systems—privacy, path unlinkability, and tag unlinkability—whereas Tracker cannot guarantee the privacy requirements. Finally, Tracker+ has been proven to be secure against supply chain inside attacks, while Tracker is vulnerable to inside attacks. So, Tracker+ beats Tracker in both security and efficiency.

Table 1. Comparisons of Tracker and Tracker+.

	Storage requirement	**Privacy**	**Tag unlinkability**	**Path unlinkability**	**Inside attacks**
Tracker[1]	800 bits	N	Y	Y	N
Tracker+	640 bits	Y	Y	Y	Y

7. Conclusions

One of the major applications of RFID technology is the supply chain management. RFID tags have advantages over traditional barcodes in that they are able to provide real-time visibility, *etc.* Such visibility relies on the track and trace function of RFID-based supply chains. In this paper, we refined the privacy-related models of RFID-based track and trace systems to capture the security requirements of supply chains. Then, we clarified the relations among the three existing privacy related models. Our results simplify the privacy requirements of RFID-based supply chains and promise to produce efficient and simple privacy-preserving track and trace schemes. Finally, we proposed Tracker+, an efficient privacy-preserving track and trace scheme, which is compatible with EPC Class 1 Gen 2 tags and is provably secure against inside attacks.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (61272542; 61300204); the Fundamental Research Funds for the Central Universities (2013B07014); the Foundation for Distinguished Young Teachers in Higher Education of Guangdong under Grant No. Yq2013051; the Project of Science and Technology New Star of Guangzhou Pearl River (2014J2200006); the Natural Science Foundation of Guangdong (No. 2014A030313439); the Project of Science and Technology of Guangzhou City (No. S2013020011913); and the research project of the Department of Education of Guangdong Province (No. 2013KJCX0055).

Author Contributions

Xunjun Chen, Yuelong Zhu, and Jiguo Li designed the research; Xunjun Chen performed the research and analyzed the data; Xunjun Chen, Yamin Wen, and Zheng Gong provided the formal proofs. Xunjun Chen wrote the paper. All authors have read and approved the final manuscript.

Conflicts of Interest

The authors declare no conflict of interests.

References

1. Blass, E.-O.; Elkhiyaoui, K.; Molva, R. Tracker: Security and Privacy for RFID-based Supply Chains. In Proceedings of the 18th Annual Network and Distributed System Security Symposium (NDSS'11), San Diego, CA, USA, 2–9 February 2011; pp. 455–472.
2. EPCglobal. *EPC™ Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocols for Communications at 860 Mhz–960 Mhz*, Version 1.2.0; Epcglobal, Inc.: Lawrenceville, NJ, USA, 2008.
3. Krawczyk, H.; Bellare, M.; Canetti, R. *HMAC: Keyed-Hashing for Message Authentication*, *IETF RFC 2104*; IETF: Reston, VA, USA, 1997.
4. Juels, A. RFID Security and Privacy: A Research Survey. *IEEE J. Sel. Areas Commun.* **2006**, *24*, 381–394.
5. Avoine, G. RFID Security & Privacy Lounge. Available online: http://www.avoine.net/rfid/ (accessed on 5 June 2015).
6. Ateniese, G.; Camenisch, J.; de Medeiros, B. Untraceable RFID Tags via Insubvertible Encryption. In Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS'05), Alexandria, VA, USA, 7–11 November 2005; pp. 92–101.
7. Burmester, M.; van Le, T.; de Medeiros, B. Provably Secure Ubiquitous Systems: Universally Composable RFID Authentication Protocols. In Proceedings of the 2nd International Conference on Security and Privacy in Communication Networks (SecureComm'06), Baltimore, MD, USA, 28 August–1 September 2006; pp. 1–9.
8. Juels, A.; Weis, S. Authenticating Pervasive Devices with Human Protocols. In Proceedings of the 25th Annual International Cryptology Conference (CRYPTO'05), Santa Barbara, CA, USA, 14–18 August 2005; pp. 293–308.
9. Molnar, D.; Soppera, A.; Wagner, D. A Scalable, Delegatable Pseudonym Protocol Enabling Ownership Transfer of RFID Tags. In Proceedings of the 12th International Conference on Selected Areas in Cryptography (SAC'05), Kingston, ON, Canada, 11–12 August 2005; pp. 276–290.
10. Ohkubo, M.; Suzuki, K.; Kinoshita, S. Efficient Hash-Chain Based RFID Privacy Protection Scheme. In Proceedings of the 6th International Conference on Ubiquitous Computing (Ubicomp'04), Nottingham, UK, 11–14 September 2004.
11. Peris-Lopez, P.; Hemandez-Castro, J.C.; Estevez-Tapiador, J.M.; Ribagorda, A. LMAP: A Real Lightweight Mutual Authentication Protocol for Low-cost RFID tags. In Proceedings of the Workshop on RFID Security 2006 (RFIDSec'06), Graz, Austria, 12–14 July 2006.
12. Tsudik, G. YA-TRAP: Yet another Trivial RFID Authentication Protocol. In Proceedings of the 4th Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom'06), Pisa, Italy, 13–17 March 2006; pp. 640–643.
13. Weis, S.; Sarma, S.; Rivest, R.; Engels, D. Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. In Proceedings of the 1st International Conference on Security in Pervasive Computing (SPC'03), Washington, DC, USA, 12–14 March 2003; pp.454–469.
14. Ouafi, K.; Vaudenay, S. Pathchecker: An RFID Application for Tracing Products in Supply-Chains. In Proceedings of the Workshop on RFID Security 2009 (RFIDSec'09), Leuven, Belgium, 30 June–2 July 2009; pp. 1–14.

15. Li, Y.; Ding, X. Protecting RFID Communications in Supply Chains. In Proceedings of 2007 ACM Symposium on Information, Computer and Communications Security (ASIACCS'07), Singapore, 20–22 March 2007; pp. 234–241.
16. Ma, C.; Li, Y.; Deng, R.; Li, T. RFID Privacy: Relation between Two Notions, Minimal Condition, and Efficient Construction. In Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09), Chicago, IL, USA, 9–13 November 2009; pp. 54–65.
17. ElGamal, T. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory* **1985**, *31*, 469–472.