Toward Fast and Deterministic Clone Detection for Large Anonymous RFID Systems

Kai Bu* (corresponding author), Mingjie Xu*, Xuan Liu*, Jiaqing Luo•, Shigeng Zhang°
*College of Computer Science and Technology, Zhejiang University
*Department of Computing, The Hong Kong Polytechnic University
•School of Computer Science and Engineering, University of Electronic Science and Technology of China
°School of Information Science and Engineering, Central South University
*kaibu@zju.edu.cn, *edwin_xmj@163.com, *csxuanliu@comp.polyu.edu.hk, •csjluo@hotmail.com, °sgzhang@csu.edu.cn

***Abstract*—Cloning attacks seriously impede the security of Radio-Frequency Identification (RFID) applications. In this paper, we tackle deterministic clone detection for anonymous RFID systems without tag identifiers (IDs) as a priori. Existing clone detection protocols either cannot apply to anonymous RFID systems due to necessitating the knowledge of tag IDs or achieve only probabilistic detection with a few clones tolerated. We propose two protocols, BASE and DeClone, toward fast and deterministic clone detection for large anonymous RFID systems. BASE leverages the observation that clone tags make tag cardinality exceed ID cardinality. DeClone is built on a recent finding that clone tags cause collisions that are hardly reconciled through re-arbitration. For DeClone to achieve detection certainty, we design breadth first tree traversal toward quickly verifying unreconciled collisions and hence the cloning attack. We validate their detection performance through analysis and simulation. The results show that BASE delivers faster detection for small systems while DeClone for large ones especially when clone ratio increases.**

***Keywords*—RFID, Anonymity, Security, Clone Detection**

I. Introduction

Tag cloning attacks stand in the way of secure Radio-Frequency Identification (RFID) applications. Launching a cloning attack, an attacker compromises genuine tags and produces their replicas, namely *clone tags* [1], [2]. Since clone tags copy compromised genuine tags' data such as identifiers (IDs) and keys, they can impersonate genuine tags. Clone tags thus threaten RFID applications that use the genuineness of tags to validate the authenticity of tagged objects [3]–[5]. For example, attached with clone tags, products in RFID-enabled supply chains cause financial losses [6], while healthcare facilities in RFID-aided hospitals jeopardize personal safety [7]. However, cloning attacks are hard to prevent because most tags are resource constrained and cannot afford complex cryptographic solutions [3]. Leveraging the inherent variability in manufactured circuits, physically unclonable functions (PUFs) once ignited the hope of defeating cloning attacks for low-cost tags [3], [8]. Unfortunately, recent findings unveil PUFs' security vulnerabilities [9] and extinguish the hope of they preventing cloning attacks. We therefore need to detect whether clone tags hide in an RFID system.

Most existing clone detection protocols necessitate the knowledge of tag IDs. Such protocols are built on the ethos of knowing IDs before detecting which IDs associate with clone tags. Protocols in [6], [10]–[12] focus solely on RFID-enabled supply chains; they collect IDs from supply chain partners and detect the cloning attack when an ID simultaneously shows up at different places. Protocols in [4], [13], [14] write a new random number on a tag's memory each time the tag is scanned and maintain a map of tag IDs and corresponding random numbers. These protocols collect both tag IDs and random numbers from tags. A cloning attack is detected whenever any pair of the ID and random number is not identical with that in the map. A recent protocol in [15] further enhances the random number based approach to detect clone tags at a batch level. Besides, protocols in [16], [17] observe that responses from clone tags will turn supposed intact responses into collisions. Given tag IDs, they first determine when a tag to respond. During response collection, if the protocol expects only one response but receives a collision, it is likely that the current ID associates with more than one tag (i.e., one genuine plus at least one clone). In summary, all the preceding protocols share a belief that tag IDs are a must for clone detection.

Recent advances in clone detection start paying attention to anonymous RFID systems. In such systems, the knowledge of tag IDs should be isolated from protocol designs to enable privacy-sensitive applications [18]–[20]. Such a strict requirement renders ID-dependent protocols in [4], [6], [10]–[17] inapplicable to anonymous RFID systems. GREAT [20], to our best knowledge, is the only anonymous clone detection protocol. It shares the motivation with protocols in [16], [17]—clone tags induce unexpected collisions of tag responses. GREAT proposes reconciling collisions by greedily arbitrating tags that cause collisions. Since clone tags have the same IDs with that of their corresponding genuine tags, a collision caused by a genuine tag and its clone(s) cannot be reconciled. On the other hand, if a collision is due to multiple genuine tags, it might be reconciled after some rounds of arbitration. GREAT, however, adopts probabilistic arbitration protocol and therefore fits better in RFID applications with a few clones tolerable [20]. For applications requiring intact anonymity and authenticity, we still desire solutions that favor anonymous RFID systems yet provide deterministic clone detection.

In this paper, we take on the challenge of deterministic clone detection for anonymous RFID systems and contribute two protocols, BASE and DeClone. Both BASE and DeClone can accurately detect cloning attacks without tag IDs as a priori. First, BASE leverages cardinality contradiction caused by clone tags. The cardinality contradiction occurs between tag cardinality and ID cardinality because a cloning attack makes tag cardinality exceed ID cardinality. BASE, however, needs

to count almost all tags until it detects the cloning attack, although clone tags might respond at the very beginning of protocol execution. Second, inspired by the insight that clone tags induce unreconciled collisions [20], DeClone starts re-arbitrating collisions from the beginning of protocol execution and detects the cloning attack if an unreconciled collision is found. To verify unreconciled collisions, we propose breadth first tree traversal to quickly determine whether a collision is caused by tags with the same ID (i.e., a cloned ID) or different IDs. We validate BASE and DeClone's performance in terms of anonymity, accuracy, and scalability through analysis and simulation. The results show that BASE delivers faster detection for small systems while DeClone for large ones. Furthermore, the more cloned IDs a system contains, the faster DeClone detects the cloning attack.

The rest of the paper is organized as follows. Section II defines the problem. Section III and Section IV present BASE and DeClone, respectively, and analyze their performance. Section V reports performance-evaluation results. Finally, Section VI concludes the paper.

II. Problem Description and Preliminaries

In this section, we first describe the system model and adversary model. We then formulate the problem of deterministic clone detection for anonymous RFID systems.

A. System Model

Anonymous RFID system. We consider an anonymous RFID system in which readers cannot retrieve tag IDs from the server or tags. Such a strict requirement is necessary for protecting tag IDs' associated privacy and enabling privacy-sensitive applications [18]–[20]. Following a canonical RFID system architecture, the system consists of a backend server, a reader, and a number of objects each affixed with a tag. For inventory control, a tag has a unique ID across the system and represents the object that it is attached to. When multiple readers are in use for covering all tags and are well synchronized [21], they can be logically treated as one [22][1]. The reader communicates with the server via a wired or wireless yet secure channel while with tags via a wireless channel. Normally, the server does not directly communicate with tags; it dictates what monitoring operations the reader needs to execute over tags. Most existing monitoring operations entail the ID information of tags. For example, tag IDs are necessary for detecting missing tags whose IDs' correspond to no response [22] and for detecting clone tags whose IDs appear at different locations [6], [10]–[12]. However, we do not assume the knowledge of tag IDs in anonymous RFID systems. Only a tag itself knows its own ID; tags use IDs to determine when to respond under a certain anti-collision protocol (e.g., slotted Aloha [27] or tree traversal [28]).

Assumptions. To explore deterministic clone detection solutions for anonymous RFID systems, we assume that 1) ID cardinality (i.e., the number tag IDs) is known, and 2) opcodes are in use for readers to inform tags what data to transmit.

[1]Another line of research efforts customizes efficient monitoring protocols (e.g., cardinality estimation [23], information collection [24], and missing tag detection [25]) to multi-reader (or equivalently mobile handheld-reader [26]) systems. We believe that techniques therein can be leveraged to adapt our protocols; this is not the focus of this paper.

First, we assume the knowledge of ID cardinality rather than ID specifics. The reader can retrieve ID cardinality from the server. In an intact system, ID cardinality is equal to tag cardinality as each tag has a unique ID. However, when clone tags exist, tag cardinality exceeds ID cardinality. This observation motivates an intuitive clone detection solution (Section III).

Second, we assume that to fulfill various monitoring operations, the reader adopts opcodes to query corresponding data from tags. Such opcodes facilitate most established RFID protocols, whether or not being explicitly emphasized [29]. We observe that varying tag responses is necessary for protocol efficiency rather than protocol efficacy. Its purpose is to require as few data as possible from tags. For example, a 10-bit string with CRC embedded is sufficient for distinguishing an intact response from a collided one [30] while one bit is sufficient for transmitting binary information (e.g., tag presence [22], battery's residual capacity [31]). Transmitting more data than necessary therefore brings no benefits but efficiency degradation. For conciseness, we will directly indicate what data to transmit without emphasizing the use of opcodes during protocol design.

B. Adversary Model

Launching a cloning attack, the attacker first compromises genuine tags and then replicates their data to clone tags [2]. Since clone tags hold all valid data (e.g., IDs, keys) of compromised tags, clone tags can easily impersonate genuine tags. Detecting clone tags is thus practically important as most RFID applications equate tag genuineness to tagged objects' authenticity [2], [13]. However, clone tags are more challenging to detect than are counterfeit tags holding valid IDs but forged keys. Common solutions for detecting counterfeit tags such as cryptographic authentication [32]–[34] cannot conquer clone tags, which pass authentication using their copies of valid keys of compromised tags.

We assume that clone tags faithfully respond to reader queries. In other words, clone tags cannot selectively respond to evade detection. As with detecting counterfeit tags, a detection protocol fails to uncover clone tags if they keep silent to detection queries. There are two cases in which clone tags may violate the assumption of faithful response. We observe that in both cases the assumption could be satisfied with or without extra efforts. *First*, the attacker manipulates clone tags while eavesdropping the communication between the reader and tags. When the eavesdropped query is likely for clone detection, the attacker informs clone tags not to send responses. If this is the case, we can leverage advanced jamming techniques that can jam the attacker's communication without interfering the communication between the reader and tags [35], [36]. *Second*, clone tags may be made sophisticated enough such that they themselves decide to which queries not to respond. We tackle such sophisticated clone tags by making reader queries for clone detection indistinguishable from that for other monitoring operations. For example, when the clone detection protocol requires a 1-bit response from tags (Section IV), clone tags can hardly infer whether the query verifies its genuineness or its presence as in [22]. We suggest that strategically interweaving different monitoring operations

could obfuscate not only sophisticated clone tags but also sophisticated attackers; this is not the focus of this paper.

C. Problem Formulation

Consider an anonymous RFID system with ID cardinality N_{id} and at least one *cloned ID* that associates with some clone tag(s). The deterministic clone detection problem is to detect the existence of clone tags with certainty, as fast as possible, without tag IDs as a priori. Such a formulation enforces the following three requirements for designing desirable clone detection protocols.

Anonymity. To enable anonymous RFID systems (Section II-A), clone detection protocols cannot be built upon the values of tag IDs. Most existing clone detection protocols, however, require tag IDs as a priori [4], [11]–[14], [16], [17], [20] and thus provide few hints about detecting anonymous clone tags. We will explore new features for detecting clone tags while preserving tag anonymity (Section III and Section IV).

Accuracy. Clone tags are hardly distinguished from genuine tags by the reader and thus significantly threaten security-relevant applications such as RFID access control. For these applications, clone detection protocols should accurately ascertain the existence of clone tags if any. The only related work for detecting anonymous clone tags is, however, a probabilistic design [20]. We will propose new techniques for deterministic detection of anonymous clone tags (Section III and Section IV).

Scalability. RFID is experiencing an ever-increasing demand for more pervasive applications. For example, as Lux Research predicts, China's RFID card/tag market volume will expand to 2.11 billion units in 2017 [37]. Making clone detection protocols scalable to large-scale systems, we borrow ideas from related work (e.g., [4], [13], [16], [17], [20], [38]). The key idea is to collect as a few data from tags as sufficient for clone detection. This way, transmitting fewer data yields higher protocol efficiency and thus scalability.

III. BASELINE PROTOCOL USING CARDINALITY CONTRADICTION

In this section, we present a baseline protocol called *BASE*. Its intuition is that clone tags enlarge tag cardinality and therefore lead to a cardinality contradiction between tag cardinality and ID cardinality. We build BASE upon adapted slotted Aloha, whereby BASE can count tags without collecting their IDs. We also analyze BASE's performance and limitations.

A. Motivation: Cardinality Contradiction

Clone tags cause a contradiction between tag cardinality and ID cardinality. This observation is straightforward as a cloned ID associates with more than one tag. BASE leverages cardinality contradictions for clone detection. Given ID cardinality, BASE needs to obtain tag cardinality for comparison. To count tags, BASE chooses the communication protocol of slotted Aloha [27] over tree traversal [28] because slotted Aloha is faster than tree traversal in large systems [39]. Before we instantiate how BASE counts tags, we briefly introduce the basics of slotted Aloha. In slotted Aloha, the reader queries tags in a series of frames each containing a number of time slots. At the beginning of a frame, the reader issues two parameters, frame size f (i.e., the number of time slots) and random seed r. Upon receiving the query, a tag uniformly at random picks a time slot to respond—the tag determines the index of the picked time slot using a built-in hashing function $h(\cdot)$ with f, r, and its ID as operands. We regard the state of a time slot as *empty*, *singleton*, or *collision* if the time slot contains responses from none, only one, or at least two tags, respectively. At the end of a time slot, the reader 1) informs whether tags responded in the time slot need to participate in the subsequent frame and 2) triggers the next time slot. BASE can use slotted Aloha to count tags because the number of singletons faithfully estimates tag cardinality.

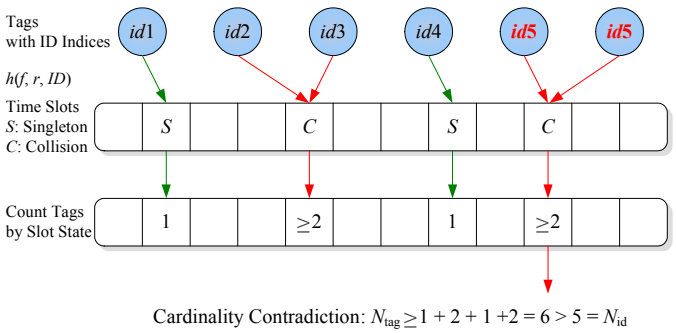

Fig. 1. BASE example. Clone tags make tag cardinality N_{tag} contradict (i.e., exceed) ID cardinality N_{id}. BASE thus uses a cardinality contradiction for detecting the existence of clone tags.

BASE example. Figure 1 instantiates how BASE detects clone tags using cardinality contradiction. The example considers an ID cardinality of $N_{id} = 5$ with one corresponding to a clone tag. For ease of presentation, we label tags with $id1$ through $id5$[2] and assign $id5$ as the cloned ID. In the query frame, $id1$ and $id4$ respond in singleton slots; $id2$ and $id3$ respond in a collision slot. The two $id5$'s with the same ID unavoidably respond in the same slot and cause another collision. Based on slot states, BASE increases the counter of tag cardinality N_{tag} as follows: an empty slot contributes no tag, a singleton slot only one, while a collision slot at least two. Two singleton slots and two collision slots as illustrated therefore yield a tag cardinality of $N_{\mathrm{tag}} \geq 1 \times 2 + 2 \times 2 = 6$. If all tags are genuine, they must correspond to $N_{\mathrm{tag}} \geq 6$ tag IDs. This contradicts with the given ID cardinality of $N_{\mathrm{id}} = 5$. Such a cardinality contradiction enables BASE to detect the existence of clone tags without knowing tag IDs.

B. BASE Design

BASE detects clone tags by finding cardinality contradiction between ID cardinality N_{id} and tag cardinality N_{tag}. Provided with ID cardinality, BASE counts tags using the number of singleton slots and maintains a rough cardinality estimation using the number of collision slots. The principle is that a singleton corresponds to only one tag while a collision at least two. Once tag cardinality is found to exceed ID cardinality can BASE detect the cloning attack.

[2]Note that $id1$ through $id5$ are only indices rather than the values of tag IDs. We do not assume ID specifics to guarantee the anonymity requirement.

BASE estimates tag cardinality N_{tag} by slightly adapting slotted Aloha based identification. The only adaptation is that tags do not respond with tag IDs as traditionally requested; instead, they respond with 10-bit strings embedded with CRC, which are sufficient for the reader to detect collisions [30]. Specifically, BASE runs some round(s) of slotted Aloha until it finds $N_{\text{tag}} > N_{\text{id}}$ (when clone tags exist) or $N_{\text{tag}} = N_{\text{id}}$ (otherwise). In each round, the reader issues a query message comprising frame size f and random seed r. For the first round, frame size f is initialized to N_{id} toward high time efficiency [27]. On receipt of the query, a tag waits to respond in the time slot with index $h(r, ID) \bmod f$. For tags in singleton slots, the reader informs them to keep silent until the end of BASE's execution. For tags in collision slots, the reader informs them to continue to respond in subsequent rounds. The reader maintains two slot-state related variants, n_{s} for the number of singleton slots and n_{c} for the number of collision slots. The value of n_{s} is initialized to zero and incremented upon singleton. The value of n_{c} is, however, set to zero at the beginning of each round and incremented upon collision. Using n_{s} and n_c, BASE then estimates N_{tag} as the following:

$$N_{\text{tag}} \geq n_{\text{s}} + 2n_{\text{c}} \tag{1}$$

At the end of each round, BASE compares tag cardinality N_{tag} against ID cardinality N_{id} and accordingly executes one of the following three cases.

- If $N_{\text{tag}} > N_{\text{id}}$ (i.e., $n_{\text{s}} + 2n_{\text{c}} > N_{\text{id}}$) as in Figure 1, BASE detects the cloning attack and terminates.
- If $N_{\text{tag}} = N_{\text{id}}$ (i.e., $n_{\text{s}} = N_{\text{id}}$) and $n_{\text{c}} = 0$, BASE verifies the genuineness of all tags and terminates.
- If $N_{\text{tag}} \leq N_{\text{id}}$ (i.e., $n_{\text{s}} + 2n_{\text{c}} \leq N_{\text{id}}$) and $n_c > 0$, BASE proceeds to the next round with an updated frame size of $f = N_{\text{id}} - n_{\text{s}}$.

C. Performance Analysis

BASE satisfies all the three performance requirements of anonymity, accuracy, and scalability as described in Section II-C. We sketch the performance analysis as follows.

Anonymity. BASE preserves the anonymity of all tag IDs except that of cloned IDs. During the execution, BASE ensures that only tags themselves use the knowledge of tag IDs. BASE neither transmits tag IDs in the air nor grants access to tag IDs on the server. The anonymity of cloned IDs is, however, breached by the cloning attacker but not by BASE. Since cloned IDs are clear to the attacker anyway, the best a clone detection protocol can do is to protect the anonymity of tag IDs other than cloned IDs. We therefore conclude that BASE satisfies the anonymity requirement.

Accuracy. BASE satisfies the accuracy requirement in that it has neither false positive nor false negative. *First*, a false positive occurs when no clone tags exist while BASE reports a cloning attack. When no clone tags exist, tag cardinality is equal to ID cardinality as each tag has a unique ID. In this case, BASE ascertains the genuineness of all tags with no false positives. *Second*, a false negative occurs when clone tags exist while BASE does not detect their existence. When clone tags exist, a compromised genuine tag and its corresponding clone tags will always respond in the same slot and cause a collision (Figure 1). Such a collision corresponds to only one ID but contributes more than one to tag cardinality estimation (Equation 1). This way, clone tags make tag cardinality exceed ID cardinality. BASE thus detects the cloning attack with no false negatives.

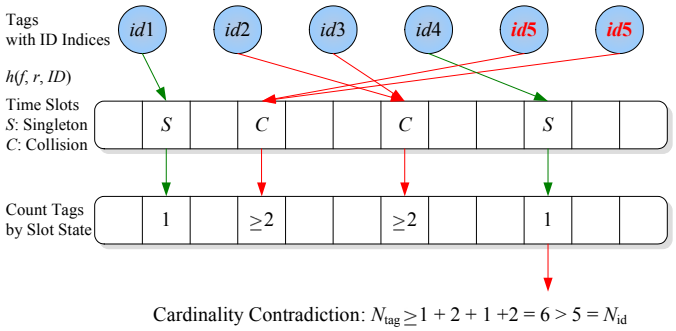

Fig. 2. BASE's efficiency limitation. Clone tags with $id5$ may respond at the early stage of BASE execution (e.g., the fourth slot as illustrated) whereas BASE detects their existence at the ending stage.

Scalability. BASE satisfies the scalability requirement with a linear time complexity with respect to system scale. When frame size f is set to the number of IDs of tags that will respond, the frame contains approximately f/e empty slots, f/e singleton slots, and $(1-2/e)f$ collision slots (by Lemma 1 in [30]). As aforementioned, when the number of cloned IDs is limited, the distribution of slot slates slightly varies. The first frame of BASE execution thus contains N_{id}/e empty slots, N_{id}/e singleton slots, and $(1-2/e)N_{\text{id}}$ collision slots, leaving $(N_{\text{id}} - N_{\text{id}}/e)$ IDs to the second frame. As the second frame size is set to $N_{\text{id}} - N_{\text{id}}/e$, the distribution of slot states follows the same ratios as that in the first frame; this applies to also subsequent frames. The total number of time slots of all frames is about eN_{id} [27], comprising N_{id} empty slots, N_{id} singleton slots, and $(e-2)N_{\text{id}}$ collision slots. Let T_{e}, T_{s}, and T_{c} denote the time duration of an empty slot, a singleton slot, and a collision slot, respectively. BASE's execution time, T_{BASE}, can be defined as the following:

$$\begin{aligned} T_{\text{BASE}} &\approx T_{\text{e}}N_{\text{id}} + T_{\text{s}}N_{\text{id}} + (e-2)T_{\text{c}}N_{\text{id}} \\ &= (T_{\text{e}} + T_{\text{s}} + (e-2)T_{\text{c}})N_{\text{id}}. \end{aligned} \tag{2}$$

BASE therefore has a linear time complexity of $\mathcal{O}(N_{\text{id}})$ and is scalable to large systems.

D. Limitations and Lessons Learned

BASE catches cardinality contradiction until most tags respond in singleton slots whereas clone tags may respond at early stage of BASE execution. As illustrated in Figure 2, clone tags with $id5$ respond in the fourth slot. The cardinality contradiction, however, shows up until the last third slot. More generically, when BASE executes multiple frames, clone tags respond in each frame and BASE reveals their caused cardinality contradiction at the ending slots of the last frame. If we could find a way to expose clone tags in each slot, the clone detect protocol can terminate immediately after it detects clone tags. Such a protocol promises faster detection in, for example, the fourth slot in Figure 2 than BASE does

in the tenth slot. We next present DeClone that enforces clone detection in each slot.

IV. DeClone Protocol using Unreconciled Collision

In this section, we present *DeClone*. It leverages the observation that clone tags cause unreconciled collisions, which cannot be arbitrated into singletons. We use a hybrid design of slotted Aloha and tree traversal to verify unreconciled collisions. We also propose breadth first tree traversal toward fast unreconciled collision verification. Our analysis shows that DeClone satisfies the requirements of accuracy and scalability whereas it leaks the IDs of compromised tags.

A. Motivation: Unreconciled Collision

Observations from BASE. We further distill the lessons from BASE into the following two observations that drive DeClone design.

First, clone tags cause *unreconciled collisions* that cannot be arbitrated into singletons. However, since slotted Aloha is a probabilistic arbitration protocol, it cannot deterministically detect an unreconciled collision. Specifically, slotted Aloha cannot determine whether a hard-to-arbitrate collision is due to clone tags or genuine tags with different IDs, although a collision against more rounds of arbitration yields a higher probability of being unreconciled due to clone tags [20].

Second, because of the probabilistic essence of slotted Aloha, a clone tag is likely to respond in each time slot with equal probability. Finding unreconciled collisions from the very beginning of a frame, therefore, promises faster detection of clone tags than does BASE, which exposes the existence of clone tags until almost all tags respond in singleton slots.

Motivations for DeClone. Motivated by the preceding two observations, we design DeClone by leveraging unreconciled collisions. We adopt the following two techniques to guarantee certainty and efficiency of clone detection.

First, we deterministically detect unreconciled collisions through a hybrid design of slotted Aloha and tree traversal. Different from slotted Aloha, tree traversal performs deterministic arbitration. In **tree traversal** [28], to collect l-bit tag IDs, the reader first creates a binary tree of height l and with each l-bit string mapped to a leaf. The reader then collects tag IDs through traversing the binary tree in a depth first order. Specifically, the reader broadcasts the bit string corresponding to the current tree node; tags respond if their IDs are prefixed with the bit string. If no collision occurs, the reader can correctly receive the response. Otherwise, the reader continues to collect tag IDs by broadcasting the bit string of the current tree node's child. This way, at the lth level of the binary tree, the bit string a tree node corresponds to an l-bit tag ID. If the query using an l-bit tag ID still incurs a collision of tag responses, it must be an unreconciled collision caused by the genuine tag and clone tag(s) with the queried ID. DeClone first arbitrates a collision using slotted Aloha toward quickly finding a possible unreconciled collision; then it further arbitrates the collision using tree traversal to assert the unreconciled collision. We will adapt tree traversal to breadth first traversal toward fast detection of unreconciled collisions. (Figure 3 and Section IV-B).

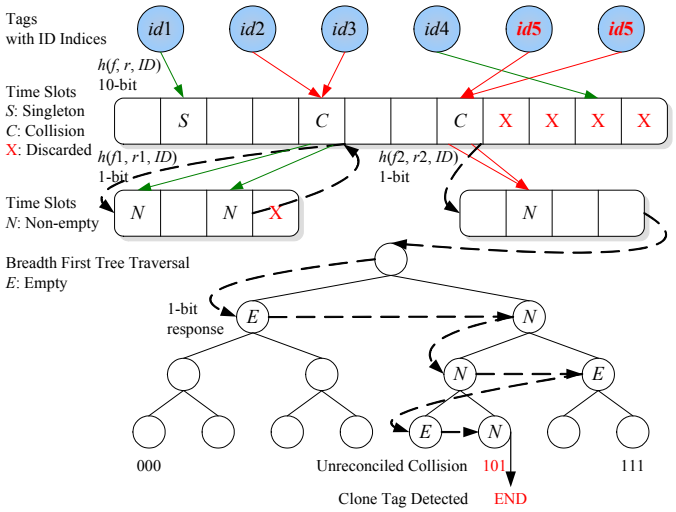

Fig. 3. DeClone achieves deterministic clone detection through a hybrid design of slotted Aloha and tree traversal. Clone tags cause unreconciled collisions. DeClone first uses slotted Aloha to quickly find a possible unreconciled collision (e.g., the collision caused by the two tags marked with $id5$) and then further verifies it using breadth first tree traversal.

Second, from the very beginning of each frame, DeClone uses a hybrid design of slotted Aloha and tree traversal to arbitrate collisions. Such design yields faster clone detection than does BASE when system scale becomes relatively large (e.g., over 1,000 tags, Section V).

DeClone Example. Figure 3 illustrates how DeClone deterministically detects clone tags in an anonymous RFID system. In accordance with Figure 1 and Figure 2, we label tags with $id1$ through $id5$ for ease of presentation instead of assuming their IDs to be known. The two tags marked with $id5$ consist of a genuine tag and a clone tag. DeClone starts with slotted Aloha and issues a frame of time slots in which tags will randomly respond. For an empty or singleton slot, the reader follows conventional slotted Aloha. For example, tag $id1$ responds in a singleton slot, the reader acknowledges it to keep silent and proceeds to the next slot. For a collision slot, the reader verifies whether it is an unreconciled collision before starting the next slot. Specifically, the reader issues a new frame to further arbitrate tags that responded in the collision slot. For example, tag $id2$ and tag $id3$ respond in a collision slot. The reader further arbitrates them in a new frame with $f1$ slots. After detecting two non-empty slots that are sufficient for negating an unreconciled collision, the reader immediately terminates the new frame and traces back to the original frame. If the new frame contains only one non-empty slot, the reader detects a possible unreconciled collision and then switches to tree traversal for verification. Take the collision caused by genuine tag $id5$ and clone tag $id5$ for example. Since the two tags have the same ID, they always simultaneously respond whether slotted Aloha or tree traversal is in use. By the end of tree traversal, the reader detects an unreconciled collision caused by tags with ID 101 and hence the cloning attack. (We use 3-bit tag IDs for ease of presentation.) Note that instead of adopting traditional depth first tree traversal, we propose breadth first tree traversal toward faster detection of unreconciled collisions. We will next detail the design specifics of DeClone.

B. DeClone Design

As Figure 3 depicts, DeClone consists of two key components, *unreconciled collision detection* and *unreconciled collision verification*. Unreconciled collision detection quickly finds possible unreconciled collisions through re-arbitrating collisions using slotted Aloha. A possible unreconciled collision still yields a collision after re-arbitration (e.g., the collision caused by two $id5$'s in Figure 3). For determining whether a possible unreconciled collision really is unreconciled, the component of unreconciled collision verification further arbitrates tags that cause the possible unreconciled collision using breadth first tree traversal. Since unreconciled collisions are attributable to clone tags, DeClone can detect cloning attacks with certainty by the evidence of unreconciled collisions. We next detail DeClone design with focus on the two components.

Unreconciled collision detection. In this component, we re-arbitrate collisions using slotted Aloha to find possible unreconciled collisions. Specifically, the reader first issues frame size f and random seed r. Tags then use f and r to derive in which time slot they respond. The response is a 10-bit random string with CRC embedded for distinguishing singletons from collisions. For empty and singleton slots, the reader follows conventional slotted Aloha. For a collision slot, the reader does not directly trigger next time slot. Instead, the reader re-arbitrates tags responded in the collision slot using slotted Aloha with re-arbitration frame size f_{r} and another random seed. The response is of length 1 bit for distinguishing empties from non-empties. Take the first collision caused by $id2$ and $id3$ in Figure 3 for example. If the re-arbitration yields more than one non-empty slot, we make sure that the collision is not unreconciled, trace back to the original frame, and proceed to the next slot. However, if the re-arbitration yields only one non-empty slot (e.g., the collision by two $id5$'s in Figure 3), the collision is likely unreconciled. The possibility of it being really unreconciled depends on the value of re-arbitration frame size f_{r}; we formulate it in Theorem 1 [20].

Theorem 1: Given a possible unreconciled collision with re-arbitration frame size f_{r}, the probability $P_{\mathrm{UC}}(f_{\mathrm{r}})$ of it being really unreconciled (i.e., caused by tags with the same ID) is lower bounded by the following:

$$P_{\mathrm{UC}}(f_{\mathrm{r}}) \geq 1 - \frac{1}{f_{\mathrm{r}}^2}.$$

Proof: We first derive the probability $P_{\mathrm{D}}(f_{\mathrm{r}})$ of the collision caused by tags with different IDs. The we have $P_{\mathrm{UC}}(f_{\mathrm{r}}) = 1 - P_{\mathrm{D}}(f_{\mathrm{r}})$. Let N_{PUC} denote the number of IDs of tags that cause the possible unreconciled collision. $P_{\mathrm{D}}(f_{\mathrm{r}})$ is equal to the probability that all the tags with $N_{\mathrm{PUC}} \geq 2$ different IDs respond in one of the f_{r} slots.

$$P_{\mathrm{D}}(f_{\mathrm{r}}) = \sum_{i=0}^{f_{\mathrm{r}}-1} \frac{1}{f_{\mathrm{r}}} \frac{1}{f_{\mathrm{r}}^{N_{\mathrm{PUC}}}} = \frac{1}{f_{\mathrm{r}}^{N_{\mathrm{PUC}}}} \leq \frac{1}{f_{\mathrm{r}}^2}. \tag{3}$$

We thus derive $P_{\mathrm{UC}}(f_{\mathrm{r}})$ as follows:

$$\begin{aligned} P_{\mathrm{UC}}(f_{\mathrm{r}}) &= 1 - P_{\mathrm{D}}(f_{\mathrm{r}}) \\ &\geq 1 - \frac{1}{f_{\mathrm{r}}^2}. \end{aligned}$$

The second line substitutes $P_{\mathrm{D}}(f_{\mathrm{r}})$ with Formula 3. ■

Following BASE design, DeClone initializes frame size f to N_{id} and decreases it by the number of singletons in the ith round for the $(i+1)$th round, where $i \geq 1$. This is because DeClone also desires a high ratio of singletons in a frame. For a singleton slot, if its corresponding tag has been cloned, the singleton will turn to an unreconciled collision. Therefore, the higher ratio of singletons we expect from a frame, the higher probability of at least one of the expected singletons becoming an unreconciled collision. This further yields a higher probability of clone detection in the frame.

Unreconciled collision verification. In this component, we re-arbitrate the possible unreconciled collision using breadth first tree traversal to verify unreconciled collision. We design breadth first tree traversal toward faster execution as follows. The reader traverses the binary tree by broadcasting the ID prefix corresponding to the node being traveled. Tags matching the broadcast ID prefix respond with 1-bit responses as we need only two response states of empty and non-empty. *For a non-leaf level*, the reader travels in a breadth first search fashion until it detects two non-empties. Since two non-empties indicate that the possible unreconciled collision is not really unreconciled, the reader traces back to the original frame and triggers the next slot. However, if the level contains only one non-empty node, the reader continues traversing the subtree rooted at the non-empty node. *For the leaf level*, the reader still traces back to the original frame upon detecting two non-empties. If the leaf level contains only one non-empty, the reader verifies an unreconciled collision, detects the cloning attack, and terminates the protocol. In summary, breadth first tree traversal is fast because the reader travels only two nodes at each level; this property can also be drawn from Figure 3.

C. Performance Analysis

For the requirements described in Section II-C, DeClone guarantees accuracy and scalability but might leak a few tag IDs with slight probability. We sketch the analysis as follows.

Anonymity. DeClone cannot preserve the anonymity of cloned IDs either (as in Section III-C) and might leak a small number of tag IDs with a controllable, slight probability. DeClone leaks two tag IDs when 1) their corresponding tags simultaneously respond in both the original frame and the re-arbitration frame, and 2) their corresponding leaves have the same parent in the binary tree. If condition 1 is not satisfied, the two tag IDs do not enter tree traversal and thus are protected. By Theorem 1, the probability of two different tag IDs falling into the same slot can be very low under a certain f_{r} (i.e., $1/f_{\mathrm{r}}^2$). Furthermore, condition 2 is also rarely satisfied. Only when the two tag IDs are consecutive and have the same parent will DeClone need to broadcast them for tree traversal. If the two tag IDs' corresponding leaves have different parents, DeClone will detect two non-empties on the second highest level as latest and trace back to the original frame. In this more common case, DeClone does not enter the leaf level or leak tag IDs.

Accuracy. DeClone satisfies the accuracy requirement with neither false positive nor false negative. *First*, when no clone tags exist, no unreconciled collisions will occur. DeClone thus reports no cloning attack and leads to no false positives. *Second*, when clone tags exist, they will cause unreconciled collisions. According to DeClone design, once a cloned ID picks a

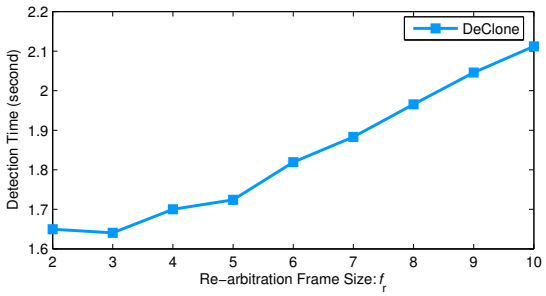

Fig. 4. DeClone detection time with ID cardinality N_{id}=1,000, one cloned ID, and varying re-arbitration frame size f_{r}.

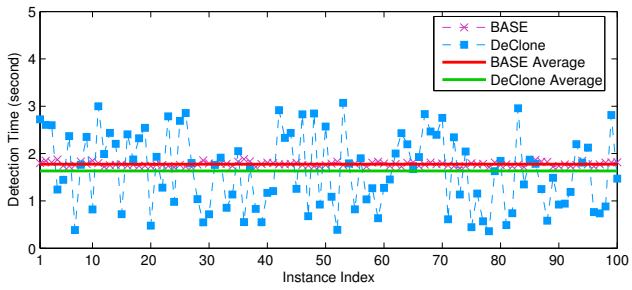

Fig. 5. Detection time fluctuation with N_{id} =1,000 tag IDs including a cloned one.

time slot without other contending IDs, its caused unreconciled collision can be verified by DeClone. It is straightforward that an unreconciled collision must occur within a sufficiently a large number of rounds of arbitration. DeClone thus detects cloning attack without false negatives.

Scalability. DeClone satisfies the scalability requirement with a linear time complexity with respect to system scale. DeClone terminates when it detects an unreconciled collision, which is supposed to be a singleton without clone tags. Intuitively, more cloned IDs yield faster clone detection. We now analyze the case with only one cloned ID among N_{id} ones. Since DeClone sets frame size f to be the number of tag IDs to count, each frame contains approximately f/e expected singletons. The probability of detecting cloning attack in the first frame, that is, the probability of the cloned ID corresponding to one of the expected singletons. is therefore $1/e$. The probability p_i of DeClone detecting cloning attack in the ith frame is defined as the following:

$$p_i = \frac{1}{e}(1-\frac{1}{e})^{i-1}. \tag{4}$$

With frame size initialized to N_{id} and reduced by a ratio about $1/e$, the frame size f_i for the ith round is as the following:

$$f_i = N_{\mathrm{id}}(1-\frac{1}{e})^{i-1}. \tag{5}$$

As discussed in Section III-C, f_i slots consist of f_i/e empties, f_i/e singletons, and $(1-2/e)f_i$ collisions on average. An empty and a singleton spans T_{e} and T_{s}, respectively. For most collisions, it takes T_{c} for collision detection and at most $f_{\mathrm{r}}T_{\mathrm{e}}$ for re-arbitration. For only a few collisions, DeClone starts tree traversal and rarely enters the leaf level. We can amortize the additional time cost by tree traversal to each collision slot with a small constant ϵ.

We approximate the execution time of DeClone as follows:

$$\begin{aligned} T_{\mathrm{DeClone}} &= \sum_{i\geq 1} p_i(\frac{f_i}{e}T_{\mathrm{e}} + \frac{f_i}{e}T_{\mathrm{s}} + (1-\frac{2}{e})f_i(T_{\mathrm{c}} + f_{\mathrm{r}}T_{\mathrm{e}} + \epsilon)) \\ &\approx C(\frac{(1-\frac{1}{e})^2}{2-\frac{1}{e}} + \frac{1}{e})N_{\mathrm{id}}, \end{aligned} \tag{6}$$

where $C = \frac{1}{e}T_{\mathrm{e}} + \frac{1}{e}T_{\mathrm{s}} + (1-\frac{2}{e})(T_{\mathrm{c}} + f_{\mathrm{r}}T_{\mathrm{e}} + \epsilon)$ and the second line substitutes p_i and f_i by Equation 4 and Equation 5, respectively. DeClone thus has a linear time complexity of $\mathcal{O}(N_{\mathrm{id}})$ and satisfies the requirement of scalability.

V. Performance Evaluation

In this section, we validate the performance of BASE and DeClone through simulations. As Section I reviews, most existing clone detection protocols (e.g., in [4], [6], [10]–[17]) require known tag IDs and cannot apply to anonymous RFID systems. Furthermore, the only existing protocol in [20] for anonymous RFID systems cannot detect clone tags with certainty. In comparison with the prior art, BASE and DeClone are the first line of protocols toward deterministic clone detection for anonymous RFID systems. We thus evaluate their performance without comparing against that of existing protocols. As the results will show, both BASE and DeClone can accurately detect the existence of clone tags.

A. System Configuration

We simulate the system and attack according to the model in Section II-A and Section II-B. Following the C1G2 standard [40] and Philips I-CODE specification [41], a tag has a 96-bit ID and a reader takes 0.4 ms to distinguish an empty from a nonempty and 0.8 ms a singleton from a collision. That is, we have $T_{\mathrm{e}} = 0.4$ ms and $T_{\mathrm{s}} = T_{\mathrm{c}} = 0.8$ ms for Equation 2 and Equation 6. For DeClone, tree traversal issues queries of varying-length ID prefixes; we approximate the average transmission time for a single bit to 25 µs [1]. Re-arbitration frame size f_{r} of DeClone is also worthy of discussion. If f_{r} is too small, collisions caused by tags with different IDs may frequently lead to possible unreconciled collisions, which further invoke tree traversal and increase detection time. On the other hand, if f_{r} is too large, collisions caused by tags with different IDs can be highly likely reconciled to at least two nonempties and will not invoke tree traversal. A large re-arbitration frame, however, does increase detection time as well. We empirically choose f_{r} for DeClone execution. Figure 4 instantiates f_{r} selection when the ID cardinality is N_{id} =1,000 with one being cloned. The detection time is averaged over 10,000 runs. We observe that f_{r} =3 yields a faster detection on average than do other f_{r}'s. We thus select f_{r} =3 for N_{id} =1,000 and follow the same way to select f_{r} for other execution instances.

B. Detection Time Fluctuation

We first report fluctuation trend of the detection time. Per protocol designs, BASE detects clone tags until almost all genuine tags are counted whereas DeClone depends on when a cloned ID picks an expected singleton slot. The detection time of BASE is thus relatively stable. In comparison, the detection

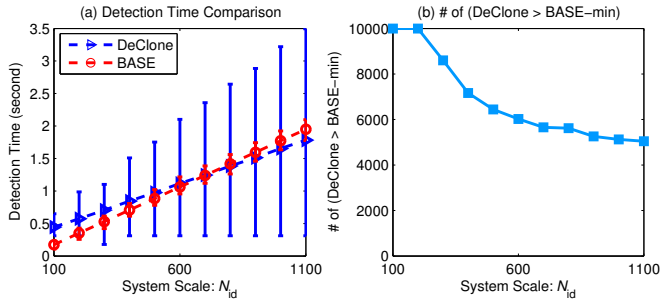

Fig. 6. Detection time comparison with relatively small system scales with only one cloned ID.

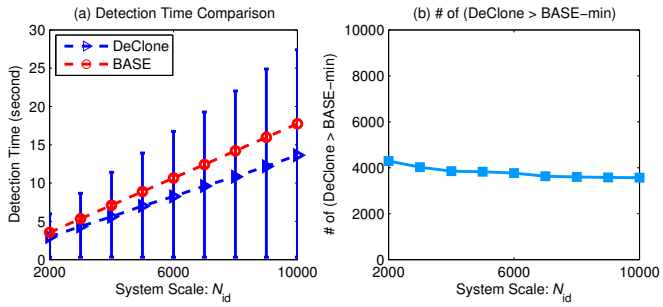

Fig. 7. Detection time comparison with large system scales with only one cloned ID.

time of DeClone fluctuates more because a tag picks each slot in a frame with similar probability. That is, if the cloned ID chooses a starting yet expected singleton slot, DeClone might detect the cloning attack very quickly. Otherwise, if the cloned ID falls to an ending slot of the frame, DeClone takes a longer detection time.

Figure 5 reports detection time of 100 successive instances of BASE and DeClone. All instances share a set of N_{id} =1,000 tag IDs with one being cloned. As we expect, BASE delivers a relatively stable detection time ranging from 1.68 seconds to 1.88 seconds. The detection time of DeClone, however, varies a lot, ranging from 0.36 seconds to 3.07 seconds. Out of the 100 runs, 49 runs of DeClone take more time than the fastest BASE instance. For the average detection time, DeClone takes 1.63 seconds and outperforms BASE, which takes 1.78 seconds. It indicates that DeClone is more favorable for frequent detection while BASE is more desirable for some critical checking points that solicit predictable detection time. As later results will show, the gap of average detection time broadens with system scale and clone ratio; the case of DeClone being slower than BASE becomes less frequently and vanishes as the number of cloned IDs increases.

C. Varying System Scale

We now evaluate the performance of BASE and DeClone under varying system scale. We expect that DeClone delivers faster detection for large systems while BASE might be faster for small systems. An extreme case is when ID cardinality N_{id} =1 and the only ID is cloned. In this case, BASE can detect cardinality contradiction and hence the cloning attack using only one time slot. For DeClone, besides the one time slot in the original frame, it takes an extra re-arbitration frame as well as tree traversal, costing more time for detection than does BASE.

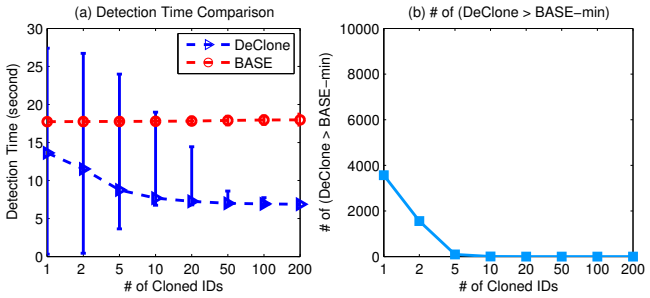

Fig. 8. Detection time with tag cardinality N_{id} =10,000 including a varying number of cloned IDs.

Figure 6 and Figure 7 respectively report the results for relatively small systems (N_{id} from 100 to 10,000) and relatively large ones (N_{id} from 2,000 to 10,000). We still focus on the one–cloned-ID scenario. Both BASE and DeClone guarantee a linear average detection time with respect to system scale. Since DeClone has a less stable detection time (Figure 5), we report also the number of DeClone runs with longer detection time than the minimum of BASE runs among 10,000 runs in Figure 6(b) and Figure 7(b). As shown in Figure 6, when the system scale is as small as 100, all 10,000 BASE runs outperform DeClone. When the system approaches 800, DeClone starts delivering a lower average detection time than that of BASE, although having over 4,000 runs beaten by BASE. As the system scales up thereafter, the average detection time gap between BASE and DeClone widens and the case when DeClone is slower than BASE becomes less frequently. Another interesting observation is that when only one cloned ID exists, detection time of 10,000 DeClone runs per system scale follows a uniform distribution. This explains why the average detection time of DeClone approximates to half of the maximum plus the minimum.

D. Varying Clone Ratio

We further evaluate the performance of BASE and DeClone under varying clone ratio. Figure 8 reports average detection time of BASE and DeClone over 10,000 runs with ID cardinality N_{id} =10,000 and varying number of cloned IDs. The more cloned IDs, the faster DeClone detects the cloning attack. BASE is, however, not sensitive to the change in clone ratio. When there are 10 cloned IDs, we observe only one instance of DeClone being slower than BASE. We the number of cloned IDs exceeds 10, the case of DeClone being slower than BASE vanishes. The fluctuation of DeClone's detection time eases as clone ratio increases. We thus suggest that DeClone is more suitable for large systems with a few tags compromised.

VI. Conclusion

We have studied deterministic clone detection for anonymous RFID systems. Most existing clone detection protocols necessitate the knowledge of tag IDs and may not well support privacy-sensitive RFID applications. The only dedicated effort on clone detection in anonymous systems still tolerates the existence of a few clone tags. In this paper, we propose BASE and DeClone, two clone detection protocols with detection certainty and scalability for large anonymous RFID systems.

The evaluation results show that BASE delivers faster detection for small systems while DeClone for large. Comparing with existing work, the proposed protocols can better secure RFID applications without sacrificing privacy. Future work lies in enriching applicability of the protocols (e.g., adapting to scenarios with tags distributed across multi-place like RFID-enabled supply chains).

ACKNOWLEDGMENT

This work is supported in part by the Fundamental Research Funds for the Central Universities and the National Science Foundation of China under Grant No. 61103203. We would also like to sincerely thank MASS chairs and reviewers for their helpful feedback.

REFERENCES

[1] S. Weis, S. Sarma, R. Rivest, and D. Engels, "Security and privacy aspects of low-cost radio frequency identification systems," *Security in Pervasive Computing, Lecture Notes in Compute Science*, vol. 2802, pp. 201–212, 2004.

[2] A. Juels, "RFID security and privacy: A research survey," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 381–394, 2006.

[3] L. Bolotnyy and G. Robins, "Physically unclonable function-based security and privacy in RFID systems," in *IEEE PerCom*, 2007, pp. 211–220.

[4] M. Lehtonen, D. Ostojic, A. Ilic, and F. Michahelles, "Securing RFID systems by detecting tag cloning," *Pervasive Computing*, vol. 5538, pp. 291–308, 2009.

[5] J. Liu, B. Xiao, K. Bu, and L. Chen, "Efficient distributed query processing in large RFID-enabled supply chains," in *IEEE INFOCOM*, 2014, pp. 163–171.

[6] R. Koh, E. Schuster, I. Chackrabarti, and A. Bellman, "Securing the pharmaceutical supply chain," *Auto-ID Center MIT, White Paper*, 2003.

[7] B. Janz, M. Pitts, and R. Otondo, "Information systems and health care-ii: Back to the future with RFID: Lessons learned-some old, some new," *Communications of the Association for Information Systems*, vol. 15, no. 1, pp. 132–148, 2005.

[8] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola, and V. Khandelwal, "Design and implementation of PUF-based unclonable RFID ICs for anti-counterfeiting and security applications," in *IEEE RFID*, 2008, pp. 58–64.

[9] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *ACM CCS*, 2010, pp. 237–249.

[10] L. Mirowski and J. Hartnett, "Deckard: a system to detect change of RFID tag ownership," *International Journal of Computer Science and Network Security*, vol. 7, no. 7, pp. 89–98, 2007.

[11] M. Lehtonen, F. Michahelles, and E. Fleisch, "How to detect cloned tags in a reliable way from incomplete RFID traces," in *IEEE RFID*, 2009, pp. 257–264.

[12] D. Zanetti, L. Fellmann, and S. Capkun, "Privacy-preserving clone detection for RFID-enabled supply chains," in *IEEE RFID*, 2010, pp. 37–44.

[13] D. Zanetti, S. Capkun, and A. Juels, "Tailing RFID tags for clone detection," in *NDSS*, 2013.

[14] L. Mirowski, "Exposing clone RFID tags at the reader," in *IEEE TrustCom*, 2013, pp. 1669–1674.

[15] J. Shi, M. K. Su, and Y. Li, "Batch clone detection for RFID-enabled supply chain," in *IEEE RFID*, 2014.

[16] K. Bu, X. Liu, and B. Xiao, "Fast cloned-tag identification protocols for large-scale RFID systems," in *IEEE/ACM IWQoS*, 2012, pp. 1–4.

[17] ——, "Approaching the time lower bound on cloned-tag identification for large RFID systems," *Ad Hoc Networks*, vol. 13, Part B, no. 0, pp. 271–281, 2014.

[18] M. Kodialam, T. Nandagopal, and W. Lau, "Anonymous tracking using RFID tags," in *IEEE INFOCOM*, 2007, pp. 1217–1225.

[19] H. Han, B. Sheng, C. Tan, Q. Li, W. Mao, and S. Lu, "Counting RFID tags efficiently and anonymously," in *IEEE INFOCOM*, 2010, pp. 1–9.

[20] K. Bu, X. Liu, J. Luo, B. Xiao, and G. Wei, "Unreconciled collisions uncover cloning attacks in anonymous RFID systems," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 3, pp. 429–439, 2013.

[21] K. Leong, M. Ng, A. Grasso, and P. Cole, "Synchronization of RFID readers for dense RFID reader environments," in *International Symposium on Applications and the Internet Workshops (SAINTW)*, 2006.

[22] T. Li, S. Chen, and Y. Ling, "Identifying the missing tags in a large RFID system," in *ACM MobiHoc*, 2010, pp. 1–10.

[23] C. Qian, H. Ngan, and Y. Liu, "Cardinality estimation for large-scale RFID systems," in *IEEE PerCom*, 2008, pp. 30–39.

[24] H. Yue, C. Zhang, M. Pan, Y. Fang, and S. Chen, "A time-efficient information collection protocol for large-scale RFID systems," in *IEEE INFOCOM*, 2012, pp. 2158–2166.

[25] R. Zhang, Y. Liu, Y. Zhang, and J. Sun, "Fast identification of the missing tags in a large RFID system," in *IEEE SECON*, 2011, pp. 278–286.

[26] X. Liu, B. Xiao, K. Bu, and S. Zhang, "Lock: A fast and flexible tag scanning mechanism with handheld readers," in *IEEE/ACM IWQoS*, 2014.

[27] L. Roberts, "ALOHA packet system with and without slots and capture," *ACM SIGCOMM Computer Communication Review*, vol. 5, no. 2, pp. 28–42, 1975.

[28] D. Hush and C. Wood, "Analysis of tree algorithms for RFID arbitration," in *IEEE ISIT*, 1998, p. 107.

[29] B. Sheng, Q. Li, and W. Mao, "Efficient continuous scanning in RFID systems," in *IEEE INFOCOM*, 2010, pp. 1–9.

[30] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in RFID systems," in *ACM MobiCom*, 2006, pp. 322–333.

[31] S. Chen, M. Zhang, and B. Xiao, "Efficient information collection protocols for sensor-augmented RFID networks," in *IEEE INFOCOM*, 2011, pp. 3101–3109.

[32] A. Juels and S. A. Weis, "Authenticating pervasive devices with human protocols," in *CRYPTO*, 2005, pp. 293–308.

[33] V. Lakafosis, A. Traille, H. Lee, E. Gebara, M. Tentzeris, G. DeJean, and D. Kirovski, "RFID-CoA: The RFID tags as certificates of authenticity," in *IEEE RFID*, 2011, pp. 207–214.

[34] E.-O. Blass, A. Kurmus, R. Molva, G. Noubir, and A. Shikfa, "The F_f-family of protocols for RFID-privacy and authentication," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 3, pp. 466–480, 2011.

[35] S. Gollakota, H. Hassanieh, B. Ransford, D. Katabi, and K. Fu, "They can hear your heartbeats: Non-invasive security for implantable medical devices," in *ACM SIGCOMM*, 2011, pp. 2–13.

[36] W. Shen, P. Ning, X. He, and H. Dai, "Ally friendly jamming: How to jam your enemy and maintain your own wireless connectivity at the same time," in *IEEE Symposium on Security and Privacy*, 2013, pp. 174–188.

[37] China's RFID Card Market to Nearly Double to $807 Million in 2017. [Online]. Available: http://www.luxresearchinc.com/news-and-events/press-releases/159.html

[38] X. Liu, B. Xiao, S. Zhang, and K. Bu, "Unknown tag identification in large RFID systems: An efficient and complete solution," *IEEE Transactions on Parallel and Distributed Systems*, 2014.

[39] C. Qian, Y. Liu, H. Ngan, and L. Ni, "ASAP: scalable identification and counting for contactless RFID systems," in *IEEE ICDCS*, 2010, pp. 52–61.

[40] EPC class-1 generation-2 RFID protocol V.1.0.9,. [Online]. Available: http://www.epcglobalinc.org/home

[41] Philips Semiconductors, "I-CODE Smart Label RFID Tags". [Online]. Available: http://www.semiconductors.philips.com/acrobat_download/other/identification/SL092030.pdf