ERAP: ECC based RFID Authentication Protocol

Sheikh Iqbal Ahamed
Marquette University
Milwaukee, Wisconsin, USA
iq@mscs.mu.edu

Farzana Rahman
Marquette University
Milwaukee, Wisconsin, USA
farzanarahman02@gmail.com

Md. Endadul Hoque
Marquette University
Milwaukee, Wisconsin, USA
endadulhoque@gmail.com

Abstract

RFID tags are a new generation of small devices used for identification in many applications today. RFID authentication plays an important role in applications where security and privacy is a major concern. As an example, RFID has gained appreciation as an emerging technology to thwart counterfeiting problems. Public key cryptography (PKC) provides an impeccable solution to the counterfeiting problem. One recent family of public key cryptosystem is Elliptic curve cryptography (ECC) which is a better choice than RSA cryptographic system because of its shorter key length. In this paper, we propose a secure, mutual offline authentication protocol which is based on ECC. Finally, we present security analysis of our proposed authentication protocol.

1. Introduction

The primary goal of RFID technology is object identification. RFID tags do not need physical contact for identification which is performed through the transmission of radio signals. This allows objects to be read in large numbers without physically handling them. It is expected that these tags will facilitate many new applications and will link the real and virtual world in the near future.

Supply chain management can be referred to as a successful application of RFID technology. It is an enabling technology that has the potential of helping retailers provide the right product at the right place at the right time, thus maximizing sales and profit. RFID helps to uniquely identify each container, pallet, case and item being manufactured, shipped and sold, thus providing the building blocks for increased visibility throughout the supply chain.

Another important application of RFID systems is to detect counterfeit products. In supply chain, product authentication provides a great opportunity to find illicit trade and business by identifying counterfeit products. Counterfeiting is a rapidly growing problem that affects a great number of industries and harms societies in many ways. Therefore, new technologies must be put in place to prevent a counterfeit threat. And RFID technology has been identified as one of the major anti-counterfeiting technologies.

The anti-counterfeiting problem can also be rephrased as an authentication problem. In order to avoid counterfeiting, RFID tags need to be unclonable. Moreover, retrieving a tag's secret information by attacking the protocols that are carried out between the reader and a tag needs to be impossible and must be avoided. Protection against physical unclonability is achieved by using physical countermeasures [1] and protection against passive or active attacks on the protocols is provided by using cryptographic functions such as secure identification protocols.

Public key cryptography (PKC) offers an attractive solution to the counterfeiting problem. But most of the previous work on RFID security considered only symmetric-key algorithms such as AES [2]. It is still not clear whether public key algorithms can be implemented on small constrained devices such as RFID tags and can comply with memory, area, performance and power requirements typical of these applications. However in [3], authors investigated PKC based identification protocols that are useful for anti-counterfeiting applications.

In comparison with symmetric key based identification schemes, public key cryptography is more flexible requiring no complicated pre-distributed key and pair wise key sharing negotiation. The RSA cryptographic system is perhaps the most popular public key algorithm. Elliptic curve cryptography (ECC) is a relatively new family of public key algorithms that can provide shorter key lengths; depending upon the environment and application in which it is used, improved performance can be achieved. However, it is a common belief in the research community that public key cryptography, such as RSA and ECC, is not practical because the required computational intensity is prohibitive for devices with limited computation capability and extremely constrained memory space. Some recent progress in ECC and RSA implementation shows that public key cryptography is feasible for small sensor devices and RFID tags.

Recently a few papers [1], [4] discussed the feasibility of ECC based PKC on RFID tags. Here, we adopt the belief that ECC based public key algorithms are feasible for RFID identification or authentication. In this paper, we propose ECC based RFID authentication protocol (ERAP) which is secure against some major passive and active attacks. Our proposed protocol is a mutual offline authentication protocol which ensures that the tag and the reader authenticate each other prior to any data exchange. As it is a mutual authentication protocol, here a tag releases its data only to an authenticated reader and the reader can access only those tags for which it is authorized

The remainder of this paper is organized as follows. The next section provides an overview of related works. In section 3, we present some technical preliminaries related to our protocol. Section 4 contains our proposed protocol, ERAP. This is followed by the security analysis of ERAP in section 5. Some demonstrative application of ERAP is presented in section 6. Finally, we conclude in section 7.

2. Related works

The research literature in RFID security is already quite extensive and growing. For reference, an online repository is available at [5].

Some lightweight authentication protocols are already proposed in [6] and [7]. But they mainly considered online authentication, in which the reader shares a pre-assigned key with the tag being authenticated. Moreover, they do not consider physical cloning attacks. RFID tags that withstand general cloning attacks, in addition to physical attacks, are introduced in [1]. In [8] and [9], a PUF security based identification scheme was introduced which allows offline authentication.

Public key cryptography has been used in data encryption, digital signatures, user authentication, and access control. Some research literature focuses on the hardware implementation of PKC on RFID tags or other small sensors. A recent work of Wolkerstorfer [4] is the first to claim that it is possible to have low-power and compact implementation of ECC which meets the constraints imposed by the EPC standards.

Moreover, in [3] and [10] the authors investigated the possibility of building RFID hardware that is capable of performing public key algorithms which are based on ECC. Though RSA is the most popular public key algorithm, it is mostly used in web browsers and email packages. ECC is a new public key algorithm that can provide shorter key lengths. Therefore, ECC is considered a good replacement for RSA based public key cryptosystems since 160 bit ECC offers the same level of security as 1024 bit RSA encryption.

3. Technical preliminaries

Since our protocol is based on *Elliptic Curve Cryptography* (ECC), we first focus on some preliminaries of ECC.

3.1. ECC preliminaries

The foundation of ECC based encryption-decryption scheme and digital signature scheme is an *elliptic curve* (E). *Domain parameters* of an elliptic curve scheme describe an elliptic curve E defined over a finite field F_q.

Definition 3.1 A set $D=(q, FR, S, a, b, G, n, h)$ of *domain parameters* consists of:
The field size q.

1. FR (field representation) is an indication of the representation used for the elements of F_q.
2. If the elliptic curve is randomly generated in accordance with [11], a s*eed* S is used. The seed length is at least160 bits [12].
3. Two coefficients $a, b \in \mathrm{F}_q$ that defines the equation of E over F_q.
4. A finite point $G=(x_G, y_G) \in E(\mathrm{F}_q)$ where $x_G, y_G \in \mathrm{F}_q$. G has prime order and is called *base point*.
5. The order n of the point G, with $n > 2^{160}$ and $n > 4\sqrt{q}$.
6. The co-factor $h = \#E(\mathrm{F}_q)/n$.

Detail descriptions of domain parameters are provided in [11-13]. There are security risks associated with multiple users sharing the same elliptic curve domain parameters [11]. Detail security considerations are described in the standard X9.62 [11].

Like any other *public-key crypto*, ECC is associated with a key pair– a *private key* and a *public key*. The private key is a statistically unique and unpredictable integer $d \in [1, n-1]$. And the corresponding public key is the scalar multiplication of d and G, i.e., $Q = dG$. Therefore the key pair (Q, d) is associated with the domain parameters D of the elliptic curve.

3.2. The model of RFID system

A model for our RFID authentication system has two direct components: tags T and readers R. Tags are wireless transponders embedded in physical objects for detection and prevention of product counterfeiting. Readers are transceivers— they can query tags for identification of objects and/or subjects. A Certification Authority CA is an indirect party that is trusted by all the tags and the readers. However, CA is not mentioned as a direct component of our RFID system since an offline authentication scheme demands no direct participation of a back-end server. CA's mere function is to deploy all the tags as well as to

authorize readers. In other words, CA performs only at the time of establishment of the system. That's why CA is not included as a direct component.

As our protocol is based on ECC, each party (CA, tag and reader) needs to be capable of performing calculations based on ECC. CA generates elliptic curve domain parameters as well as key pairs in accordance with X9.62 [11]. To thwart the attacks of an adversary and to enhance the system security, we propose to select a unique elliptic curve for each RFID tag in the system.

Definition 3.2 A *Certification Authority* CA, the indispensable and indirect component of the RFID system, is equipped with four algorithms:

1. A *domain parameter generation algorithm* that generates a set D of domain parameters for each tag in the system. This algorithm randomly selects an elliptic curve over F_q according to X9.62.
2. A *domain parameter validation algorithm* that checks the validity of the set D before moving on to the next task.
3. A *key generation algorithm* that takes as input the set D and generates a key pair (Q,d), where d is the *private* key and Q is the corresponding *public* key. A key pair is generated for each elliptic curve, i.e., for each tag.
4. A *public key validation algorithm* that takes the set D and the associated public key Q as inputs and checks the validity of Q for the given set of D.

CA is assumed not to be compromised and is trusted by each party.

An RFID tag T is the smallest of all the components of our system. During the enrollment, each tag T receives its unique identifier ID, a unique set of domain parameters D and the associated private key d from CA (see Figure 1(a)). All these data are written into the ROM (EEPROM) memory of the tag. These data are secret for each tag. We assume that a tag does not reveal these secret data to any reader, not even to any other tag of the system. However, we do not consider here any hardware-based physical attack. Moreover, each tag is capable of performing an elliptic curve computation along with a modular computation.

The other direct component of our system is the RFID reader R. Since a reader can perform more extensive computations than a tag, the reader has the major function in our authentication protocol. During the enrollment, each reader R gets its *contact list* L from CA after R authenticates itself to CA (see Figure 1(b)). The contact list contains the identifying information of each tag T that R is authorized to access. We assume the communication between R and CA is performed via a secure channel.

Definition 3.3 If a reader R is authorized to access the tags $T_1, T_2, \cdots, T_\gamma$, then after authenticating itself to CA the reader R gets its *contact list* L as follows:

$$L = \{(ID_i, Cert_i) \mid ID_i \text{ is the identifier of } T_i;\ Cert_i \text{ is the certificate of } T_i; 1 \le i \le \gamma\}$$

where a *certificate* $Cert_i$ of Tag T_i is

$$Cert_i = \{(D_i, Q_i) \mid D_i \text{ is the set of domain parameters of } T_i;\ Q_i \text{ is the public key of } T_i\}$$

ID_i	parameter D_i	d_i

(a)

ID_1	parameter D_1	Q_1
ID_2	parameter D_2	Q_2
⋮		
ID_γ	parameter D_γ	Q_γ

(b)

Figure 1: (a) Identifying and secret information of Tag T_i received from CA (b) The contact list L of Reader R received from CA

In our system a certificate of a tag has the same content like ECC based public-key crypto. But the availability and accessibility of a certificate are restricted in our system. Here the certificate of a tag is accessible only to the readers that have access to the tag. Conversely, in public-key crypto, the certificate of a party is available and easily accessible to another party that wants to communicate with the former.

3.3. The adversary

The goal of an adversary in any RFID system is to counterfeit a real tag with its real data such that, only with small probability, can it be distinguished from the real one. Evidently, this fake tag can let a fake product be identified as an authentic one just by embedding the fake tag into the fake product. In ERAP, an adversary is denoted as Ă. The adversary can control a number of readers and tags. Each reader and tag controlled by Ă is denoted as Ř and Ť, respectively. Ř is unauthorized to have access to real tags as it cannot get any contact list from CA. Similarly, Ť is not valid as it does not have secret and identifying information of any tag. Moreover, we assume all the entities (tags, readers, CA including adversary, adversarial readers and adversarial tags) have polynomially bounded resources.

3.4. Attack model

We assume that Ă is more formidable than a passive attacker. In addition to eavesdropping (passive attack) on the channel between a valid reader and a valid tag, Ă, like an active attacker, can install a rogue reader Ř that can communicate with a valid tag. Even Ă can also install a fake tag Ť to communicate with an authorized reader. In

both cases, the adversary wants to counterfeit a tag with the learned information. Furthermore Ă can launch physical attacks. However, hardware-based defenses against physical attacks are beyond the scope of this paper.

4. Details of ERAP

A mutual authentication protocol enables communicating parties (a reader and a tag) to satisfy themselves respectively about each other's identity. A challenge-response based protocol facilitates both parties (a reader and a tag) to generate a challenge for the other party to respond to. Each party proves its authenticity by sending the accurate response to the other party. In an offline authentication protocol, the authentication is solely performed by a reader and a tag without any direct involvement of the back-end server, like *CA*'s. ERAP is a challenge-response based offline protocol for mutual authentication. ERAP includes the following set of algorithms and definitions:

1. δ_R is a random number generated by reader R. It is the challenge from R to tag T.
2. δ_T is a random number generated by tag T. It is the challenge from T to R.
3. $\delta_R, \delta_T < n$ [13], where n is the order of the point G in ECC (see §3.1).
4. $\text{ProveTag}(D, d, \delta_R) \rightarrow (r_T, s_T)$: This algorithm is executed by the tag T. It takes in the domain parameter D, the associated private key d and the challenge δ_R, received from the reader R. It returns the tag's response as an ordered pair (r_T, s_T) that T transmits to R.
5. $\text{VerifyTag}(D, Q, (r_T, s_T)) \rightarrow \alpha$: The reader R executes this algorithm upon receiving the response (r_T, s_T) from the tag T. It takes as input the domain parameters D, the corresponding public key Q and the received response (r_T, s_T). It verifies whether the response is accurate or not. It outputs α, where $\alpha \in \{\text{"}Accept\text{"}, \text{"}Reject\text{"}\}$.
6. $\text{ProveReader}(D, Q, \delta_T) \rightarrow (r_R, s_R)$: This algorithm is executed by R. It determines the reader's response (r_R, s_R) to the received challenge δ_T from T, given its domain parameters D and the corresponding public key Q.
7. $\text{VerifyReader}(D, d, (r_R, s_R)) \rightarrow \alpha$: This is analogous to $\text{VerifyTag}(.)$ except that it is executed by T upon receiving the response (r_R, s_R) from R. It generates α as output, where $\alpha \in \{\text{"}Accept\text{"}, \text{"}Reject\text{"}\}$.

The complete authentication scheme is portrayed in Figure 2. Since it is a mutual authentication protocol, we describe the protocol in two steps: one is "Tag authenticates itself to reader" and other is "Reader authenticates itself to tag".

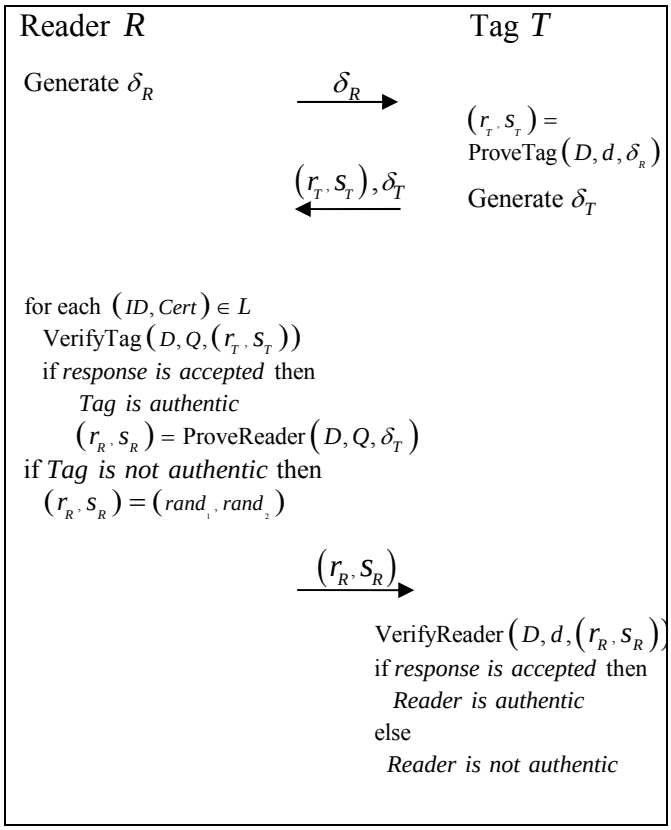

Figure 2: The ERAP

4.1. Step 1: Tag authenticates itself to Reader

At the beginning of the protocol, a reader R generates a random number δ_R and transmits it as a challenge to the tag T. Now it is time for the tag to prove its identity. T executes ProveTag(.) (see ALGORITHM 4.1) and replies to R with the response (r_T, s_T) against the challenge.

In step 1, the reader R plays the role of the verifier. Reader executes VerifyTag(.) to verify the received response (see ALGORITHM 4.2). R accepts the tag T as authentic only if:

$$\left.\begin{array}{l}\exists (ID, Cert) \in L \text{ such that} \\ \text{VerifyTag}(D, Q, (r_T, s_T)) \text{ accepts the response.}\end{array}\right\} \text{(Exhaustive Search)}$$

Step 1 is based on *Digital Signature Algorithm* (DSA). In fact it is more like *Elliptic Curve Digital Signature Algorithm* (ECDSA). A digital signature offers *data integrity* along with *authentication* and *nonrepudiation*.

Our concern is the authentication feature of a digital signature in that the other entity in the data exchange is shown to be the entity it claims to be. Here ProveTag(.) is the signature generation algorithm and VerifyTag(.) is the signature verification algorithm. Though ProveTag(.) and VerifyTag(.) are plausibly similar to the signature generation and verification algorithms of ECDSA, there are subtle dissimilarities. Instead of the message digest, both algorithms use the challenge generated by the reader. The proof that the verification algorithm works is given in §4.3.

ALGORITHM 4.1 ProveTag

INPUT: Domain Parameters $D=(q,FR,S,a,b,G,n,h)$, private key d and a challenge δ_R

OUTPUT: Response (r_T,s_T), an ordered pair based on δ_R

1. Select a random integer $k\in[1,n-1]$.
2. Compute $P=kG$ and convert the field element x_P to integer $\bar{x}_P$.
3. Compute $r_T=\bar{x}_P \bmod n$. If $r_T=0$, goto step 1.
4. Compute $s_T=k^{-1}(\delta_R+dr_T)\bmod n$. If $s_T=0$, goto step 1.
5. Return (r_T,s_T).

4.2. Step 2: Reader authenticates itself to Tag

During step 1, in addition to the response (r_T,s_T) the tag T sends a new challenge δ_T for the reader R. Now it is the reader's turn to prove itself to T. If the reader finds the validity of the purported response of T, it produces a response (r_R,s_R) to be sent to the tag T. Otherwise R generates two different random numbers as the response. To generate a valid response the reader R executes ProveReader(.) (see ALGORITHM 4.3) using the certificate of the tag. The seed parameter S (see §3.1) plays a significant role to generate the response. Its length makes the response hard to be forged.

Upon receiving the response from the reader, T verifies (r_R,s_R) by using VerifyReader(.) (see ALGORITHM 4.4) to ascertain that the other entity is indeed the legitimate reader. After this step ERAP terminates. In step 2, the tag T does not have to search exhaustively. One execution of VerifyReader(.) is enough for T to figure out the validity of the reader.

ALGORITHM 4.2 VerifyTag

INPUT: Domain Parameters $D=(q,FR,S,a,b,G,n,h)$, public key Q and received response (r_T,s_T) based on δ_R

OUTPUT: Acceptance or rejection of the response.

1. Verify that $r_T\in[1,n-1]$ and $s_T\in[1,n-1]$. If any verification fails, then Return (*Reject the response*).
2. Compute $w=s_T^{-1}\bmod n$.
3. Compute $u_1=\delta_R w\bmod n$ and $u_2=r_T w\bmod n$.
4. Compute $P'=u_1G+u_2Q$. If $P'=\infty$, then Return (*Reject the response*).
5. Convert $x_{P'}$ to integer $\bar{x}_{P'}$ and compute $v=\bar{x}_{P'}\bmod n$.
6. If $v=r_T$, then Return (*Accept the response*); Else Return (*Reject the response*).

ALGORITHM 4.3 ProveReader

INPUT: Domain Parameters $D=(q,FR,S,a,b,G,n,h)$, public key Q and a challenge δ_T

OUTPUT: Response (r_R,s_R), an ordered pair based on δ_T

1. Select a random integer $k'\in[1,n-1]$.
2. Compute $P=k'Q$ and convert the field element x_P to integer $\bar{x}_P$.
3. Compute $r_R=\bar{x}_P\bmod n$. If $r_R=0$, goto step 1.
4. Compute $s_R=k'^{-1}(\delta_T+Sr_R)\bmod n$. If $s_R=0$, goto step 1.
5. Return (r_R,s_R).

4.3. Proof of the verification algorithms

4.3.1. Proof that VerifyTag(.) works. If the response (r_T,s_T) against the challenge δ_R is indeed generated by the legitimate tag, then $s_T=k^{-1}(\delta_R+dr_T)\bmod n$. From the equation we obtain,

$$\begin{aligned} k\equiv s_T^{-1}(\delta_R+dr_T) &\equiv s_T^{-1}\delta_R+s_T^{-1}r_Td \\ &\equiv w\delta_R+wr_Td \\ &\equiv u_1+u_2d \pmod n \end{aligned}$$

Now, in VerifyTag(.) a point $P'=u_1G+u_2Q$ is generated. If the reader has access to T, then its contact list contains valid G and Q to generate P'. Thus

$$P'=u_1G+u_2Q=(u_1+u_2d)G=kG=P$$

and therefore, $v=r_T$ as required.

4.3.2. Proof that VerifyReader(.) works. If the response (r_R,s_R) against the challenge δ_T is certainly produced by the authorized reader, then $s_R=k'^{-1}(\delta_T+Sr_R)\bmod n$. From this equation we obtain,

$$\begin{aligned} k'\equiv s_R^{-1}(\delta_T+Sr_R) &\equiv s_R^{-1}\delta_T+s_R^{-1}r_RS \\ &\equiv w\delta_T+wr_RS \\ &\equiv u_1+u_2S \pmod n \end{aligned}$$

The tag T generates a point $P' = (u_1 + u_2 S)dG$ according to VerifyTag(.) . Since the tag has its own domain parameters and private key, we can say the point P' is

$$P' = (u_1 + u_2 S)dG = k'dG = k'Q = P$$

and as a result $v = r_R$, as required.

ALGORITHM 4.4 VerifyReader

INPUT: Domain Parameters $D = (q, FR, S, a, b, G, n, h)$, private key d and received response (r_R, s_R) based on δ_T

OUTPUT: Acceptance or rejection of the response.

1. Verify that $r_R \in [1, n-1]$ and $s_R \in [1, n-1]$. If any verification fails, then Return (*Reject the response*).
2. Compute $w = s_R^{-1} \bmod n$.
3. Compute $u_1 = \delta_T w \bmod n$ and $u_2 = r_R w \bmod n$.
4. Compute $P' = (u_1 + u_2 S)dG$. If $P' = \infty$, then Return (*Reject the response*).
5. Convert $x_{P'}$ to integer $\bar{x}_{P'}$ and compute $v = \bar{x}_{P'} \bmod n$.
6. If $v = r_R$, then Return (*Accept the response*); Else Return (*Reject the response*).

5. Security analysis of ERAP

In this section, we outline a number of attacks that an RFID system must prevent. This is followed by the counter-measures an RFID system should take. Then, we explain how ERAP defends the system against these attacks. A legitimate reader and a legitimate tag are denoted by R and T , respectively.

PRIVACY PROTECTION. According to [14], RFID gives rise to two major privacy concerns: covert tracking and inventorying.

A. TRACKING. As RFID tags respond to any reader's challenges, tags can provide a ready vehicle for tracking by responding with a constant reply, for example, a fixed serial number. This privacy problem exacerbates when any personal information is combined with the tag serial number. To prevent clandestine physical tracking, a tag's response must be scrambled, and must not carry any personal data.

Ă can launch an attack to track a tag T by controlling a rogue reader Ř. Now we consider an active attack initiated by Ă to track T . However, a passive attacker can even harvest enough information so that it is able to track T . If repeatedly challenging T with a same value yields a consistent reply, then Ă can distinguish the tag from other RFID tags since the consistent reply becomes a signature of T . Therefore Ř starts querying T with a fixed δ_R . But a random integer k used in ProveTag(.) makes the responses of T random to Ř, even though Ř reuses the same δ_R . Thus, ProveTag(.) thwarts the physical tracking of a tag.

B. INVENTORYING. Sometimes a tag's response contains a unique serial number as well as the information of the product to which the tag is attached. People carrying such tags are subject to secret inventorying. A rogue reader comes to know about the consumer's personal information. To protect the consumer's interests, a protocol must ensure that the tag's reply contains no product information. Moreover, a tag's reply should be disguised so that only authorized readers can identify them.

ERAP prevents secret inventorying attacks since ERAP does not exploit any intrinsic information, such as tag ID. Rather it employs domain parameters, private and public keys, and the challenges and responses between tags and readers. Therefore, ERAP protects user privacy.

CLONING. Here, we consider the cloning attack launched by an active attacker. To launch this attack, Ă first queries the tag T and obtains a response. By placing this response in a fake tag Ť, Ă attempts to counterfeit the real tag. Ă becomes successful in the attempts if Ă can deceive a legitimate reader R , i.e. R fails to distinguish Ť from T .

Under our protocol, whenever Ă challenges the tag T , Ă gets a different response each time due to random integer k in ProveTag(.) . Now Ă places this response in Ť. But Ť frustrates Ă as it fails to fool the valid R , because Ă cannot predict the challenge δ_R , that R will use to query the tag Ť, at the time of getting the response from T . Therefore ERAP is secured against the cloning attack.

EAVESDROPPING. So far we considered the active attacks that Ă can launch in our RFID system. We assume Ă can eavesdrop on both the channels between R and T . Therefore Ă learns the challenges-responses between R and T , and later uses these data to launch any of the above mentioned attacks.

According to our protocol, Ă cannot track T because each time it is queried, T replies with a different response, even if the same δ_R is reused. Even Ă fails to get any inventory information as ERAP does not reveal any personal and/or product related information. Moreover, ERAP prevents Ă from launching a cloning attack by eavesdropping. Suppose Ă tries to impersonate tag T which is named as Ť and wants to fool an honest reader R with which T has communicated recently. To deceive R , Ť has to generate a valid response. But each time R generates a new challenge δ'_R that was not used before, so Ť fails to generate a valid response (r'_T, s'_T) .

PHYSICAL ATTACK. Physical attack means Ă compromises either the reader R or the tag T . We consider each case. Our assumptions include that once Ă compromises R or T , Ă learns everything about the reader or the tag. However, here we do not consider hardware based physical attack.

A. Ă **COMPROMISES** R**.** When Ă compromises the reader R, Ă acquires knowledge about the contact list L of R. Therefore, the adversarial reader Ř of Ă can successfully impersonate R and communicate with the tags that the reader R has access to. Suppose the tag T resides in L and Ă wants to counterfeit T denoted as Ť. Ă succeeds in its attempt if Ť is able to fool another legitimate reader R_x that is authorized to access T. But under our scheme, to counterfeit a tag and consequently to deceive a valid reader is possible only when Ă succeeds to recover the private key d of T. And this problem is known as *Elliptic Curve Discrete Logarithm Problem* (ECDLP) [11] that cannot be resolved by the adversary Ă equipped with polynomially bounded resources. Therefore, Ă fails to counterfeit T.

B. Ă **COMPROMISES** T. Whenever Ă compromises T, Ă learns about the domain parameters D and the private key d of T. As a result, the adversarial tag Ť of Ă can effectively impersonate the tag T. Using this information, Ă wants to clone another valid tag T_x. And with this cloned tag, Ă wants to cheat an honest reader R that is authorized to access T_x. Under ERAP, knowing the D and d of T does not help Ă to clone T_x. As T_x gets uniquely different domain parameters D and private key d from CA, Ă cannot recover by compromising only T. Consequently Ă fails to create a fake tag to fool R.

6. Illustrative examples

ASSET TRACKING. When beyond our sight , RFID can be used to track numerous things. The tracking and locating capabilities of RFID makes it a recent preference for security and safety purpose. Manufacturing is using RFID to track different stages of products. Even tracking goods in transit becomes viable due to RFID. Healthcare is using RFID to track and locate critical assets, to ameliorate patients alongside care procedures, to establish protection for the tracking of movements of newborn babies, etc. One purposel of RFID is to track people people and items in and out of secure and sensitive areas. RFID is being used to track and locate valuable assets such as computer, server room, smart spaces, museums, art galleries. To successfully perform in all these applications, RFID entities need to be authenticated before exchanging or referring to product/people related information. Each RFID reader has to make sure that the tag is legitimate. Conversely, each tag must be certain that the reader it is communicating with is indeed valid. Our proposed protocol ERAP is a feasible solution for the authentication purpose.

7. Conclusion

In this paper, we present an RFID authentication protocol which is entirely based on Elliptic curve cryptography (ECC). It is a better choice than RSA because ECC can provide the same security level as RSA can with shorter key lengths. Our proposed ECC based authentication scheme can be used to solve the product counterfeiting problem which has experienced a steep increase in recent years. This protocol is a mutual authentication protocol as it provides reader-to-tag and tag-to-reader authentication and is secure against common major attacks. Finally, we demonstrate the application of our protocol in asset tracking.

8. References

[1] P. Tuyls and L. Batina, "RFID-tags for Anti-Counterfeiting". In D. Poincheval, editor, *Topics in Cryptology - CT-RSA 2006*, vol. 3860 of *LNCS*, pp. 115-131, Springer Verlag, Feb. 2006.

[2] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, "Strong Authentication for RFID Systems Using the AES Algorithm". In M. Joye and J. -J. Quisquater, editors, *Cryptographic Hardware and Embedded Systems – CHES 2004*, vol. LNCS 3156, pp. 357-370, Springer, 2004.

[3] L. Batina, J. Guajardo, T. Kerins, N. Mentens, P. Tuyls, and I. Verbauwhede, "Public key cryptography for RFID-tags". *RFIDSec* 2006.

[4] J. Wolkerstorfer, "Scaling ECC Hardware to a Minimum". In ECRYPT workshop – Cryptographic Advances in Secure Hardware – CRASH 2005, September 2005. Invited Talk.

[5] G. Avoine, "Security and privacy in RFID systems". http://www.avoine.net/rfid/, 2008.

[6] A. Juels., "Strengthening EPC Tags Against Cloning". In M. Jakobsson and R. Poovendran, editors, *ACM Workshop on Wireless Security – WiSe 2005*, pp. 67-76. ACM Press, 2005.

[7] A. Juels and S. A. Weis, "Authenticating pervasive devices with human protocols". In V. Shoup, editor, *Advances in Cryptology: Proceedings of CRYPTO 2005*, vol. 13621 of *LNCS*, pp. 293-308, Springer-Verlag, 2005.

[8] P. Tuyls and B. Skoric, "Secret Key Generation from Classical Physics: Physical Uncloneable Functions". In S. Mukherjee, E. Aarts, R. Roovers, F. Widdershoven, and M. Ouwerkerk, editors, *AmIware: Hardware Technology Drivers of Ambient Intelligence*, vol. 15 of *Philips Research Book Series*, Springer-Verlag, September 2006.

[9] B. Gassend, D. E. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions". In Vijayalakshmi Atluri, editor, *ACM Conference on Computer and Communications Security—CCS 2002*, pp. 148-160, ACM November 2002.

[10] L. Batina, J. Guajardo, T. Kerins, N. Mentens, P. Tuyls, and I. Verbauwhede, "An elliptic curve processor suitable for RFID-tags". CryptologyePrint Archive, Report 2006/227.

[11] "ANSI X9.62-1998, Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)," American National Standard Institute, 1998.

[12] D. Johnson, A. Menezes, and S. Vanstone, "The Elliptic Curve Digital Signature Algorithm (ECDSA)". *International Journal of Information Security*, 2001.

[13] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography.* Springer-Verlag, 2004.

[14] A. Juels, "RFID Security and Privacy: A Research Survey". Manuscript, 2005.