Untraceable RFID Authentication Protocols: Revision of EC-RAC

Yong Ki Lee[(1)], Lejla Batina[(2)], and Ingrid Verbauwhede[(1),(2)]

***Abstract*— Radio Frequency IDentification (RFID) systems are steadily becoming paramount due to their vast applications such as supply chains, inventory, tolling, baggage management, access control etc. While they have potentials to improve our lives, they also present a privacy risk. Privacy is often overlooked in many applications, but due to pervasiveness of RFIDs the issue has to be taken into account. However, additional security always comes at price and the scarceness of resources on a tag makes conventional privacy-preserving protocols infeasible.**

In this paper we propose several authentication protocols that are all made of the same building blocks. More precisely, we first revise the EC-RAC (Elliptic Curve Based Randomized Access Control) protocol and we expand it into several authentication protocols. All the proposed protocols satisfy the basic requirements, which are the system scalability, un-traceability and security against cloning attacks and replay attacks, but each protocol has different security properties. The security proofs are implied by means of cryptographic reductions, *i.e.* they are based on the security of the Schnorr protocol and the hardness of the decisional Diffie-Hellman problem.

***Index Terms*— Authentication Protocol, RFID System, Elliptic Curve Cryptography, Tracking Attack, Un-traceability, Mutual Authentication.**

I. INTRODUCTION

The properties that RFID authentication protocols should preserve are the system scalability and the security against cloning attacks, replay attacks and tracking attacks. However, security and privacy for RFIDs are challenging problems due to the scarceness of resources on a tag such as the computing capability, memory and energy/power. In addition, each item should cost only a few cents. Previously, it was shown that public-key cryptography (PKC) algorithms are necessary to solve the requirements [4]. Therefore, it is not possible to satisfy the requirements only with symmetric cryptographic algorithms such as hash algorithms and symmetric key encryption algorithms. Furthermore, some conventional PKC based authentication protocols such as the Schnorr protocol [18] and the Okamoto protocol [16] were designed without concerns for the tracking attack, hence they fail to satisfy the un-traceability which is shown in [11].

In this paper we propose several authentication protocols by revising and expanding EC-RAC [11]. EC-RAC was proposed to resolve these requirements but there have been reports indicating problems by T. Deursen *et. al.* [6] and J. Bringerl *et. al.* [3] independently. The proposed protocols are consisting of a few components which are the ID-transfer scheme, the password-transfer scheme and the server's authentication scheme. These components can be combined in different ways depending on the system and/or security requirements of applications, which result in 6 different authentication protocols. Different compositions require different amounts of computations on the server and a tag. The security proofs are done by cryptographic reductions. Therefore, the proposed protocols are secure as long as the underlying primitives are secure, which are the Schnorr protocol and the Diffie-Hellman scheme.

The remainder of this paper is organized as follows. In Sec. 2, some background and related work are given. The system parameters and the security of EC-RAC are given in Sec. 3. The components of the authentication protocols are proposed in Sec. 4, and they are composed to produce final authentication protocols in Sec. 5. The conclusions of this paper are given in Sec. 6.

II. BACKGROUND AND RELATED WORK

The operational and cryptographic properties for RFID systems can be summarized as follows:

- **Scalability**
  Some protocols using hash or symmetric key algorithms, *e.g.* [24], [15], are not scalable since the computational workload on the server increases linearly with the number of tags. Considering that a general RFID system can have a large number of tags, the scalability is a required property.
- **Anti-cloning**
  If a group of tags share the same secret key and use it for the authentication, it is vulnerable to cloning attacks. If an attacker succeeds to crack one of the tags, he or she can use the revealed secret to clone some other tags. Therefore, a secret key should be pertinent only to a single tag so that a revealed secret key cannot be used for any other tag.
- **Replay Attack (Impersonation Attack)**
  An attacker should not be able to generate a valid set of messages when he does not know the secret keys of a tag. An attacker may actively query a tag and/or perform some polynomial time computation utilizing known information such as the system parameters, the public-key of the server and the history of exchanged messages.
- **Un-traceability (Security against the tracking attack)**
  RFID tags are supposed to respond with some message

(1) The authors are with the Department of Electrical Engineering – University of California, Los Angeles, 56-125B Engineering IV Building 420 Westwood Plaza, Los Angeles, CA 90095-1594 USA

(2) The authors are with the Department of Electrical Engineering – ESAT/SCD-COSIC, Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, B-3001, Belgium.

Email: jfirst@ee.ucla.edu, Lejla.Batina@esat.kuleuven.be

whenever they receive a query message from a reader. If the responses are fixed or predictable by an attacker, it results in a privacy problem. An attacker is possibly able to track a tag, and hence its owner too, and collect data for malicious purpose. Therefore, the responses of tags should be randomized so that it is infeasible to extract any information of the communications between a tag and a reader.

- **Backward/Forward Un-traceability**
  Even if all the information of the tag is revealed to an attacker at a certain moment, an attacker should not be able to track a tag in the past or future communications. Therefore, for the proof of this property, we assume an attacker knows the secret keys of a tag. However, an attacker still does not know random numbers temporarily generated and used inside of a tag. This property is a sufficient condition for the un-traceability. We put this strong property as an option in the proposed protocols.

Some detailed definitions and models can be found in [10], [13], [17], [23]. We use definitions from [13]. Since the security of proposed protocols are proved by cryptographic reductions, they are rather independent of the security models. By a reduction, we show that the protocols are as secure as the underlying primitive schemes. Showing the scalability and the anti-cloning are negligible since these are trivially followed by using PK (Public-Key) algorithms and so are our protocols. The proposing protocols are scalable since the computation amount is fixed independent of the number of tags, and are anti-cloning since the used secret keys are different in each tag. To show the security against the replay attack and un-traceability we reduce the protocols into the Schnorr protocol and the Diffie-Hellman scheme. Therefore, the proposed protocols are as secure as the Schnorr protocol and the Diffie-Hellman scheme in targeted security properties.

Most of RFID authentication protocols use a hash algorithm [24], [9], [1], [15], [12], [5], [21], or secret key cryptographic algorithms [7], [8], [20], for a tag's cheap implementation compared to a PKC-based algorithm. However, these protocols cannot satisfy the basic properties for RFID systems. This is a consequence of the proof in [4].

There are some papers which propose to use PKC-based RFID systems [25], [22], [2], [11]. In [25] no specific authentication protocol is mentioned, and the Schnorr protocol [18] and the Okamoto protocol [16] are adopted in [22] and [2] respectively. However, it is shown that these two protocols are not proper for RFID systems due to their vulnerability against the tracking attack [11]. In [11] EC-RAC is also proposed to solve all the basic properties required in RFID systems. However, the works in [3], [6] showed EC-RAC's vulnerability against tracking attacks and impersonation attacks. The work in [3] also proposed the randomized Schnorr protocol as a replacement of EC-RAC.

III. System Parameters and Overview

Before designing RFID cryptographic protocols, we should note that the requirements are different from conventional cryptographic systems. Considering RFID protocols as authentication protocols, there are some differences from conventional password protocols and PKC based authentication prtocols as the following:

1) Unlike conventional password protocols, in RFID systems a tag should not just transfer its ID. Transmitting an ID in plain text will cause tracking attacks.
2) Unlike conventional PKC based authentication protocols, the protocols are many to one protocols, *i.e.* many RFID tags communicate with one reader/server. Due to this property, tags' public-keys do not need to be publicly announced and hence, they can and should be securely stored and used for authentications in the server.

Since we cannot just transfer a tag's ID, we need to encrypt the ID to transfer securely. The server can authenticate a tag by using the ID. Therefore, a tag and the server should agree on an encryption key, which means that a PKC algorithm is inevitable in order to keep the properties of the scalability and the anti-cloning. Moreover, the encryption of a tag's ID should be randomized and different each time a protocol is executed to be secure against tracking attacks and replay attacks.

Among PKC algorithms, EC based algorithms would be best choice for RFID systems due to their small key sizes and efficient computations. For example, a key size of 163 bits in EC based cryptography has a compatible security level with a key size of 1024 bits in RSA or DLP (Discrete Logarithm Problem) based cryptography.

As a start of designing new protocols, two secret keys are assigned to each tag, which are x_1 (ID) and x_2 (password), similarly to conventional password protocols. Since revealing a tag's ID causes the tracking attack, the ID is also secret information just like the password. The public-keys, x_1P and x_2P, are used as an ID-verifier and a password-verifier which are securely stored in the server unlike general public-keys. For an attacking model, we suppose that an attacker knows the system parameters which can be revealed by cracking any of the tags. The system parameters and the storage of each entity are summarized in Table I. Note that the base point P must be chosen to have a prime order as required in ECC [14], [19].

TABLE I
System Parameters

System Parameters	y : Server's private key $Y(= yP)$: Server's public-key x_1 : Tag's ID x_2 : Tag's password $X_1(= x_1P)$: Tag's ID-verifier $X_2(= x_2P)$: Tag's password-verifier P : Base point in the EC group whose order is a prime n : Prime order of P
Server's storage	y, X_1, x_1, X_2, P, n
Tag's storage	x_1, x_2, Y, P, n
Attacker's storage	Y, P, n : Publicly known information

In the rest of the section, EC-RAC [11] and its security analysis are summarized.

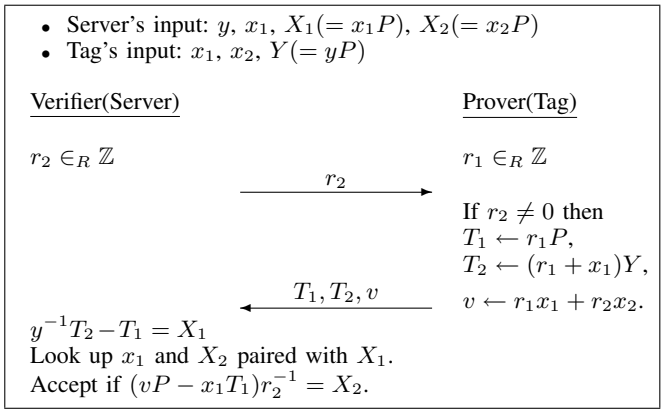

Fig. 1. The EC-RAC Protocol Flow

A. The EC-RAC protocol

The EC-RAC protocol is shown in Fig. 1. After receiving a random number from the server, a tag generates and sends three messages $T_1(= r_1P)$, $T_2(= (r_1+x_1)Y)$ and $v(= r_1x_1 + r_2x_2)$. Then, the server derives X_1 and uses it to search out x_1 and X_2 in a local database. Finally, the server validates a tag by checking whether the result of $(vP - x_1T_1)r_2^{-1} = X_2$ matches with the stored X_2.

B. Cryptanalysis of EC-RAC

The EC-RAC protocol is aimed to be secure against the tracking attack [11]. However, it is shown that in EC-RAC a tag can be still tracked by an active attack as proposed in [6]. The failure of the security proof is caused by neglecting the possibility that an attacker can use multiple sets of authentic communication history. An attacker can generate a random number c for r_2 and use it twice to get two different sets of responses from a tag. A tag will generate two random numbers k_1 and k_2 for each of the authentication protocol.

$$\{T_1^{(1)},\ T_2^{(1)},\ v^{(1)}\} = \{k_1P,\ (k_1 + x_1)Y,\ k_1x_1 + cx_2\},$$
$$\{T_1^{(2)},\ T_2^{(2)},\ v^{(2)}\} = \{k_2P,\ (k_2 + x_1)Y,\ k_2x_1 + cx_2\}.$$

Then, an attacker can perform the following calculation.

$$(T_1^{(1)} - T_1^{(2)}) \cdot (v^{(1)} - v^{(2)})^{-1}$$
$$= (k_1 - k_2)Y \cdot \{(k_1 - k_2)x_1\}^{-1} = x_1^{-1}Y.$$

Since the result $x_1^{-1}Y$ can be a fixed value for a specific tag, a tag can be traced by an attacker.

More generalized tracking attacks and impersonation attacks are reported in [3].

IV. Protocol Design in Components

We consider the protocols in three parts as follows:

1) A tag's ID transfer to the server
   a) A tag encrypts and transfers its ID-verifier to the server.
   b) The server decrypts a tag's ID-verifier.
2) A tag's password transfer to the server
   a) A tag encrypts and transfers its password-verifier to the server.
   b) The server decrypts a tag's password-verifier and authenticates a tag.
3) The server's proof of its authentication to a tag
   a) The server transfers a session identifier to a tag.
   b) A tag verifies the server by checking the received identifier.

These three parts can be independently designed and analyzed, and can be composed differently depending on the required system and/or security requirements.

A. Secure ID Transfer Scheme

1) Protocol Description: The ID-transfer scheme is shown in Fig. 2 and 3. In this scheme, a tag generates a random number r_{t1} and T_1, and transfers T_1 to the server. Then, the server responds with a random challenge r_{s1}, and a tag produces and transfers T_2 to the server. After receiving T_2, the server calculates a tag's ID-verifier $x_1P(= X_1)$. Note that the server decrypt the tag's ID-verifier instead of the ID itself. This ID transfer scheme is a basic component that is required for all protocols in the paper.

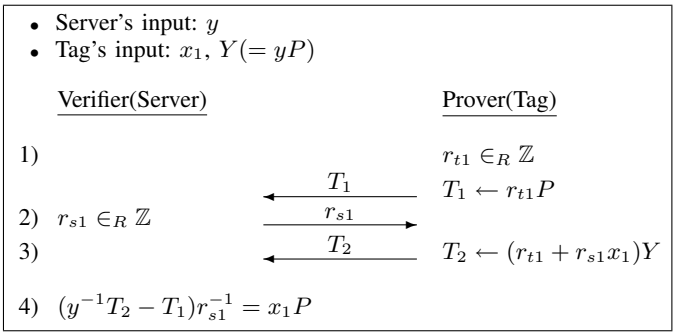

Fig. 2. Secure ID Transfer Flow

1) Tag generates a random number r_{t1} and T_1, and sends T_1 to Server.
2) Server generates and sends a random challenge r_{s1} to Tag.
3) Tag generates and sends T_2 to Server.
4) Server derives Tag's ID-verifier: $(y^{-1}T_2 - T_1)r_{s1}^{-1}$ $= (y^{-1}(r_{t1} + r_{s1}x_1)yP - r_{t1}P)r_{s1}^{-1} = x_1P = X_1$.

Fig. 3. Secure ID Transfer Description

It is possible to use only the ID transfer scheme for the authentication of a tag. The server may authenticate a tag by checking whether the decrypted ID-verifier exists in the list. However, a large number of tags may weaken the security level of the system. For example, if there are millions of tags, the probability that a randomly selected ID is identified as a valid one becomes millions times bigger than the case of only one tag. This would results in a reduction of the security level. Therefore, if the number of tags is large, the password transfer scheme should be used together with the secure ID transfer scheme to guarantee the full security of a key size.

2) Security Analysis: Security of the ID transfer scheme can be proved by cryptographic reductions. This can be reduced into two different schemes, the Schnorr protocol [18] and the Diffie-Hellman scheme.

Theorem 1: The secure ID transfer scheme can be cryptographically reduced to the Schnorr protocol. Therefore, the cryptographic properties of the Schnorr protocol are inherited to the secure ID transfer scheme.

Proof: The only difference between the ID transfer scheme and the Schnorr protocol is in the message of the last round. While the ID transfer scheme transfers $T_2(= (r_{t1} + r_{s1}x_1)Y)$, the Schnorr protocol transfers $(r_{t1} + r_{s1}x_1)$. Given $(r_{t1} + r_{s1}x_1)$ and Y which are known values to attackers, T_2 can be easily calculated. However, given T_2 and Y, $(r_{t1}+r_{s1}x_1)$ cannot be easily calculated since it is the ECDLP (Elliptic Curve Discrete Logarithm Problem). Therefore, the proposed ID transfer scheme can be cryptographically reduced to the Schnorr protocol and is at least as secure as the Schnorr protocol. ■

According to Theorem 1, the ID transfer scheme is secure against replay attacks as much as the Schnorr protocol is. However, since the Schnorr protocol is vulnerable to tracking attacks as shown in [11], the security of the ID transfer scheme against tracking attacks should be separately shown.

Theorem 2: The secure ID transfer scheme can be cryptographically reduced to the Diffie-Hellman scheme. Therefore, the cryptographic properties of the Diffie-Hellman scheme are inherited to the secure ID transfer scheme.

Proof: The ID transfer scheme can be considered as an encryption scheme using the Diffie-Hellman key agrement protocol. A one-time session key can be established between the server and a tag by a tag's transmission of $T_1(= r_{t1}P)$. Since the public-key of the server is already known to a tag, a shared session key can be derived as $r_{t_1} \cdot Y(= r_{t1}yP)$ in a tag, and as $y \cdot T_1(= yr_{t1}P)$ in the server. Now, we can interpret the message T_2 as an encryption of $r_{s1}x_1yP$ with the key of $r_{t1}yP$. The encryption is done by performing an EC point addition, which can be seen in $T_2 = r_{t1}yP + r_{s1}x_1yP$. Since $r_{t1}yP$ is a randomly generated session key and used only once, the EC point addition is sufficient for the secure encryption.

Since r_{s1} is used just to scramble the message being encrypted, there is no effect in the encryption unless $r_{s1} = 0$ which has negligible probability. Even if we consider the case of $r_{s1} = 0$ which may occur by attackers, T_2 becomes the shared random session key itself, $r_{t1}yP$, without encrypting anything. Nevertheless, it does not reveal anything since the used session key would not be used in the future and the generated messages cannot be used to be authenticated by the server.

Therefore, the ID transfer scheme can be cryptographically reduced to the Diffie-Hellman scheme and is at least as secure as the Diffie-Hellman scheme. ■

Since the used session key is a randomly generated one-time key, it is infeasible to link encrypted messages, *i.e.* exchanged messages, to a specific tag. Therefore, Theorem 2 ensures that tags are untraceable in the ID transfer scheme as long as the decisional Diffie-Hellman problem is hard. The ID transfer scheme is also backward/forward un-traceable. Since the used one-time session key $(r_{t1}yP)$ is produced by a tag's site key (r_{t1}) instead of using a tag's secret key (x_1), the revealed secret key of a tag does not help to produce $r_{t1}yP$.

In the ID-transfer scheme, we encrypt $r_{s1}x_1yP$ instead of the ID-verifier x_1P where r_{s1} is inserted to prevent the replay attack, which can be interpreted as a part of the Schnorr protocol, and y to simplify the computation on a tag. Note that $r_{s1}x_1yP$ is equivalent to x_1P for the server since r_{s1} and y are known to the server.

B. Secure Password Transfer Scheme

1) Protocol Description: The password transfer scheme is performed after the ID-transfer scheme. Therefore, this scheme starts with an assumption that the server knows the ID-verifier (X_1) already. Since the server stores a set consisting of the ID-verifier (X_1), the ID (x_1) and the password-verifier (X_2) for each tag, the server can search the database to find x_1 and X_2 which are paired with X_1 to use for the password verification. The password transfer scheme is described in Fig. 4 and 5.

- Server's input: y, $X_1(= x_1P)$, x_1, $X_2(= x_2P)$
- Tag's input: x_1, x_2, $Y(= yP)$

	Verifier(Server)		Prover(Tag)
1)			$r_{t1} \in_R \mathbb{Z}$
		$\xleftarrow{T_1}$	$T_1 \leftarrow r_{t1}P$
2)	$r_{s1} \in_R \mathbb{Z}$	$\xrightarrow{r_{s1}}$	
3)		$\xleftarrow{T_2}$	$T_2 \leftarrow (r_{t1}x_1+r_{s1}x_2)Y$
4)	$(y^{-1}T_2 - x_1T_1)r_{s1}^{-1} = x_2P$		

Fig. 4. Secure Password Transfer Flow

1) Tag generates a random number r_{t1} and T_1, and sends T_1 to Server.
2) Server generates and sends a random number r_{s1} to Tag.
3) Tag generates and sends T_2 to Server.
4) Server derives Tag's password-verifier:
$(y^{-1}T_2 - x_1T_1)r_{s1}^{-1}$
$= (y^{-1}(r_{t1}x_1+r_{s1}x_2)yP - x_1r_{t1}P)r_{s1}^{-1} = x_2P = X_2.$

Fig. 5. Secure Password Transfer Description

2) Security Analysis: The design concept of the password transfer scheme is the same as the ID transfer scheme. Instead of $(r_{t1} + r_{s1}x_1)Y$, $(r_{t1}x_1 + r_{s1}x_2)Y$ is used for T_2. Note that the used random number r_{t1} between the two schemes may not be the same at this moment. The security analysis of the password-transfer scheme can be performed in a similar way as the ID-transfer scheme, and has the same security properties.

C. Server's Authentication to a tag

Authentication of the server to a tag is somewhat different from the tag-to-server authentication since there is only one

TABLE II
AUTHENTICATION PROTOCOL CONSTRUCTIONS AND THEIR SECURITY PROPERTIES

Clasify		Protocol 1	Protocol 2	Protocol 3	Protocol 4	Protocol 5	Protocol 6
Components	ID-Transfer	O	O	O	O	O	O
	Pwd-Transfer 1	X	O	X	X	O	X
	Pwd-Transfer 2	X	X	O	X	X	O
	Server's Auth.	X	X	X	O	O	O
# of point multiplications in the server		2	4	4	4	6	6
# of point multiplications in a tag		2	3	4	4	5	6
Properties	Number of tags	Small	Large	Large	Small	Large	Large
	Backward/Forward un-traceability	Secure	Vulnerable	Secure	Secure	Vulnerable	Secure
	Authentication	One-way	One-way	One-way	Mutual	Mutual	Mutual

* Common properties: scalability, protection against cloning attacks, replay attacks, tracking attacks

server. There is no need to transfer the server's ID since a tag is expecting the specific server. The server proves its authenticity to a tag by sending a session identifier $yr_{t1}P$ which is calculated by $y \cdot T_1$. The tag can check the validity of the received identifier from the server by calculating $r_{t1} \cdot Y$. Since there is no information to encrypt or decrypt, showing the session identifier will be enough to show its authenticity.

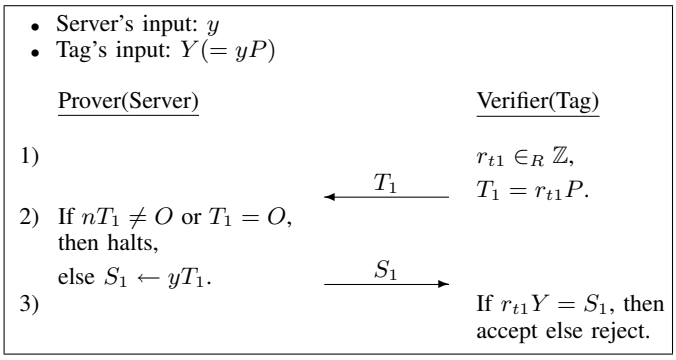

Fig. 6. Protocol Flow for Server's Authenticity

1) Tag generates a random number r_{t1} and T_1, and sends T_1 to Server.
2) Server checks if $nT_1 = O$ and $T_1 \neq O$. If the result is valid, Server generates and sends S_1 to Tag.
3) Tag checks if $r_{t1}Y = S_1$. If the result is valid, Tag authenticates Server as a valid one.

Fig. 7. Protocol Description for Server's Authenticity

In step 2 in Fig. 6 and 7, the server should check the validity of T_1. If $nT_1 = O$, then the possible orders of T_1 are the factors of n. Since n is a prime, there are only two possible orders, 1 and n itself. Therefore, by excluding the case of 1, *i.e.* $T_1 = O$, the order of T_1 can be assured to be n. This is important since an attacker may control the value of T_1 and use the responses of the server to guess the server's private key y. He possibly chooses and sends an EC point of a small order. For example, if an attacker choose a point of the order 2, he can decide whether y is a even or odd number by checking whether the responded point is T_1 or O. This kind of attacks will reveal the private key of the server. To prevent this attack, the server should check whether T_1 has the correct order n.

V. AUTHENTICATION PROTOCOL CONSTRUCTION

The proposed schemes, the ID transfer scheme, the password transfer scheme and the server's authentication scheme can be combined to produce a proper authentication protocol depending on the required system and security properties.

The possible combinations and their properties are summarized in Table II. In the protocol 1, we are just using the ID transfer scheme for a tag's authentication. The server authenticates a tag by checking the existence of a tag's ID-verifier on the list. This would be effective to minimize the computation workload on a tag if the number of tags is relatively small. If the number of tags is large enough, the ID transfer scheme should be combined with the password transfer scheme. The combination can be done in two different ways which are marked with 'Pwd-Transfer 1' and 'Pwd-Transfer 2' in Table II. These have different computation amounts and different security properties. The protocols 1, 2 and 3 are expanded by combining with the server's authentication scheme to the protocols 4, 5 and 6 respectively. In Table II, an 'O' indicates the used components, otherwise a 'X' is marked.

All the protocols are scalable and secure against cloning attacks, replay attacks and tracking attacks. The protocol 1 is the most simple version requiring only two EC point multiplications on both the server and a tag, but it has a limitation on the number of tags and no server's authentication. While the protocol 3 (protocol 6) has an additional security property of the backward/forward un-traceability compared to the protocol 2 (protocol 5), it requires one more EC point multiplication on a tag.

A. Protocol Variations: Protocol 2

1) Protocol Description: The first combination of the ID transfer scheme and the password transfer scheme is the protocol 2 in Table II which is described in Fig. 8 and 9. In this protocol, the random point $T_1(= r_{t1}P)$ is used not only for the ID transfer scheme but also for the password transfer scheme in order to minimize the computation amount on a tag.

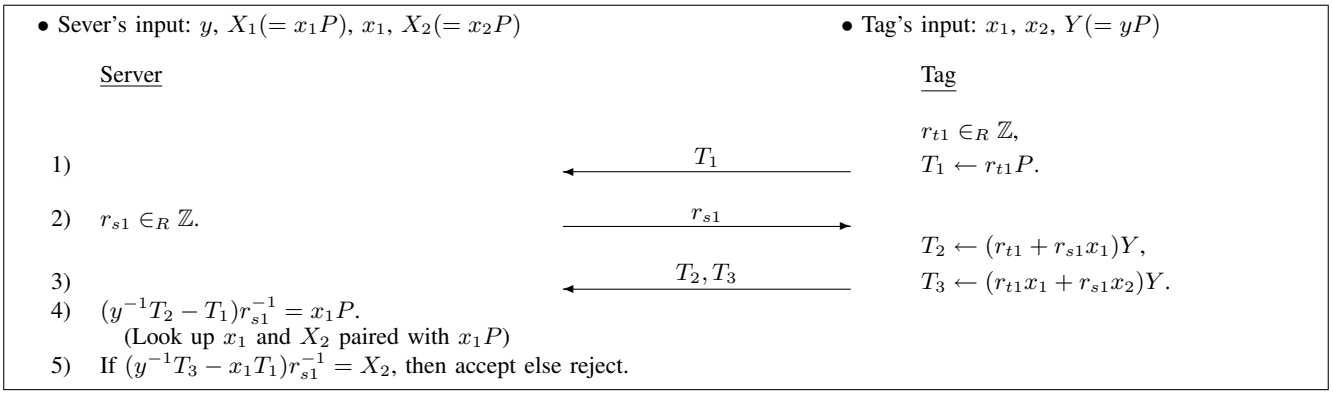

Fig. 8. Protocol 2 Flow

1) Tag generates a random number r_{t1} and T_1, and sends T_1 to Server.
2) Server generates and sends a random number r_{s1} to Tag.
3) Tag generates and sends two messages T_2 and T_3 to Server.
4) Server calculates $(y^{-1}T_2 - T_1)r_{s1}^{-1} = \left\{y^{-1}(r_{t1} + r_{s1}x_1)yP - r_{t1}P\right\} r_{s1}^{-1} = x_1P = X_1$ and using the result (X_1) searches for x_1 and X_2. If there is no valid set for X_1, then Server halts.
5) Server computes $(y^{-1}T_3 - x_1T_1)r_{s1}^{-1} = \left\{y^{-1}(r_{t1}x_1 + r_{s1}x_2)yP - x_1r_{t1}P\right\} r_{s1}^{-1} = x_2P$ and checks whether it is equal to the stored X_2. If so, Server authenticates Tag as a valid one.

Fig. 9. Protocol 2 Description

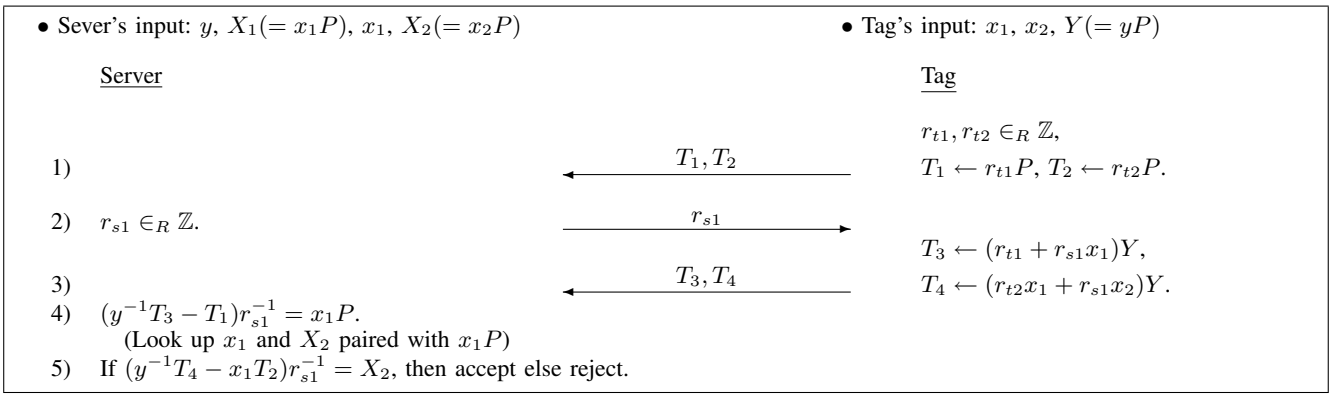

Fig. 10. Protocol 3 Flow

2) Security Analysis: By generating and transmitting $T_1(= r_{t1}P)$ to the server, a tag and the server can share two one-time session keys: $r_{t1}yP$ and $r_{t1}x_1yP$. Though two session keys use the same random number r_{t1}, they are independent since x_1 is unknown to an attacker. This can be interpreted as follows. Since x_1 is a randomly chosen fixed number, x_1yP can be seen as another random public-key of the server. Although the used random umbers are the same, they are independent since the used public-keys of the server are independent. In other words, two one-time session keys are independent as long as x_1 is unknown. The session key $r_{t1}yP$ is used to encrypt $r_{s1}x_1yP$ and $r_{t1}x_1yP$ to encrypt $r_{s1}x_2yP$. Therefore, the security property is directly inherited from the security analyses of the ID transfer scheme and the password transfer scheme if x_1 is unknown to an attacker.

However, Protocol 2 does not satisfy the forward/backward un-traceability. For example, assuming that an attacker knows x_1 and x_2, he can compute $\{T_2 - (T_3 - r_{s1}x_2Y)/x_1\}/r_{s1} = x_1Y$ which can be used to track a tag. This weakness can be explained as follows: If x_1 is revealed to an attacker, the two one-time session keys, $r_{t1}yP$ and $r_{t1}x_1yP$, are dependent since the second session key can be derived from the first one. Specifically, $r_{t1}yP \cdot x_1 = r_{t1}x_1yP$, which means that the two session keys are basically the same given x_1. This violates our purpose to use each session key only once. Therefore, this leaves a possibility of security weakness and actually causes the tracking attack.

B. Protocol 3

1) Protocol Description: In order to be secure against the forward/backward tracking attack, the protocol 2 is revised to the protocol 3 (Fig. 10). In this case a tag generates two

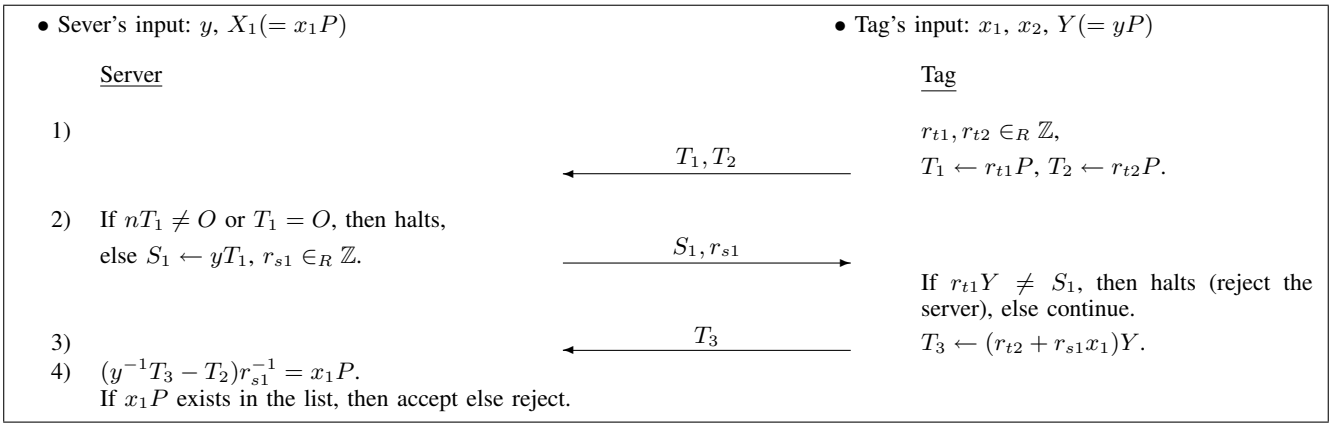

Fig. 11. Protocol 4 Flow

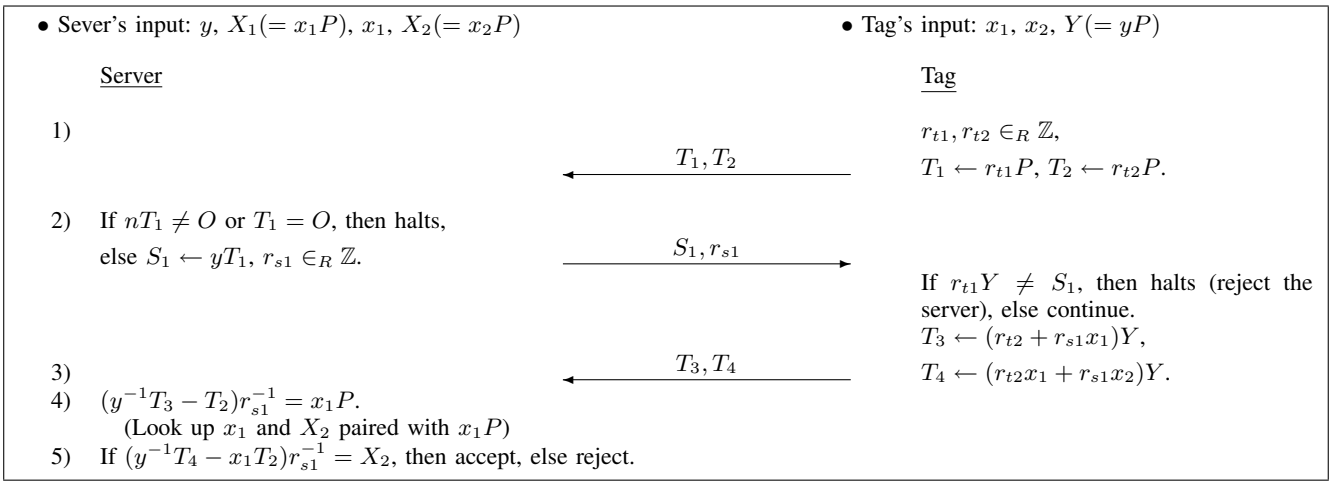

Fig. 12. Protocol 5 Flow

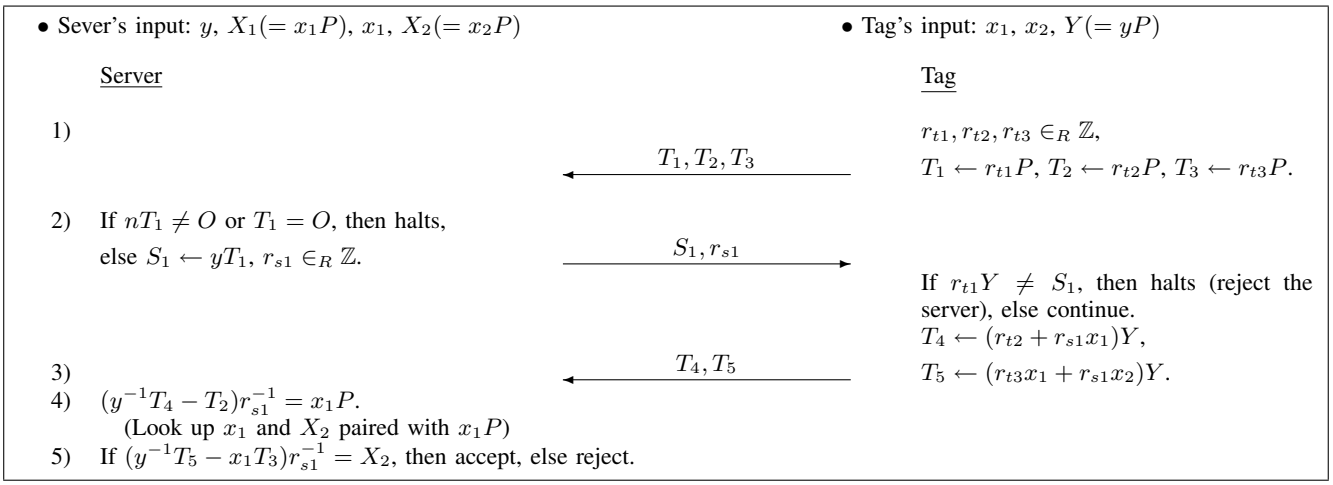

Fig. 13. Protocol 6 Flow

random numbers and each one is used only once to encrypt its ID-verifier or password-verifier.

2) Security Analysis: If we simplify the protocol 3 by setting $r_{t1} = r_{t2}$, it becomes exactly the same as the protocol 2. Therefore, the protocol 3 inherits the security properties of the protocol 2. Moreover, even if we assume that an attacker knows x_1 and x_2, two session keys, $r_{t1}yP$ and $r_{t2}x_1yP$, are independent from each other since two different random numbers make the two session keys independent. As a result, the forward/backward un-traceability is kept from the ID transfer scheme and the password transfer scheme.

C. Protocol 4, 5 and 6

The protocols 1, 2 and 3 are directly combined with the server's authentication scheme to produce the protocols 4, 5 and 6 (Fig. 11, 12 and 13). Since the server transmits the session identifier $S_1 = r_{t1}yP$ itself, the random number used in S_1 cannot be reused for other one-time session keys. Since the server's authentication scheme is combined with an independently generated random number, security proofs of the protocols 4, 5 and 6 are not necessary. They will directly inherit all the security properties of the protocols 1, 2 and 3, respectively.

D. Some Other Consideration

There should be some secret information of a tag which is not stored in the server. This will prevent duplicating tags even if the server is cracked and all the information in the server is revealed to an attacker. In the protocols 1 and 4, the server does not store x_1, and in the protocols 2, 3, 5 and 6, the server does not store x_2. Therefore, in the proposed protocols, an attacker cannot duplicate tags even if he succeeds to crack the server.

In order to simplify the computations and controls, inversions of scalars and general EC point additions/substractions are avoided on tags while they are not on the server. For example, in the protocol 2 (Fig. 8), $(y^{-1}T_2 - T_1)r_{s1}^{-1}$ requires the inversions of y and r_{s1}, and a general point substraction of $(y^{-1}T_2) - T_1$ in the server.

VI. Conclusion

In this paper, we proposed composable RFID authentication protocols by revising and expanding EC-RAC. The three components, the secure ID transfer scheme, the secure password transfer scheme, and the server's authentication, can be differently constructed depending on the required system and security properties, resulting in 6 different protocols. The proposed protocols are designed to minimize the computation amount on tags, and the security of the protocols is proved by cryptographic reductions with assumptions of the Schnorr protocols' security and the hardness of the decisional Diffie-Hellman problem.

VII. Acknowledgments

This work is supported by NSF CCF-0541472, SRC, FWO and funds from the Katholieke Universiteit Leuven.

References

[1] G. Avoine and P. Oechslin. A Scalable and Provably Secure Hash-Based RFID Protocol. In *IEEE International Workshop on Pervasive Computing and Communication Security - Persec'05*, 2005.

[2] L. Batina, J. Guajardo, T. Kerins, N. Mentens, P. Tuyls, and I. Verbauwhede. Public-Key Cryptography for RFID-Tags. In *IEEE International Workshop on Pervasive Computing and Communication Security - Persec'07*, 2007.

[3] J. Bringer, H. Chabanne, and T. Icart. Cryptanalysis of EC-RAC, a RFID Identification Protocol. In *International Conference on Cryptology and Network Security - CANS'08, Lecture Notes in Computer Science*. Springer-Verlag, 2008.

[4] M. Burmester, B. Medeiros, and R. Motta. Robust Anonymous RFID Authentication with Constant Key Lookup. In *ACM Symposium on Information, Computer and Communications Security - ASIACCS'08*. ACM, 2008.

[5] M. Burmester, T. van Le, and B. de Medeiros. Provably secure ubiquitous systems: Universally Composable RFID Authentication Protocols. In *IEEE/CreateNet International Conference on Security and Privacy in Communication Networks - SECURECOMM'06*, 2006.

[6] T. Deursen and S. Radomirović. Attacks on RFID Protocols. In *Cryptology ePrint Archive: listing for 2008 (2008/310)*, 2008.

[7] M. Feldhofer. An Authentication Protocol in a Security Layer for RFID Smart Tags. In *IEEE Mediterranean Electrotechnical Conference - IEEE MELECON'04*, 2004.

[8] 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'04, Lecture Notes in Computer Science*, volume 3156, pages 357–370. Springer-Verlag, 2004.

[9] X. Gao, Z. Xiang, H. Wang, J. Shen, J. Huang, and S. Song. An Approach to Security and Privacy of RFID System for Supply Chain. In *IEEE International Conference on E-Commerce Technology for Dynamic E-Business - CEC-East'04*, 2004.

[10] A. Juels and S.A. Weis. Defining Strong Privacy for RFID. In *Annual IEEE International Conference on Pervasive Computing and Communications - PerCom'07*, pages 342–347, 2007.

[11] Y. K. Lee, L. Batina, and I. Verbauwhede. EC-RAC (ECDLP Based Randomized Access Control): Provably Secure RFID authentication protocol. In *IEEE International Conference on RFID*, pages 97–104, 2008.

[12] Y. K. Lee and I. Verbauwhede. Secure and Low-cost RFID Authentication Protocols. In *IEEE International Workshop on Adaptive Wireless Networks - AWiN05*, pages 1–5, 2005.

[13] C. H. Lim and T. Kwon. Strong and Robust RFID Authentication Enabling Perfect Ownership Transfer. In *International Conference on Information and Communications Security - ICICS'06, Lecture Notes in Computer Science*, volume 4307, pages 1–20. Springer-Verlag, 2006.

[14] NIST. Recommended Elliptic Curves for Federal Government Use. http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf, 1999.

[15] M. Ohkubo, K. Suzuki, and S. Kinoshita. Cryptographic Approach to "Privacy-Friendly" Tags. In *RFID Privacy Workshop @ MIT*, 2003.

[16] T. Okamoto. Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. In E. F. Brickell, editor, *Advances in Cryptology - CRYPTO'92, Lecture Notes in Computer Science*, volume 740, pages 31–53. Springer-Verlag, 1992.

[17] R. Paise and S. Vaudenay. Mutual authentication in RFID: security and privacy. In *ASIAN ACM Symposium on Information, Computer and Communications Security*, pages 292–299. ACM, 2008.

[18] C.-P. Schnorr. Efficient Identification and Signatures for Smart Cards. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO'89, Lecture Notes in Computer Science*, volume 435, pages 239–252. Springer-Verlag, 1989.

[19] SECG. SEC 2: Recommended Elliptic Curve Domain Parameters. http://www.secg.org/download/aid-386/sec2_final.pdf, 2000.

[20] B. Toiruul and K. Lee. An Advanced Mutual-Authentication Algorithm Using AES for RFID Systems. *International Journal of Computer Science and Network Security*, 6(9B), September 2006.

[21] G. Tsudik. YA-TRAP: Yet another trivial RFID authentication protocol. In *IEEE International Conference on Pervasive Computing and Communications - PerCom'06*, 2006.

[22] P. Tuyls and L. Batina. RFID-tags for Anti-Counterfeiting. In D. Pointcheval, editor, *Topics in Cryptology - CT-RSA'06, Lecture Notes in Computer Science*, volume 3860, pages 115–131. Springer-Verlag, 2006.

[23] S. Vaudenay. On Privacy Models for RFID. In K. Kurosawa, editor, *Advances in Cryptology - ASIACRYPT'07, Lecture Notes in Computer Science*, volume 4833, pages 68–87. Springer-Verlag, 2007.

[24] S. A. Weis, S. E. Sarma, R. L. Rivest, , and D. W. Engels. Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. In *the First International Conference on Security in Pervasive Computing - SPC'03*, 2003.

[25] J. Wolkerstorfer. Is Elliptic-Curve Cryptography Suitable to Secure RFID Tags? In *Workshop on RFID and Light-weight Cryptography*, 2005.