Security Analysis of RAPP: An RFID Authentication Protocol based on Permutation

Wang Shao-hui[1,2,3], Han Zhijie[1,2], Liu Sujuan[1,2], Chen Dan-wei[1,2]

{1.College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210046, China
2. Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing, Jiangsu 210003, China
3. Network and Data Security Key Laboratory of Sichuan Province }

Abstract. One of the key problems in Radio Frequency Identification(RFID) is security and privacy. Many RFID authentication protocols have been proposed to preserve security and privacy of the system. Nevertheless, most of these protocols are analyzed and it is shown that they can not provide security against some RFID attacks. RAPP is a new ultralightweight authentication protocol with permutation. In RAPP, only three operations are involved: bitwise XOR, left rotation and permutation. In this paper, we give an active attack on RAPP. We first collect some authentication messages through impersonating valid tag and readers; Then we forge valid reader to communicate with the tag about 2^{30} times. Using the property of the left rotation and permutation operation, we can deduce the relationship of bits of random number or secret keys at different positions, thus obtain all the secret shared by the reader and the tag.

Keywords: RFID; Lightweight Authentication; Permutation; Privacy; Active Attack

1. Introduction

Radio Frequency Identification (RFID) systems are used for automated identification of objects and people. Applications that use RFID technology include warehouse management, logistics, railroad car tracking, product identification, library books check-in/check-out, asset tracking, passport and credit cards, etc. Most of the RFID systems comprise of three entities: the tag, the reader and the back-end database. The tag is a highly constrained microchip (with antenna) that stores the unique tag identifier and other related information about an object. The reader is a device that can read/modify the stored information of the tags and transfer these data to a back-end database, with or without modification. Back end database stores this information and will keep track of the data exchanged by the reader [1].

One of the key problems in RFID is privacy and security. It is typically critical to the communication between the reader and the tag because wireless transmissions are more vulnerable to malicious adversaries. The possible security threats to RFID systems include denial of service (DoS), man in the middle (MIM), counterfeiting, spoofing, eavesdropping, traffic analysis, traceability, de-synchronization etc.

An effective and flexible way to assure privacy and security is to adopt authentication protocols. The low cost deployment demand for RFID tags forces the lack of resources for performing true cryptographic operations to provide security. It is worthwhile to study ultralightweight authentication protocols which require tags to involve only simple bitwise operations such as bitwise XOR, bitwise OR, bitwise AND and rotation.

Providing light weight security in RFID systems is not a trivial task. Several ultralightweight protocols have already been proposed. However, they all have certain flaws and vulnerabilities. Vajda and L.Buttyan [2] have proposed a set of extremely lightweight challenge response authentication algorithms. These can be used for authenticating the tags, but they may be easily attacked by a powerful adversary. Juels [3] proposed a solution based on the use of pseudonyms, without using any hash function. But after a set of authentication sessions, the list of pseudonyms will need to be reused or updated through an out-of-band channel, which limits the practicality of this scheme.

Peris-Lopez *et al.* proposed a family of ultralightweight mutual authentication protocols (e.g., [4] and [5]). But later it was reported that these protocols are vulnerable to desynchronization attack and full-disclosure attack [6]. In addition to this there are other lightweight mutual authentication protocols proposed in the literature and attacks have been successfully mounted on all of these as demonstrated in literature [7-10]. Chien introduced another ultralightweight protocol called SASI [11] to provide strong authentication and strong integrity. However, vulnerabilities have also been illustrated such as tag traceability, de-synchronization and secret disclosure attack [12-15]. [16] presented Gossamer protocol which is inspired by SASI and tries to be devoid of the weakness of SASI. Nonetheless, the de-synchronization attack in [14] still works.

Recently, Yun Tian et.al.[17] propose a new ultralightweight RFID authentication protocol with permutation (called *RAPP*). They introduce the permutation operation to break the orders of the bits. Moreover, in their scheme, the last messages are sent by the reader rather than by the tag to resist de-synchronization attacks. This also economizes the storage of the tag. Tags in *RAPP* involve three operations: bitwise XOR, left rotation and permutation.

In this paper, we give a security analysis of this new proposed protocol—*RAPP*. An active attack is proposed, in which we first collect some authentication messages through impersonating valid tag and reader; Then we forge valid reader to communicate with the tag. The analysis shows when querying about 2^{30} times with the tag, we can deduce all the secrets shared by the reader and the tag utilizing the property of the left rotation and permutation operations. The rest of the paper is organized as follows: *RAPP* is briefly illustrated in section 2. Section 3 describes the detail security analysis of this new protocol with permutation, and shows how to extract all the secrets shared by the reader and the tag. Section 4 gives the complexity of the attack and an example of our attack with reduced length is presented. Conclusion is given in section 5.

2. RAPP Scheme

In this section, we give a brief description of *RAPP*. In this protocol, costly operations such as multiplications and hash evaluations are not used at all, and random number generation is only done at the reader's side. All the variables in the protocol are 96 bit. Frequently used notations in this paper are listed below:

ID :	Tag's unique identifier
$IDS^{(n)}$:	Tag's dynamic pseudonym at the n^{th} successful run of protocol.
$K_1^{(n)}$, $K_2^{(n)}$, $K_3^{(n)}$:	Secret keys shared at the n^{th} successful run of protocol.
n_1, n_2 :	Pseudorandom numbers generated by Reader.
A,B,C,D,E :	Messages transferred between Reader and Tag.
$\oplus$	Bitwise XOR operation

In *RAPP*, the tags and readers only involve 3 operations: bitwise XOR $\oplus$, left rotation $Rot(x,y)$ and permutation $Per(x,y)$. Suppose x and y are two 96-bit strings, $Rot(x,y)$ is defined to left rotate x by $wt(y)$ bits, where $wt(y)$ is the Hamming weight of y.

The permutation operation $Per(x,y)$ is defined as follows:

Definition 2.1: Suppose x and y are two 96-bit strings, where

$$x = x_1x_2......x_{96},\ \ x_i \in \{0,1\}, i = 1,2,...,96;\ \ y = y_1y_2......y_{96},\ \ y_i \in \{0,1\}, i = 1,2,...,96$$

Moreover, the Hamming weight of y, $wt(y)$, is m ($0 \le m \le 96$) and

$$y_{k_1} = y_{k_2} = ... = y_{k_m} = 1, y_{k_{m+1}} = y_{k_{m+2}} = ... = y_{k_{96}} = 0,$$

where $1 \le k_1 < k_2 < ... < k_m \le 96$ and $1 \le k_{m+1} < k_{m+2} < ... < k_{96} \le 96$. Then, the permutation of x according to y, denoted as $Per(x,y)$, is $Pert(x,y) = x_{k_1}x_{k_2}...x_{k_m}x_{k_{96}}x_{k_{95}}...x_{k_{m+2}}x_{k_{m+1}}$.

In *RAPP*, every tag shares a fixed and unique identifier (ID) with the reader. At the n^{th} authentication, the tag and the reader share a pseudonym $IDS^{(n)}$ and three secrets ($K_1^{(n)}$, $K_2^{(n)}$, $K_3^{(n)}$), which will update to $IDS^{(n+1)}$, $K_1^{(n+1)}$, $K_2^{(n+1)}$, $K_3^{(n+1)}$ if authentication is successful. Every authentication contains three rounds: tag identification, mutual authentication and IDS, secrets updating, which is presented as follows:

(I.) Tag Identification. After receiving the "Hello" message from the reader, the tag sends the $IDS^{(n)}$ to the reader, which will look up the tags in the database with the same pseudonym and get the corresponding information.

(II) Mutual Authentication. The reader and tag will authenticate to each other through the following step:

Step1. Reader first generates a random number n_1, computes and sends the tag the messages (A,B) as equation (1) and (2) . The tag can deduce the random number n_1 through message A, and make sure whether the reader is valid via checking the correctness of message B:

$$A = Per(K_2^{(n)}, K_1^{(n)}) \oplus n_1 \qquad (1)$$

$$B = Per(K_1^{(n)} \oplus K_2^{(n)}, Rot(n_1, n_1)) \oplus Per(n_1, K_1^{(n)}) \qquad (2)$$

Step2. If the reader is valid, the tag sends back the answer message C to authenticate himself:

$$C = Per(K_1^{(n)} \oplus n_1, n_1 \oplus K_3^{(n)}) \oplus ID \qquad (3)$$

Step3. After authenticating the validity of the tag, Reader generates another random number n_2, computes and sends the tag the messages (D,E) as follow. The tag can deduce the random number n_2 through message D, and make sure that n_2 is not changed via checking the correctness of message E:

$$D = Per(K_3^{(n)}, K_2^{(n)}) \oplus n_2 \qquad (4)$$

$$E = Per(K_3^{(n)}, Rot(n_2, n_2)) \oplus Per(n_1, K_3^{(n)} \oplus K_2^{(n)}) \qquad (5)$$

(III) IDS and Secrets Updating. After authenticating successfully, the reader and tag will update the pseudonym IDS and secrets in the follow way:

$$IDS^{(n+1)} = Per(IDS^{(n)}, n_2 \oplus n_1) \oplus K_1^{(n)} \oplus K_2^{(n)} \oplus K_3^{(n)} \qquad (6)$$

$$K_1^{(n+1)} = Per(K_1^{(n)}, n_1) \oplus K_2^{(n)} \qquad (7)$$

$$K_2^{(n+1)} = Per(K_2^{(n)}, \oplus n_2) \oplus K_1^{(n)} \qquad (8)$$

$$K_3^{(n+1)} = Per(K_3^{(n)}, n_2 \oplus n_1) \oplus IDS^{(n)} \qquad (9)$$

3. Security Analysis of RAPP

In the section, we give the security analysis of the *RAPP* and some superscripts are omitted for convenience in the following paper. We denote by $[x]_i$ the bit at position i in x and $\overline{x}_{i,i+1}$ the string with the same bit as x except for the bits in position i and $i+1$. $\overline{0}_{i,i+1}$ means the string with all the bits are 0 except that the bit in position i and $i+1$ is 1.

As to the operations $Per(\cdot,\cdot)$ and $Rot(\cdot,\cdot)$, we can get the following observations:

Observation 1. As to any two 96-bit strings x, y and z, it is easy to see that operation $Per(\cdot,\cdot)$ has the property: $Per(x,y)\oplus Per(z,y)=Per(x\oplus z,y)$.

Observation 2. As to any $i=1,2,...,95$, if $[x]_i$ and $[x]_{i+1}$ are different, $\overline{x}_{i,i+1}$ and x have the same hamming weight. If $[x]_i=[x]_{i+1}=0$, $wt(\overline{x}_{i,i+1})=wt(x)+2$; and if $[x]_i=[x]_{i+1}=1$, $wt(\overline{x}_{i,i+1})=wt(x)-2$.

Observation 3. As to any $i=1,2,...,95$, if $[x]_i$ and $[x]_{i+1}$ are different, $Rot(\overline{x}_{i,i+1},\overline{x}_{i,i+1})\oplus Rot(x,x)=\overline{0}_{s,s+1}$ for some bit position s. That is to say $Rot(\overline{x}_{i,i+1},\overline{x}_{i,i+1})$ is almost the same as $Rot(x,x)$ except for the bits in position s and $s+1$.

In addition, there is only one different number in the sequence of number i satisfying $[Rot(\overline{x}_{i,i+1},\overline{x}_{i,i+1})]_i=1$ from that of $[Rot(x,x)]_i=1$, and also one different position number in the sequence of number j satisfying $[Rot(\overline{x}_{i,i+1},\overline{x}_{i,i+1})]_j=0$ from that of $[Rot(x,x)]_j=0$.

Observation 4. From the observation 3, we can see as to any $i=1,2,...,95$, if $[x]_i$ and $[x]_{i+1}$ are different, $Per(y,Rot(\overline{x}_{i,i+1},\overline{x}_{i,i+1}))$ either equals to $Per(y,Rot(x,x))$, or has different bits in 2 positions , i.e. $Per(y,Rot(\overline{x}_{i,i+1},\overline{x}_{i,i+1}))\oplus Per(y,Rot(x,x))=\overline{0}_{s,t}$ with bit position s and t.

Example: Given $x=10100111$, we can get $\overline{x}_{2,3}=11000111$, $\overline{x}_{4,5}=10111111$, $wt(\overline{x}_{2,3})=wt(x)=5$, $wt(\overline{x}_{4,5})=wt(x)+2$. $Rot(x,x)=11110100$, $Rot(\overline{x}_{2,3},\overline{x}_{2,3})=11111000$ and $Rot(\overline{x}_{4,5},\overline{x}_{4,5})=11011111$. $Rot(\overline{x}_{2,3},\overline{x}_{2,3})$ has the different bits from $Rot(x,x)$ at bit position 5 and 6, while $Rot(\overline{x}_{4,5},\overline{x}_{4,5})$ has 4 different bits from $Rot(x,x)$. At position 1, 2, 3, 4, 5, the bit of $Rot(\overline{x}_{2,3},\overline{x}_{2,3})$ is 1, and position 6, 7, 8, the bit of $Rot(\overline{x}_{2,3},\overline{x}_{2,3})$ is 0; while the bit of $Rot(x,x)$ is 1 at the position 1, 2, 3 ,4 ,6, and 0 at the position 5 ,7 ,8.

In the following, we will illustrate how to deduce all the secrets shared between the tag and reader. Our attack belongs to active attack, that the adversary can impersonate a legal tag or reader to communicate with the corresponding reader or tag. We first show how to recover the random number n_1 generated by the reader; Then the secret key K_3 is deduced and secret key K_2 is obtained by analyzing some linear equations; All the other secrets including K_1 and ID can be recovered in the end.

3.1 Recovery of random number n_1

In our attack, the adversary first forges a legal reader to communicate with the tag to obtain its pseudonym IDS; Then he forges as the legal tag to authenticate himself. After receiving the pseudonym IDS, the reader generates random number n_1 to compute messages A and B as equation (1) and (2):

$$A=Per(K_2,K_1)\oplus n_1; \qquad B=Per(K_1\oplus K_2,Rot(n_1,n_1))\oplus Per(n_1,K_1)$$

To recover the random number n_1**,** the adversary forge a legal reader to launch the authentication with the valid tag. And for any $i=1,2,...,95$, the adversary calculates the message $A'=\overline{A}_{i,i+1}$, and it is easy to know the random number used by adversary is $\overline{n_1}_{i,i+1}$. From the observation 1, we can see the valid message B' must satisfy:

$$\begin{aligned}B'\oplus B&=Per(K_1\oplus K_2,Rot(\overline{n_1}_{i,i+1},\overline{n_1}_{i,i+1}))\oplus Per(\overline{n_1}_{i,i+1},K_1)\oplus Per(K_1\oplus K_2,Rot(n_1,n_1))\oplus Per(n_1,K_1)\\&=Per(K_1\oplus K_2,Rot(\overline{n_1}_{i,i+1},\overline{n_1}_{i,i+1}))\oplus Per(K_1\oplus K_2,Rot(n_1,n_1))\oplus Per(\overline{0}_{i,i+1},K_1)\end{aligned}$$

From the observation 2, 3 and 4, we can see:

1. If $[n_1]_i$ and $[n_1]_{i+1}$ are different, $Per(K_1\oplus K_2,Rot(\overline{n_1}_{i,i+1},\overline{n_1}_{i,i+1}))\oplus Per(K_1\oplus K_2,Rot(n_1,n_1))$ either equals to 0 or $\overline{0}_{s,t}$ with unknown position s and t, and $Per(\overline{0}_{i,i+1},K_1)$ equals to $\overline{0}_{u,v}$ with unknown bit position u and v. That is to say, in this conditions, $wt(B'\oplus B)\le 4$.
2. If $[n_1]_i$ and $[n_1]_{i+1}$ are the same, the permutation of $K_1\oplus K_2$ according to $Rot(\overline{n_1}_{i,i+1},\overline{n_1}_{i,i+1})$ behaves randomly compared with $Rot(n_1,n_1)$. So it is hard to predicate the changes and $wt(B'\oplus B)$ will be bigger than 4 with overwhelming probability.

From the above analysis, we proceed the following algorithm to deduce the random number n_1:

```
Algorithm 1. Recovery of the random number $n_1$
for i=1 to 95
  with all the possible $1 \le u < v < 96$, the adversary sends the tag $(A', B') = (\overline{A}_{i,i+1}, B \oplus \overline{0}_{u,v})$
    if tag sends back the message $C'$, we conclude $[x]_i \ne [x]_{i+1}$.
    Otherwise $(A', B') = (\overline{A}_{i,i+1}, B \oplus \overline{0}_{s,t} \oplus \overline{0}_{u,v})$ is send with all the possible $1 \le s < t < 96$, $1 \le u < v < 96$
      if tag sends back the message $C'$, we conclude $[x]_i \ne [x]_{i+1}$.
      Otherwise we conclude $[x]_i = [x]_{i+1}$.
```

3.2 Recovery of secret key K_3

It is easy to see we do not deduce the actual value of random number n_1 but the relationship of adjoining bit . So we can always obtain 2 possible random numbers, one starting with the bit 1, and the other with bit 0. In fact, the two possible random numbers are n_1 and $\overline{n_1}$ **.** After obtaining the random number n_1, we can get $Per(K_2, K_1) = A \oplus n_1$.

To recover the secret key K_3, we use the following observation:

Observation 5. As to any $i = 1,2,...,95$, if $[x]_i$ and $[x]_{i+1}$ are different, $Per(y, \overline{x}_{i,i+1})$ either equals to $Per(y, x)$, or has different bits at 2 positions.

The adversary first sends the actual message (A, B) to the tag, and receives the response message C. From the Algorithm 1, if $[n_1]_i$ and $[n_1]_{i+1}$ are different, the adversary can forge valid message (A', B') and tag sends back message C', which satisfies:

$$C \oplus C' = Per(K_1 \oplus \overline{n}_{1\,i,i+1}, \overline{n}_{1\,i,i+1} \oplus K_3) \oplus ID \oplus Per(K_1 \oplus n_1, n_1 \oplus K_3) \oplus ID = Per(K_1 \oplus \overline{n}_{1\,i,i+1}, \overline{n}_{1\,i,i+1} \oplus K_3)$$
$$\oplus Per(K_1 \oplus n_1, n_1 \oplus K_3) = Per(K_1 \oplus \overline{n}_{1\,i,i+1}, \overline{(n_1 \oplus K_3)}_{i,i+1}) \oplus Per(K_1 \oplus n_1, n_1 \oplus K_3)$$

From the observation 5, we can see if $[n_1 \oplus K_3]_i$ and $[n_1 \oplus K_3]_{i+1}$ are different, $wt(C \oplus C')$ must equal to 2; while if $[n_1 \oplus K_3]_i$ equals to $[n_1 \oplus K_3]_{i+1}$, $wt(C \oplus C')$ will be larger than 2 with overwhelming probability.

However, as shown in Algorithm 1, we can not obtain all the relationship of adjoining bit of $n_1 \oplus K_3$, because if $[n_1]_i$ and $[n_1]_{i+1}$ are the same, we can not forge valid message (A', B') to obtain C'. So, the adversary can forge tag and the reader again to communicate with the corresponding valid ones obtain other authentication messages $(A^{(r)}, B^{(r)}, C^{(r)})$, $r = 1,2,...,l$. The value of l will be discussed in section 4. Because $Per(K_2, K_1)$ is known, the random numbers $n_1^{(r)}, r = 1,2,...,l$ can be computed as $n_1^{(r)} = A^{(r)} \oplus Per(K_2, K_1)$.

Thus we present the following Algorithm 2 to recover the secret key K_3.

```
Algorithm 2. Recovery of the secret key $K_3$
for i=1 to 95
  if $[n_1]_i \ne [n_1]_{i+1}$
    if $wt(C \oplus C') = 2$, we conclude $[n_1 \oplus K_3]_i \ne [n_1 \oplus K_3]_{i+1}$, which means $[K_3]_i = [K_3]_{i+1}$
    Otherwise, we conclude $[n_1 \oplus K_3]_i = [n_1 \oplus K_3]_{i+1}$, which means $[K_3]_i \ne [K_3]_{i+1}$
  Otherwise find the value $t$: $[n_1^{(t)}]_i \ne [n_1^{(t)}]_{i+1}$, and call the Algorithm 1 to obtain $C^{(t)}{}'$
    if $wt(C^{(t)} \oplus C^{(t)}{}') = 2$, we conclude $[n_1 \oplus K_3]_i \ne [n_1 \oplus K_3]_{i+1}$
    Otherwise, we conclude $[n_1 \oplus K_3]_i = [n_1 \oplus K_3]_{i+1}$.
```

3.3 Recovery of secret key K_1

We should note that just as Algorithm 1, Algorithm 2 is utilized to obtain the relationship of adjoining bit of $n_1 \oplus K_3$. As to each possible random number n_1 and $\overline{n_1}$**,** there are 2 possible secret key K_3 and $\overline{K_3}$. So there are 4 possible combinations (n_1**,** K_3), (n_1**,** $\overline{K_3}$), ($\overline{n_1}$**,** K_3) and ($\overline{n_1}$**,** $\overline{K_3}$). We should try all these 4 possible combinations. We use the variable K_3 and n_1 to show how to recover the secret key K_1**.**

From the equation C, we can obtain:

$$C = Per(K_1 \oplus n_1, n_1 \oplus K_3) \oplus ID = Per(n_1, n_1 \oplus K_3) \oplus Per(K_1, n_1 \oplus K_3) \oplus ID$$

i.e. $Per(K_1, n_1 \oplus K_3) = C \oplus Per(n_1, n_1 \oplus K_3) \oplus ID$

As to different s and t, we can get :

$$Per(K_1, n_1^{(s)} \oplus K_3) \oplus Per(K_1, n_1^{(t)} \oplus K_3) = C^{(t)} \oplus C^{(s)} \oplus Per(n_1^{(s)}, n_1^{(s)} \oplus K_3) \oplus Per(n_1^{(t)}, n_1^{(t)} \oplus K_3)$$

Secret key K_3 and all the random numbers $n_1^{(i)}, i = 1,2,...,l$ are known, so the right part of the above equation can be computed. $Per(K_1, n_1^{(s)} \oplus K_3)$ and $Per(K_1, n_1^{(t)} \oplus K_3)$ are two different permutations of secret key K_1. Thus the left part of the equation involves the relationship of bit at different bit position and the linear equations can be set up. However we do not solve the linear equations, we can obtain the relationship of bit of K_1 at different positions from the equations.

3.4 Recovery of all the other secrets

After obtaining the random number n_1 and secret keys K_1 and K_3, the identifier ID can be deduced using equation (3), and the secret key of K_2 can be computed through the equation (1). In addition, we can use the messages $(A^{(r)}, B^{(r)}, C^{(r)})$, $r = 1,2,...,m$ to check whether the possible guessing is right or not.

4. Experiment Results

In this section, we first give an attack example with all the variables having 24 bits, and then the general complexity of our attack is analyzed.

4.1 An Example with Reduced Length

Here we give an example with reduced length. We take the identifier $ID = 0x86c76a$, three secrets keys $K_1 = 0xa649a0, K_2 = 0x3e8426, K_3 = 0x15f49b$, and the first random number chosen $n_1 = 0xc617b3$. Thus we can compute:

$$Per(K_2, K_1) = 011100001011000010010110, \quad K_2 \oplus K_1 = \; 100110001100110110000110$$

$$Rot(n_1, n_1) = 111101100111100011000010, \quad Per(n_1, K_1) = \; 101100111110010111000001$$

$$Per(K_2 \oplus K_1, Rot(n_1, n_1)) = 100100100110101000101101$$

So the messages the reader generated are:

$$A = 101101101010011100100101, \quad B = 001000011000111111101100,$$

The attack procedure is present briefly as follows:

Step1 Recovery the random number n_1. when $i = 1$, we know now $\overline{n}_{1\,1,2} = 0x0617b3$, $Rot(\overline{n}_{1\,1,2}, \overline{n}_{1\,1,2}) =$ 101111011001100000110000, $Per(K_2 \oplus K_1, Rot(\overline{n}_{1\,1,2}, \overline{n}_{1\,1,2})) = 101100101000110011010100$, and $Per(\overline{n}_{1\,1,2}, K_1) =$ 001100111110010111000000. The message B' should equal to 100000010110100100010100, and $wt(B \oplus B') = 12$. So as to the Algorithm 1, the adversary can not compute the valid B' to authenticate himself. We can conclude $[n_1]_1 = [n_1]_2$.

In the table 1, as i from 1 to 12, we list the corresponding values with the new random number $\overline{n}_{1\,i,i+1}$. We can see when $i = 2,5,7,11,12$, $wt(B \oplus B') \le 4$, and we can forge a valid B' from the Algorithm 1. So $[n_1]_i \ne [n_1]_{i+1}$ when $i = 2,5,7,11,12$.

Table 1 part of the values with new random number $\overline{n}_{1\,i,i+1}$

i	$\overline{n}_{1\,i,i+1}$	$Per(K_2 \oplus K_1, Rot(\overline{n}_{1\,i,i+1}, \overline{n}_{1\,i,i+1}))$	$Per(\overline{n}_{1\,i,i+1}, K_1)$	B'	$wt(B \oplus B')$
1	0x0617b3	101100101000110011010100	001100111110 010111000000	100000010110 100100010100	12
2	0xa617b3	100100100110101000101101	111100111110 010111000000	011000011000 111111101101	2
3	0xf617b3	101101100101010100011000	111100111110 010111000011	010001011011 000011011011	14
4	0xde17b3	101101101101010100010000	101100111110 010111000111	000001010011 000011010111	14
5	0xca17b3	100100100110101000101101	100100111110 010111000101	000000011000 111111101000	2
6	0xc017b3	101100101100110000110100	100000111110 010111000001	001100010010 100111110101	8

7	0xc517b3	100100100110101000101101	101000111110 010111001001	001100011000 111111100100	2
8	0xc797b3	101101100110010100110000	101100111110 010111011001	000001011000 000011101001	8
9	0xc6d7b3	101101100100010101110000	101110111110 010111010001	000011011010 000010100001	12
10	0xc677b3	101101100100010101110000	101110111110 010111100001	000011011010 000010010001	14
11	0xc627b3	100100100110101000101101	101100111110 010110100001	001000011000 111110001100	2
12	0xc60fb3	100100100110011000101101	101101111110 010110000001	001001011000 001110101100	4

Finally We can conclude that $[n_1]_1 = [n_1]_2 \neq [n_1]_3 = [n_1]_4 = [n_1]_5 \neq [n_1]_6 = [n_1]_7 \neq [n_1]_8 = [n_1]_9 = [n_1]_{10} = [n_1]_{11} \neq [n_1]_{12} \neq [n_1]_{13} \neq [n_1]_{14} = [n_1]_{15} = [n_1]_{16} = [n_1]_{17} \neq [n_1]_{18} \neq [n_1]_{19} = [n_1]_{20} \neq [n_1]_{21} = [n_1]_{22} \neq [n_1]_{23} = [n_1]_{24}$ **.** So the two possible random numbers are $n_1 =$ **110001100001011110110011** and $\overline{n_1} =$ **001110011110100001001100.**

Step2 Recovery the secret key K_3 **.** As to the original message A and B, we know $K_1 \oplus n_1 =$ 011000000101111000010011, $K_3 \oplus n_1 =$ 110100111110001100101000, $Per(K_1 \oplus n_1, n_1 \oplus K_3)$ =01000010100011 0100111001，and the response message send by the tag is $C =$110001000100101001010011.

As to the algorithm 2, we know when $i = 2,5,7,11,12$, the adversary can forge valid authentication message A' and B', and the tag will send back C'. Taken $i = 2,5,7$ as an example, we show how to deduce the relationship of K_3**:**

$i = 2$: $K_3 \oplus \overline{n}_{1\,2,3} =$ 101100111110001100101000, $K_1 \oplus \overline{n}_{1\,2,3} =$ 000000000101111000010011, $Per(K_1 \oplus \overline{n}_{1\,2,3}, \overline{n}_{1\,2,3} \oplus K_3) =$ 000000101000110100111000, and $C' =$ 100010001001010010100010. Because $wt(C \oplus C') = 2$, we conclude $[K_3 \oplus n_1]_2 \neq [K_3 \oplus n_1]_3$. Because $[n_1]_2 \neq [n_1]_3$**,** $[K_3]_2 = [K_3]_3$.

$i = 5$: $K_3 \oplus \overline{n}_{1\,5,6} =$ 110111111110001100101000, $K_1 \oplus \overline{n}_{1\,5,6} =$ 011011000101111000010011, $Per(K_1 \oplus \overline{n}_{1\,5,6}, \overline{n}_{1\,5,6} \oplus K_3) =$ 010110000101000110100111 1, and $C' =$ 110111100110010000100101. Because $wt(C \oplus C') = 12$, we conclude $[K_3 \oplus n_1]_5 = [K_3 \oplus n_1]_6$. Because $[n_1]_5 \neq [n_1]_6$**,** $[K_3]_5 \neq [K_3]_6$

$i = 7$: $K_3 \oplus \overline{n}_{1\,7,8} =$ 110100001110001100101000, $K_1 \oplus \overline{n}_{1\,7,8} =$ 011000110101111000010011, $Per(K_1 \oplus \overline{n}_{1\,2,3}, \overline{n}_{1\,2,3} \oplus K_3) =$ 010010100011010011111001, and $C' =$ 110011001111001110010011. Because $wt(C \oplus C') = 8$, we conclude $[K_3 \oplus n_1]_7 = [K_3 \oplus n_1]_8$. Because $[n_1]_7 \neq [n_1]_8$**,** $[K_3]_7 \neq [K_3]_8$.

To obtain all the relationship of bit in conjoining position, we need to collect other random numbers and we do not show this in detail here. The relationship of bit at different position that we can get is:

$$[K_3]_1 = [K_3]_2 = [K_3]_3 \neq [K_3]_4 \neq [K_3]_5 \neq [K_3]_6 \neq [K_3]_7 \neq [K_3]_8 = [K_3]_9 = [K_3]_{10} = [K_3]_{11} =$$
$$[K_3]_{12} \neq [K_3]_{13} \neq [K_3]_{14} \neq [K_3]_{15} = [K_3]_{16} \neq [K_3]_{17} \neq [K_3]_{18} = [K_3]_{19} \neq [K_3]_{20} = [K_3]_{21} \neq [K_3]_{22} \neq [K_3]_{23} = [K_3]_{24}$$

So the two possible secret key K_3 are $K_3 =$ **000101011111010010011011** and $\overline{K_3} =$ **111010100000101101100100.**

Step3 Recovery the secret key K_1**.** Here we get 4 possible combinations (n_1, K_3), $(\overline{n_1}, K_3)$, $(n_1, \overline{K_3})$ and $(\overline{n_1}, \overline{K_3})$. Suppose we get another authentication messages $A^{(1)}$, $B^{(1)}$ and $C^{(1)}$ from the valid reader and tag. The new random number chosen is $r = 0x49377a$, and $A^{(1)}$ =001110011000011111101100, $B^{(1)}$=100100110111011110111110 and $C^{(1)}$=001100000000110010010101.

We only choose (n_1, K_3) as an example. Now We know $Per(K_2, K_1) =$011100001011000010010110, and we can get the new random number is $r = 0x49377a$. Set $K = K_1 \oplus n_1$ and $K^* = K_1 \oplus r$, then we have:

$$C \oplus C' = Per(K, K_3 \oplus n_1) \oplus Per(K^*, K_3 \oplus r) = K_{1,2,4,7,8,9,10,11,15,16,19,21,24,23,22,20,18,17,14,13,12,6,5,3} \oplus$$
$$K^*_{2,4,5,6,9,10,15,16,17,18,19,24,23,22,21,20,14,13,12,11,8,7,3,1} = 111101000100011011000110.$$

$K_{1,2,4,7,8,9,10,11,15,16,19,21,24,23,22,20,18,17,14,13,12,6,5,3}$ means the permutation of K according to 1,2,4,7,…,6,5,3. So we can get $[K]_1 \oplus [K^*]_2 = [n_1]_1 \oplus [K_1]_1 \oplus [r]_2 \oplus [K_1]_2 = 1$, Because $[n_1]_1 = 1, [r]_2 = 1$, so $[K_1]_1 \oplus [K_1]_2 = 1$. Thus we can get the relationship of bit at different positions in the same way.

Usually we can not get all the relationship we need. In the example, we can not get the relationship of bit with position 19 and 20. At that time, we need to choose another messages to try the above procedure to get the relationship of bit in position 19 or 20 with other bit positions.

As the computation of the other secret and identifier is straightforward, we do not discuss here.

4.2 Complexity Analysis of the Attack

Here we will analysis the complexity of our attack. There are two factors we should consider:

1. The number(l) of authentication messages that the adversary need to collect through impersonating valid tag or reader. If the number n_1 is chosen randomly, we know for any $i = 1,2,...,95$, $pr([n_1]_i = [n_1]_{i+1}) = 0.5$. For l random number, the probability that the bit at position i equals to the bit at position $i+1$ is $(0.5)^l$. Take $l = 10$, $(0.5)^l = 0.00097$. That is to say, when collecting 10 authentication messages, we can find $[n_1]_i \neq [n_1]_{i+1}$ with overwhelming probability for any $i = 1,2,...,95$.

2. the number we need to query the tag. To recover the random number and secret key K_3, we need to send the forged messages (A', B') to the tag to deduce the relationship of bit in different positions. From the Algorithm 1 and algorithm 2, we can conclude the number is about $48 \times C_{96}^2 \times C_{96}^2 \approx 2^{30}$.

5. Conclusion

In this paper, we give an active attack on *RAPP*, a new ultralightweight authentication protocol with permutation. We first collect some authentication messages through impersonating valid tag and readers; Then we forge valid reader to communicate with the tag about 2^{30} times. Using the property of the left rotation and permutation operation, we can deduce relationship of bits of random number or secret keys at different positions, thus obtain all the secret shared by the reader and the tag. In practice, the number needed to query the tag is much larger. How to reduce the analysis complexity will be considered in the future work.

Acknowledgements

This work is supported by the Priority Academic Program Development of Jiangsu Higher Education Institutions(PAPD), National Natural Science Funds (Grant No.60903181) and Nanjing University of Post and Telecommunication Funds (Grant No.NY208072).

REFERENCE

1. Hunt, V.D., Puglia, A., Puglia, M.: RFID: A Guide to Radio Frequency Identification. Wiley-Inter science (2007).
2. Vajda, I., Buttyan, L.: Lightweight authentication protocols for low-cost RFID tags. In: Proc. of UBICOMP 2003 (2003)
3. Juels, A.: Minimalist Cryptography for Low-Cost RFID Tags (Extended Abstract). In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 149–164. Springer, Heidelberg (2005)
4. P. Peris-Lopez, J. C. Hernandez-Castro, J. M. E. Tapiador, and A. Ribagorda. LMAP: a real lightweight mutual authentication protocol for low-cost RFID tags. in Proc. 2006 Workshop RFID Security.
5. P. Peris-Lopez, J. C. Hernandez-Castro, J. M. E. Tapiador, and A. Ribagorda. M2AP: a minimalist mutual-authentication protocol for lowcost RFID tags. in Proc. 2006 International Conference on Ubiquitous Intelligence and Computing, pp. 912–923.
6. T. Li and G. Wang. Security analysis of two ultra-lightweight RFID authentication protocols. in Proc. 2007 IFIP RC-11 International Information Security Conference, pp. 109–120.
7. Sadighian, Jalili, R.: Afmap: Anonymous forward-secure mutual authentication protocols for rfid systems. In: Third IEEE International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2009), pp. 31–36 (2009)
8. Sadighian, Jalili, R.: Flmap: A fast lightweight mutual authentication protocol for rfid systems. In: 16th IEEE International Conference on Networks (ICON 2008), New Delhi, India, pp. 1–6 (2008)
9. Safkhani, M., Naderi, M., Bagher, N.: Cryptanalysis of AFMAP. IEICE Electronics Express 7(17), 1240–1245 (2010)
10. Bárász, M., Boros, B., Ligeti, P., Lója, K., Nagy, D.: Passive Attack Against the M2AP Mutual Authentication Protocol for RFID Tags. In: First International EURASIP Workshop on RFID Technology, Vienna, Austria (2007)
11. Chien, H.-Y.: SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity. IEEE Transactions on Dependable and Secure Computing 4(4), 337–340 (2007)
12. T. Cao, E. Bertino, and H. Lei. Security analysis of the SASI protocol. IEEE Trans. Dependable and Secure Computing, vol. 6, no. 1, pp. 73–77, Jan.-Mar. 2009.
13. R. C.-W. Phan. Cryptanalysis of a new ultralightweight RFID authenticaion protocol—SASI. IEEE Trans. Dependable and Secure Computing, vol. 6, no. 4, pp. 316–320, Oct.-Dec. 2009.
14. H.-M. Sun, W.-C. Ting, and K.-H. Wang. On the security of Chien’s ultralightweight RFID authentication protocol. IEEE Trans. Dependable and Secure Computing, vol. 8, no. 2, pp. 315–317, Mar.-Apr. 2011.
15. P. D’Arco and A. De Santis. On ultralightweight RFID authentication protocols. IEEE Trans. Dependable and Secure Computing, vol. 8, no. 4, pp. 548–563, July-Aug. 2011.
16. P. Peris-Lopez, J. C. Hernandez-Castro, J. M. E. Tapiador, and A. Ribagorda, “Advances in ultralightweight cryptography for low-cost RFID tags: Gossamer protocol,” in Proc. 2008 International Workshop on Information Security Applications, pp. 56–68.
17. Y. Tian, G. Chen, and J. Li. A New Ultralightweight RFID Authentication Protocol with Permutation. IEEE Communications Letters, Vol. 16, No. 5, May 2012, pp.702-705.