A Wireless Key Generation Algorithm for RFID System Based on Bit Operation

Rui Xie[1,2], Jie Ling[1], and Dao-Wei Liu[1]
(Corresponding author: Rui Xie)
School of Computers, Guangdong University of Technology[1]
No. 100, Waihuan Xi Road, Panyu District, Guangzhou 510006, China
School of Automation, Guangdong University of Technology, Guangzhou 510006, China[2]
(Email: xr_1977@126.com)
(Received Oct. 19, 2017; Revised and Accepted Mar. 18, 2018)

Abstract

Radio Frequency Identification (RFID) system consists of three parts: tags, reader and database. Because of the advantages of low cost, long life and easy deployment, RFID technology has been widely used in logistics, supply chain management system and other fields. With the widespread use of RFID technology, a number of security problems occur. The prerequisite for further promotion of RFID technology is to ensure the security of shared keys stored between legitimate readers and tags. However, because the tags and reader communicate in wireless channel, the shared key written directly to the tag by the reader is easily intercepted by the attacker. Because the computing power and the storage space of tags in RFID are limited, the key generation protocols based on cryptographic can not be used in RFID systems. In addition, the shared key written by the tag manufacturer will result in a shared key escrow problem, and the user can not customize the shared key as well. For the above reasons, the secure generation of the shared key in RFID system is somewhat difficult. The paper abandons the general way that the shared private key has been set to the tag before leaving the factory, then uses the method of dynamically generating the shared key in the application between the tag and the reader through the wireless communication, and proposes a wireless generation algorithm WKGA-BO (Wireless Key Generation Algorithm for RFID System Based on Bit Operation) for private key in RFID system based on bitwise operation. WKGA-BO uses the bit replacement operation and the self-combinatorial cross-bit operation to encrypt the transmission information. The random number is generated by the reader to ensure the freshness of the shared key, and the methods of generating the random number on the tag side is discarded, thus reducing the computational complexity. Finally, a comprehensive security analysis demonstrates the security and reliability of WKGA-BO. A comprehensive performance analysis shows the feasibility of WKGA-BO in the existing RFID system.

Keywords: RFID; Sac Internet of Things; Wireless Key Generation

1 Introduction

RFID is a technology that uses non-physical contact to achieve object recognition and data exchange. It emerged in the last century, and had been widely promoted and applied in the late nineties of the 20^{th} century [16, 20]. Radio Frequency Identification (RFID) system consists of three parts: tags, reader and database. Because of the advantages of low cost, long life and easy deployment, RFID technology has been widely used in logistics, supply chain management system and other fields [17, 21].

RFID tags are composed of coupling elements and chips, and are embedded in the object, which is used to identify the target object. According to the tag's own characteristics and operation modes, it can be divided into two categories: one is active tag; another is passive tag [1, 27]. Active tag generally carries its own batteries, so it doesn't need to use the energy of the reader. It can also take the initiative to the reader to send information, to complete some of the work independently. Compared to the passive tag, the active tag works much larger. However, because the active tag carry its own power, there are disadvantages of high production cost, too large volume, heavy quality, excessive power consumption and so on, which restricts the development of such tags [6, 18]. Because the passive tag doesn't carry its own power, and the energy required in the work process can be only obtained through the carrier sent by the reader, so the work scope of this tag is very small [19, 23].

In the existing RFID systems, the most commonly used tags are passive tags. Based on the above description, it is known that the way that the passive tag works will easily expose the information stored in the tags and even expose the privacy information stored on the reader. In order to

solve the above problems, privacy protection certification has become the most commonly used security mechanism between the tag and reader, whose main idea is using a shared private key for bidirectional authentication and identification between the tag and the reader [7, 22, 30].

Privacy protection is mainly based on the security and reliability of the shared private key, but the traditional method of generating the shared private key has some flaws. The traditional method of generating the shared private key is that the producer defaults the shared key to the tag, but it'll cause the tag escrow problems, and the user can not generate his own trusted shared private key according to his own needs [9,28]. It is very challenging to securely generate keys on RFID tags. In addition to the reasons described above, there are the following factors. First, passive tag does not have a physical interface, so it can not be connected with other devices, and it can not be used to generate the shared private key through the physical connection [29]. Second, if the reader directly write the shared private key to the tag through the wireless, then because the reader can read and write a larger scope, it's easier for attackers to obtain the shared private key by the means of eavesdropping [12]. Third, the computing ability of the tag is very limited, so the more complex traditional cryptographic calculations are disabled, which makes the existing shared private key generation protocol based on cryptographic unable to be applied to the tag [14].

In this paper, we innovatively propose an algorithm WKGA-BO to generate the shared private key wirelessly based on bitwise operation in the RFID system for the problems described above. The basic idea of WKGA-BO is as follows. The tag which does not initialize the shared private key sends the fragment of shared private key to the reader. When the tag receives another part of the shared private key fragment generated by the reader, the tag can generate the shared private key by mixing the key fragments. The information about the shared private key fragment between the tag and the reader is transmitted over the ultra-lightweight bitwise operation, and the tag does not only superimpose the shared private key fragments simply to obtain the final shared private key. So even if the attacker obtains the shared private key fragments' information exchanged between the tag and the reader through eavesdropping, it is impossible to restore or derive the shared private key, thus realizing the wireless generation of the shared private key between the tag and the reader.

The first chapter of this article is the introduction about the background of RFID technology and its existing problems, which leads to the focus of this study. The second chapter provides a comprehensive introduction to the current work in the shared private key generation in the RFID system. The third chapter introduces the mathematical knowledge and the meanings of the symbols that are used in the WKGA-BO design process. The fourth chapter describes the detailed design steps of the WKGA-BO systematically. The fifth chapter analyzes the security of WKGA-BO from two aspects: active attack and passive attack. In the sixth chapter, the WKGA-BO is rigorously proved and deduced by GNY formal logic. In the seventh chapter, the performance of WKGA-BO is analyzed in detail from the aspects of calculation, storage space and communication traffic. The eighth chapter summarizes the whole paper, and gives the next research direction.

2 Related Works

There are many ways to generate the shared private key in the existing wireless systems and wireless devices. We can classify the existing methods into two categories from the point of view of usefulness and cryptography. One is based on the cryptography method; the other is based on the non-cryptographic method. The advantages and disadvantages of the existing shared private key generation methods from these two types of methods are described as follows.

The method based on cryptography – this type of shared private key generation method is mainly based on the public key cryptography, such as Diffie-Hellman key agreement protocol [8]. Although this type of method can solve the problem of shared private key generation, but it needs more complex mathematical operations (mostly based on the mathematical problem of the decomposition of large number when using cryptography), so that it can not be used very well in the RFID system tags whose resources are severely limited, such as weak calculation capacity, small storage space and so on.

The method based on non-cryptographic – this type of shared private key generation method is different from the cryptographic-based method, but it uses physical methods or the characteristics of the communication channels to generate shared private key, so it does not require complicated calculation [24]. In the following, we'll describe the two types of methods in detail.

The physical methods of generating the shared private key involve two different ways. One way is to generate the shared private key by physical isolation; the other is to write the shared private key over a wired connection link. The physical isolation method generates the shared private key primarily by using a Faraday cage container to protect the communication channel between the wireless device or the wireless system from being intercepted by the attacker [15]. From the Reference[19] we know that the Faraday cage is a container made of metal mesh for shielding the wireless signal transmission. The wireless signal in the cage is shielded by the Faraday cage (the wireless signal outside the cage is not in the Faraday cage category), so that any two wireless devices within the Faraday cage and wireless systems can communicate in plain text. However, this type of the shared private key generation method must be carried out in the space of the Faraday cage, so it is limited by the size of Faraday cage space, and it can not be promoted in large-scale in

the practical application of RFID. For example, Faraday cage containers can not be accommodated in the RFID tags of the large objects such as trains, containers. At the same time, users may not be willing to use RFID tags before spending too much manpower and financial resources to build a large Faraday cage. In the case of the method of generating the shared private key over a wired connection link, the two communicating devices require a hardware interface to establish a physical electrical signal connection so that the shared private key can be generated or exchanged [25]. However, in the existing RFID systems, the tag does not have a physical interface, so it can not support this kind of circuit connection communication, and it does not apply to the existing RFID systems and equipment.

Generating the shared private key based on the characteristics of the communication channel: In Reference [4] it is found that the shared private key is generated between two wireless devices with limited CPU resources through an anonymous communication channel. However, it has been analyzed that although the computation of the method is small, it is necessary to generate a packet for each secret bit, so that the communication traffic is large. Meanwhile, large-scale passive tag can only reflect the reader's signal as a response, and can not produce random data packets, so the scheme proposed in Reference [4] does not apply to the existing RFID systems. Reference [2] has proposed a scheme for generating the shared private key by measuring a random variable in a wireless communication channel, which can be used to generate the shared private key with high entropy in two wireless devices. However, the above-mentioned scheme requires the wireless device to autonomously measure the data of random variables (such as signal strength) in the communication channel, but the tag in the existing RFID systems does not have this capability, so the proposed scheme can not be applied on a large scale.

In view of the shortcomings of the existing schemes, this paper proposes an algorithm WKGA-BO of generating the shared private key wirelessly based on bitwise operation in the RFID system. By dynamically generating the shared private key wirelessly, WKGA-BO resolves the problem that the shared private key between the tag and the reader must be pre-set in the RFID system, and the user can not customize it. Compared with other existing solutions, WKGA-BO has the following main advantages. Firstly, WKGA-BO doesn't need to provide additional physical hardware interface, and doesn't require complex cryptographic algorithms, which is able to minimize the amount of computation. Secondly, WKGA-BO dynamically generates the shared private key wirelessly, which solves the problem of key escrow caused by the factory setting for the shared private key. Thirdly, WKGA-BO uses bit operations for encryption. Because the ultra-lightweight characteristic of bit operations is able to meet the requirements of low cost, WKGA-BO can be implemented and scaled up in real RFID systems.

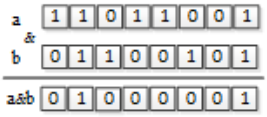

Figure 1: Bitwise AND Operation Flow Chart

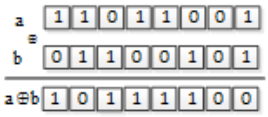

Figure 2: Bitwise XOR Operation Flow Chart

3 Related Knowledge Introduction

3.1 Bitwise AND Operation

If the two values a, b are not both 1, then the result is0; if the two values a, b are both 1, then the result is 1, where $a \in \{0,1\}^l$, $b \in \{0,1\}^l$, meaning that a and b are both binary numbers with the length of l bits. In order to facilitate the description of the symbol, we use the symbol "&" to represent bitwise AND operation [13].

The bitwise AND operation rules are described below: *0&0=0, 1&0=0, 0&1=0, 1&1=1.* Only when the values of a and b are both 1, the result of the operation is 1; otherwise, the result of the operation is0. For example, if *l=8, a=11011001, b=01100101,* then *a & b=01000001.* The specific process is shown in Figure 1.

3.2 Bitwise XOR Operation

If the two values *a,b* are not the same, the result is 1; if the two values a, b are the same, the result is0, where $a \in \{0,1\}^l$, $b \in \{0,1\}^l$, meaning that *a* and *b* are both binary numbers with the length of *l* bits. In order to facilitate the description of the symbol, we use the symbol "⊕" to represent bitwise XOR operation [13].

The bitwise XOR operation rules are described below: $0 \oplus 0=0$, $1 \oplus 0=1$, $0 \oplus 1=1$, $1 \oplus 1=0$. Only when the values of a and b are the same, the result of the operation is0; otherwise, the result of the operation is 1. For example, if *l=8, a=11011001, b=01100101,* then $a \oplus b=10111100$. The specific process is shown in Figure 2.

Bitwise XOR operation has a clever use, that is, $a \oplus b \oplus b=a$. For example, if *l=8, a=11011001, b=01100101,* then $a \oplus b=10111100$, $a \oplus b \oplus b=11011001$, so $a \oplus b \oplus b=a$. Because the bitwise XOR operation has this feature, so the WKGA-BO design process in the fourth chapter uses the bitwise XOR operation to transmit the information, and then through this feature the original information can be obtained by the corresponding encrypted information.

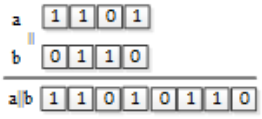

Figure 3: a || b Bitwise Concatenation Operation Flow Chart

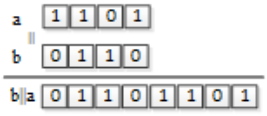

Figure 4: $b||a$ Bitwise Concatenation Operation Flow Chart

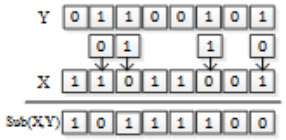

Figure 5: Bitwise Substitution Operation Flow Chart

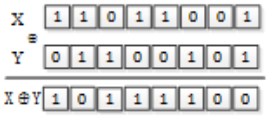

Figure 6: Bitwise Substitution Operation Flow Chart

3.3 Bitwise Concatenation Operation

If $a \in \{0,1\}^l$, $b \in \{0,1\}^l$, a and b are both binary numbers with the length of l bits, then $a||b$ or $b||a$ means concatenating the two numbers to a new binary number with the length of $2l$ bits, but the results of $a||b$ or $b||a$ are different. In order to facilitate the description of the symbol, we use the symbol "||" to represent bitwise concatenation operation [5]. For example, if l=4, a=1101, b=0110, then a || b=11010110, b || a=01101101.The specific process of $a||b$ is shown in Figure 3, and the specific process of $b||a$ is shown in Figure 4.

3.4 Bitwise Substitution Operation

In order to facilitate the description of the symbol, we use the symbol "*Sub(X, Y)*" to represent bitwise substitution operation. The definition of *Sub(X, Y)* is as follows. Let X and Y be two binary numbers with the length of l bits, $X = x_1x_2x_3 \cdots x_L$, $Y = y_1y_2y_3...y_L$, where $X \in \{0,1\}^l$, $Y \in \{0,1\}^l$. We get each bit which is one in the binary number Y, complement the corresponding bit in the binary number X and substitute it. In the binary number X, totally $n = wt(Y)$ bits are substituted, where $wt(Y)$ is the Hamming weight of Y.

When the bitwise substitution operation is implemented in the tag, using the pointer form proposed in Reference [26] will be more efficient than using a logic gate directly. Two pointers are introduced, one for P_X and the other for P_Y, where P_X points to the binary number X and pointer P_Y points to the binary number Y. When the pointer P_X starts traversing from the most significant bit of the binary number X, the pointer P_Y starts traversing from the most significant bit of the binary number Y at the same time. When the pointer P_Y points to the zero bit of binary number Y, the bit that the pointer P_X points to in the binary number X does not change. When the pointer P_Y points to the one bit of the binary number Y, the bit that the pointer P_X points to in the binary number X is substituted with its complement. Finally, *Sub(X, Y)* is the binary number X after substituted. For example, if $l = 8, X = 11011001, Y = 01100101$, then $Sub(X,Y) = 10111100$. The specific process is shown in Figure 5.

3.5 Self-assembling Cross-bit Operation

In order to facilitate the description of the symbol, we use the symbol "$Sac(Z)$" to represent self-assembling cross-bit operation. Let X, Y, Z be three binary numbers with even length l bits, $X = x_1x_2x_3...x_L, Y = y_1y_2y_3...y_L, Z = z_1z_2z_3...z_L$, where $X \in \{0,1\}^l$, $Y \in \{0,1\}^l$, $Z \in \{0,1\}^l$. X makes a bitwise XOR operation to Y and then we get the result Z. The operation $Sac(Z)$ is a new binary number W with the even length of l bits formed by the combination of the high and low bits of Z, that is, $Sac(Z) = z_1z_{L/2+1}z_2z_{L/2+2} \cdots z_{L/2}z_L$.

The self-assembling cross-bit operation can be implemented in the tag and reader as described below. Introduce two pointers, one for P_1 and the other for P_2, where the pointer P_1 points to the head of the binary number Z, and the pointer P_2 points to the end of the binary number Z. When the pointer P_1 traverses from the head of the binary number Z, the pointer P_2 starts traversing from the end of the binary number Z at the same time. The numbers traversed by the pointer P_1 are sequentially placed in the odd bits of the new binary number W, and the numbers traversed by the pointer P_2 are sequentially placed in the even bits of the new binary number W. Then through the final combination we can get the new binary number W, that is $Sac(Z)$ [10].

The self-assembling cross-bit operation only needs the shift and the bitwise OR operation, and makes the final combination, thereby reducing system throughput and storage capacity, to achieve an ultra-lightweight level. According to the different order of the assignment of the pointers, it will be combined to obtain different values, thereby increasing the difficulty of cracking. For example, if $l = 8, X = 11011001, Y = 01100101$, then $X \oplus Y = Z, Sac(Z) = 11011010$. The specific process is shown in Figure 6.

4 WKGA-BO Design

In this section, WKGA-BO is designed for the three cases of the shared private key wireless generation in the practical application: (1) Generating the shared private key for a single tag; (2) Generating each individual shared private key separately for a batch of tags at the same time; (3) Generating a shared private key for a batch of tags at the same time.

RFID system generally consists of three parts: the tag, reader, and database. Because the reader and the database communicate through the wired link, the general researches point out that the communication between the two is safe and reliable. So we regard the reader and the database as a whole, and the reader has a strong search ability.

As is the same to other related protocols,we assume that the communication between the reader and database is safe and reliable. We also suppose that the communication between the reader and the tag is unreliable and is easy to be eavesdropped by the attacker, and that the privacy information shared between the reader and the tag is safe and is what the attacker does not know in advance (such as the shared ID_L *etc.*).

4.1 Single Tag's Shared Private Key Generation

Before the start of the single tag's shared private key generation (SPKG) algorithm, the tag T_i stores the information ($ID_{i_{t_R}}$, $ID_{i_{t_L}}$), and the reader R stores the information ($ID_{i_{t_R}}$, $ID_{i_{t_L}}$) of all the tags.

The meaning of each symbol in this algorithm is given as Table 1.

Table 1: Symbol used in single tag's SPKG algorithm

Symbol	Description
T	A tag
R	A reader
DB	A database
ID_t	The tag identifier ID
ID_{t_L}	The left half of the tag identifier ID
ID_{t_R}	The right half of the tag identifier ID
Key	The generated shared private key
r	The random number generated by the reader
r_L	The left half of r
r_R	The right half of r
$\oplus$	Bitwise XOR operation
$\|\|$	Bitwise concatenation operation
&	Bitwise AND operation
$Sub(X,Y)$	Bitwise substitution operation

The flow chart of wirelessly generating the single tag's shared private key is shown in Figure 7. Table 2 shows the operation formulas that appear in this algorithm.

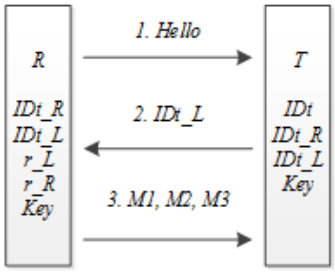

Figure 7: Single Tag's Shared Private Key Generation Flow Chart

Table 2: Formula used in single tag's SPKG algorithm

Symbol	Description
$M1$	$r_L \oplus ID_{t_R}$
$M2$	$r_R \oplus ID_{t_R}$
$M3$	$Sub(r_L\|\|ID_{t_R}, ID_{t_R}\|\|r_R)$
r'_L	$M1 \oplus ID_{t_R}$
r'_R	$M2 \oplus ID_{t_R}$
$M3'$	$Sub(r'_L\|\|ID_{t_R}, ID_{t_R}\|\|r'_R)$
Key	$Sub(r_L\&ID_{t_R}, ID_{t_R}\&r_R)$

The single tag's shared private key generation algorithm in the WKGA-BO protocol consists of four steps, as shown in Figure 7.

Step 1: The reader sends a $Hello$ message to the tag, and informs the tag to start the private key generation process.

Step 2: The tag sends ID_{t_L} as the response message to the reader.

Step 3: The reader searches the database for the result of whether ID_{t_L} exists or not. If the result exists, the reader generates a random number $r \in \{0,1\}^l$(r_L means the left half of r, and r_R means the right half of r), then calculates the values of $M1, M2, M3$, and enumerates the shared private key Key. Finally, it sends $M1, M2, M3$ to the tag. If it does not exist, $WKGA-BO$ terminates. The calculations of $M1, M2, M3, Key$ are described above.

Step 4: The tag calculates the value of r'_L, r'_R, then calculates $M3'$, and compares the value of $M3$ and $M3'$. If $M3$ is equal to $M3'$, the tag successfully verifies the reader, which meanwhile indicates that $r'_L = r_L$, $r'_R = r_R$, and then the tag starts calculating the shared private key Key. If $M3$ is not equal to $M3'$, $WKGA-BO$ terminates. The calculations of $r'_L, r'_R, M3', Key$ are described above.

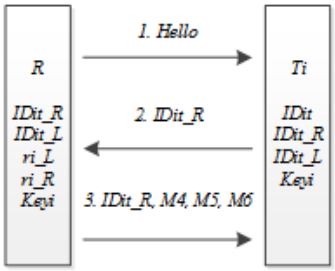

Figure 8: A Batch of Shared Private Keys Generation Flow Chart

4.2 A Batch of Shared Private Keys Generation

Before the start of a batch of SPKG algorithm, the tag T_i stores the information ($ID_{i_{t_R}}$, $ID_{i_{t_L}}$), and the reader R stores the information ($ID_{i_{t_R}}$, $ID_{i_{t_L}}$) of all the tags.

The meaning of each symbol in this algorithm is given as Tables 1 and 3.

Table 3: Symbol used in batch of SPKG algorithm

Symbol	Description
T_i	The i-th tag
ID_{i_t}	The i-th tag identifier ID
$ID_{i_{t_L}}$	The left half of ID_{i_t}
$ID_{i_{t_R}}$	The right half of ID_{i_t}
Key_i	The shared private key generated between the i-th tag and the reader
r_i	The random number generated by the reader for the i-th tag
r_{i_L}	The left half of r_i
r_{i_R}	The right half of r_i
$Sac(X)$	Self-assembling cross-bit operation

The flow chart of wirelessly generating a batch of shared private keys is shown in Figure 8. Table 4 shows the operation formulas that appear in this algorithm.

Table 4: Formula used in batch of SPKG algorithm

Symbol	Description
$M4$	$r_{i_L} \oplus ID_{i_{t_L}}$
$M5$	$r_{i_R} \oplus ID_{i_{t_L}}$
$M6$	$Sac((r_{i_L}\|\|ID_{i_{t_L}}) \ \& \ (ID_{i_{t_L}}\|\|r_{i_R}))$
r'_{i_L}	$M4 \oplus ID_{i_{t_L}}$
r'_{i_R}	$M5 \oplus ID_{i_{t_L}}$
$M6'$	$Sac((r'_{i_L}\|\|ID_{i_{t_L}}) \ \& \ (ID_{i_{t_L}}\|\|r'_{i_R}))$
Key_i	$Sac((r_{i_L}\&ID_{i_{t_L}}) \oplus (ID_{i_{t_L}}\&r_{i_R}))$

This algorithm enables the reader in the RFID system to generate individual shared private keys for a large number of different tags at the same time, and $WKGA$-BO protocol consists of four steps, as shown in Figure 8.

Step 1: The reader sends a $Hello$ message to the tag, and informs the tag to start the private key generation process.

Step 2: The tag T_i sends $ID_{i_{t_R}}$ as the response message to the reader.

Step 3: The reader searches the database for the result of whether $ID_{i_{t_R}}$ exists or not. If the result exists, the reader generates a random number $r_i \in \{0,1\}^l (r_{i_L}$ means the left half of r_i; r_{i_R} means the right half of r_i; r_i is applied to calculate the random number used by the shared private key between the reader and the tag T_i), then calculates the values of $M4, M5, M6$, and generates the shared private key Key_i. Finally, it sends $M4, M5, M6$, $ID_{i_{t_R}}$ to the tag T_i. If it does not exist, $WKGA-BO$ terminates. The calculations of $M4, M5, M6, Key_i$ are described above.

Step 4: The tag T_i compares the received value $ID_{i_{t_R}}$ with its own value ID_{t_R}. If $ID_{i_{t_R}}$ is not equal to ID_{t_R}, the tag T_i discards the message. Otherwise, the tag T_i calculates the value of r'_L, r'_R, then calculates $M6'$, and compares the value of $M6$ and $M6'$. If $M6$ is equal to $M6'$, the tag T_i successfully verifies the reader, which meanwhile indicates that $r'_{i_L} = r_{i_L}$, $r'_{i_R} = r_{i_R}$, and then the tag starts calculating the shared private key Key_i. If $M6$ is not equal to $M6'$, $WKGA-BO$ terminates. The calculations of $r'_{i_L}, r'_{i_R}, M6', Key_i$ are described above.

4.3 Group-based Shared Private Keys Generation

Before the start of group-based SPKG algorithm, the tag T_i stores the information ($ID_{i_{t_R}}$, $ID_{i_{t_L}}$), and the reader R stores the information ($ID_{i_{t_R}}$, $ID_{i_{t_L}}$) of all the tags.

The meaning of each symbol in this algorithm is given as Tables 1 and 3.

The flow chart of wirelessly generating a group of shared private keys is shown in Figure 9. Table 5 shows the operation formulas that appear in this algorithm.

This algorithm needs to generate a unique shared private key between the reader R and a group of tags $T1, T2, \cdots, T_i, \cdots, Tn$, where n is the total number of tags in the group. $WKGA-BO$ protocol consists of four steps, as shown in Figure 9.

Step 1: The reader sends a $Hello$ message to the tag group, and informs all the tags to start the private key generation process.

Step 2: The tag T_i sends $ID_{i_{t_L}}$ as the response message to the reader.

Table 5: Formula used in group-based SPKG algorithm

Symbol	Description
$M7$	$r_{i_L} \oplus ID_{i_{t_L}}$
$M8$	$r_{i_R} \oplus ID_{i_{t_L}}$
$M9$	$Sac((r_{i_L} \ \& \ r_{i_R}) \oplus ID_{i_{t_R}})$
$M10$	$Key \oplus ID_{i_{t_R}}$
r'_{i_L}	$M7 \oplus ID_{i_{t_R}}$
r'_{i_R}	$M8 \oplus ID_{i_{t_R}}$
$M9'$	$Sac((r'_{i_L} \ \& \ r'_{i_R}) \oplus ID_{i_{t_R}})$
Key	$Sac(ID_{1_{t_R}} \oplus ID_{2_{t_R}} \oplus ID_{i_{t_R}} \oplus \cdots \oplus ID_{n_{t_R}}$

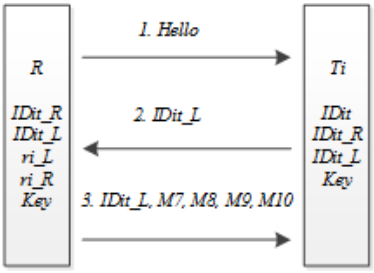

Figure 9: Group-based Shared Private Keys Generation Flow Chart

Step 3: After the reader receives the responses of all the tags, R compares all the received tags $ID_{1_{t_L}}, ID_{2_{t_L}}, \cdots, ID_{i_{t_L}}, \cdots, ID_{n_{t_L}}$ with its own tag ID_{t_L} and determines whether they are consistent or not. If they're not consistent, which indicates that some tags do not answer the reader, then R re-sends a $Hello$ message to restart the shared private key generation. If they're consistent, it indicates that all the tags have answered the reader, and that the shared private key can be generated. The reader generates a random number $r_i \in \{0,1\}^l$(r_{i_L} means the left half of r_i; r_{i_R} means the right half of r_i; r_i is the random number used by the tag T_i to verify the reader), then calculates the value of the shared private key Key and the values of $M7, M8, M9, M10$. Finally, it sends $M7, M8, M9, M10, ID_{i_{t_L}}$ to the tag T_i. The calculations of $M7, M8, M9, M10, Key$ are described above.

Step 4: The tag T_i compares the received value $ID_{i_{t_L}}$ with its own value ID_{t_L}. If $ID_{i_{t_L}}$ is not equal to ID_{t_L}, the tag T_i discards the message. Otherwise, the tag T_i calculates the value of r'_{i_L}, r'_{i_R}, then calculates $M9'$, and compares the value of $M9$ and $M9'$. If $M9$ is equal to $M9'$, the tag T_i successfully verifies the reader, which meanwhile indicates that $r'_{i_L} = r_{i_L}, r'_{i_R} = r_{i_R}$, and then the tag Ti starts calculating the shared private key $Key = M10 \oplus ID_{i_{t_R}}$. If $M9$ is not equal to $M9'$, $WKGA-BO$ terminates. The calculations of $r'_{i_L}, r'_{i_R}, M9'$, Key are described above.

The $WKGA-BO$ proposed in this paper has the following advantages. Firstly, the shared private key between the reader and the tag need not be pre-set and can be dynamically generated when being used. Secondly, the user can customize the shared private key according to his preference and requirement. Thirdly, the $WKGA-BO$ uses the ultra-lightweight bit operation to encrypt the transmitted information so that it can achieve the ultra-lightweight level, which is able to reduce the amount of calculation and achieve the goal of low cost. It's suitable for the existing RFID systems. Fourthly, the corresponding algorithms for the shared private key generation are given in different scenarios, and $WKGA-BO$ is no longer limited in real application.

5 Security

This section first gives the target of the security of $WKGA-BO$, and then gives a specific security analysis process.

5.1 Security Target

The communication channel between the tag and the reader in the RFID system can be divided into a forward channel and a backward channel. The forward channel refers to the channel from the reader to the tag; the backward channel refers to the channel from the tag to the reader. Research scholars generally believe that the forward and backward channels are not safe. The attacker can gain access to the communication between the tag and the reader by tapping the two channels. The attacker deduces the privacy information stored in the tag or reader according to what he eavesdrops. One of the WKGA-BO's security target is to ensure that the attacker can not infer any useful privacy information from the eavesdropping information.

1) Resist passive attack:
Before the shared private key is not generated, there is no trusted private key between the tag and the reader to protect the communication between the two messages, so $WKGA-BO$ must be able to resist the attacker eavesdropping, and make the attacker unable to eavesdrop the information to infer the generated shared private key. Even if the attacker can eavesdrop on the forward and backward channels, $WKGA-BO$ should make the attacker unable to infer the generated shared private key based on the information obtained from what he eavesdrops.

2) Resist active attack:
The attacker can inject the false information into the forward channel and backward channel. When the attacker injects or modifies the information into the

forward channel, the attacker is equivalent to disguise himself as the reader to send messages to the tag. This kind of attack is called the reader impersonation attack. What's more, when the attacker injects or modifies the information into the backward channel, the attacker is equivalent to disguise himself as the tag to send messages to the reader. This kind of attack is called the tag impersonation attack. WKGA-BO should be able to resist the two kinds of impersonation attacks.

5.2 Security Analysis

This section will analyze the security of $WKGA-BO$ from the aspects of resisting the passive attack and the active attack.

5.2.1 Passive Attack

The ability of $WKGA-BO$ to resist the passive attack depends on the characteristic of the random number generated by the reader and the security of $ID_{i_{t_R}}$, $ID_{i_{t_L}}$ shared between the reader and the tag. $WKGA-BO$ can resist the passive attack, if the probability of each bit (0 or 1) in the random number generated by the reader is the same, and if $ID_{i_{t_R}}$, $ID_{i_{t_L}}$ shared between the reader and the tag are secure. The *EPC Gen* 2 air interface standard has defined 16-bits pseudo-random number generators embedded in RFID tags to meet this characteristic. At the same time $WKGA-BO$ algorithm has a certain application of the scene, this section has been set in the applicable scene between the label and the reader at the time of sharing the information is safe and reliable. At the same time $WKGA-BO$ algorithm has a certain application of the scene. In this section the shared information that has been set between the tag and the reader before leaving factory is safe and reliable in the applicable scene.

In order to obtain the generated shared key, the attacker can eavesdrop on the communication between the tag and the reader. In the following we uses the single tag shared private key generation algorithm as an example, to analyze that the $WKGA-BO$ can resist the passive attack.

First, the random number are used in the calculations of$M1, M2, M3$, and the random numbers in each round of the random generation of the shared private key are different, so as to ensure that $M1, M2, M3$ in each round are various. Second, the attacker can obtain ID_{t_L}, but the calculations of $M1, M2, M3$ do not use ID_{t_L}, but the ID_{t_R}, and there is no association between ID_{t_R} and ID_{t_L}, so that the attacker can not infer ID_{t_R} from the eavesdropping ID_{t_L}. Third, there are at least two values that the attacker doesn't know in the calculations of $M1, M2, M3$, so that the attacker can not exhaust private information such as the shared private key and so on. Based on the above description, $WKGA-BO$ can resist the passive attack.

5.2.2 Active Attack

The attacker can inject the error messages into the forward channel and the backward channel, and deceive the reader and the tag, which makes them generate the wrong shared private key. So In this section we will analyze the attack from two aspects – the attacker disguises as the reader, and the attacker disguises as the tag.

1) The attacker disguises as the reader:

The attacker can inject the error messages to the forward channel before/while/after the WKGA-BO is executed at the legal reader and the legal tag, which can be seen as the attacker disguises as the legal reader and executes $WKGA-BO$ algorithm with the tag. In the following we uses the single tag shared private key generation algorithm as an example, to analyze that $WKGA-BO$ can resist the active attacks.

The attacker chooses to attack before the $WKGA-BO$ is executed. The attacker disguises as the reader to send information to the tag. Even if the attacker can pass the certifications of the first two steps, the attacker can not correctly execute the third step and the fourth step. In the third step, the attacker can not correctly calculate $M1, M2, M3$ by the unknown ID_{t_R}. In the fourth step, the tag can confirm the authenticity of $M1, M2$ and $M3$ to determine the authenticity of the reader quickly. The specific determination process is as follows.

Next the tag uses the received $M1$, its own ID_{t_R} to calculate $r'_L = M1 \oplus ID_{t_R}$, uses the received $M2$, its own ID_{t_R} to calculate $r'_R = M2 \oplus ID_{t_R}$, uses r'_L, r'_R to calculate $M3' = Sub(r'_L||ID_{t_R}, ID_{t_R}||r'_R)$, and then compares $M3'$ with $M3$. The attacker does not know the value of ID_{t_R}, so he can only choose a random number instead of ID_{t_R} to calculate and obtain the incorrect value, which results in the fourth step in which the ID_{t_R} used by the tag is not consistent with the ID_{t_R} used by the attacker. So the tag can determine that the reader is forged by the attacker, and $WKGA-BO$ terminates.

The attacker chooses to attack while the $WKGA-BO$ is executed. At this time the legal reader and tag are in the process of generating the shared private key. Because there is no initial shared private key, the tag can not distinguish the source of the message in the forward channel (that is, the tag can not determine that the message sent by the forward channel at this time is derived from the legal reader or from the attacker). Even if the attacker injects the wrong data, the tag can also distinguish the authenticity of the reader in the fourth step, and the specific analysis process is as described above.

The attacker chooses to attack after the $WKGA-BO$ is executed. At this time the tag and the legal

reader has generated the shared private key through executing $WKGA-BO$. Then the tag uses the generated shared private key to bidirectionally authenticate the reader (such as using the Hash-Lock protocol), and the attacker fails because the attacker can not obtain the shared private key.

In summary, when the attacker disguises as the legal reader to inject the message in the forward channel, the tag can always distinguish the authenticity of the reader, so $WKGA-BO$ can resist the impersonation attack that is caused by the attacker disguising as the reader.

2) The attacker disguises as the tag:
In this attack mode, the attacker sends information to a legal reader by actively injecting a message to the backward channel, in order to destroy the generation process of the shared private key or to obtain the shared private key between the reader and the tag, which is equivalent to the attacker deceiving the tag and the legal reader between the implementation of $WKGA-BO$ to generate the shared private key. Because before $WKGA-BO$ is executed, there is no shared private key between the tag and the reader, the legal reader can not distinguish the message sent by the legal tag or the tag that is disguised by the attacker. In the following we uses the single tag shared private key generation algorithm as an example, to analyze that $WKGA-BO$ can resist the active attacks.

The attacker can obtain the information $ID_{t_L}, M1, M2, M3$ through eavesdropping a complete communication process between the legal tag and the legal reader. The attacker tries to send the ID_{t_L} to the legal reader and wants to get the correct shared private key, but the attacker can not succeed. After the reader receives the ID_{t_L} sent by the attacker, although the ID_{t_L} can be found, the calculations of $M1, M2, M3$ do not use ID_{t_L}, but the ID_{t_R}, and there is no association between ID_{t_R} and ID_{t_L}. The random number are used in the calculations of $M1, M2, M3$, and the random numbers in each round of the random generation of the shared private key are different, so as to ensure that $M1, M2, M3$ in each round are various. Although the tag disguised by the attacker can obtain $M1, M2, M3$ sent by the legal reader, but because the attacker does not know ID_{t_R}, he can not calculate the random number generated by the legal reader, whick makes the attack unable to infer the shared private key from the received information. Before the subsequent tags communicate with the reader, there will be a bidirectional authentication process (such as Hash-Lock protocol). Through this process, the reader can argue that the two can not generate a unified shared private key, and then the reader and the tag re-execute $WKGA-BO$ to generate the shared private key.

Based on the above description, when the attacker disguises as the tag to inject information into the backward channel, although the $WKGA-BO$ can not generate a unified legal shared private key, the legal reader can still distinguish the authenticity of the tag. So $WKGA-BO$ can resist the impersonation attack that is caused by the attacker disguising as the tag.

6 GNY Logic Formal Proof

That the security of a complete protocol can be analyzed in words is far from enough. It can also be proved by the rigorous mathematical formulas. Based on this thought, in 1989 *Burrows* et al proposed a BAN formal logic analysis method, which was regarded as a milestone in the analysis of security protocols [3]. BAN logic is only concerned with the part of the protocol that is directly related to the authentication logic, and the rest is not a concern. It uses the rigorous mathematical rules to formalize the analysis and proof of the certification of the protocol. It also derives the target authentication step from the initialized hypothesis step of the protocol.

Because BAN form logic analysis has certain limitations, Gongli, etc. in 1990 put forward the GNY formal logic analysis method [11]. GNY formal logic analysis method is an expansion for BAN formal logic analysis method. GNY formal logic analysis method is more comprehensive than BAN logic analysis method, mainly in expanding the type and scope of analyzing the protocol. In this paper, the formal analysis and proof of $WKGA-BO$ protocol are carried out by using GNY formal logic analysis method.

$WKGA-BO$ gives three different scenarios to generate the shared private key. Because of the small differences among the three kind of the shared private key generation processes and the limited paper space, we choose to use the single tag's shared private key generation algorithm as an example to analyze and prove the protocol by GNY formal logic analysis method.

6.1 Formal Description of the Protocol

In order to make the $WKGA-BO$ protocol easy to be described by GNY formal logic, we use R to represent the reader, and T for the tag. The process of WKGA-BO protocol is as follows.

$$Msg1: R \rightarrow T: Hello;$$
$$Msg2: T \rightarrow R: ID_{t_L};$$
$$Msg3: R \rightarrow T: M1 = r_L \oplus ID_{t_R},$$
$$M3 = Sub(r_L||ID_{t_R}, ID_{t_R}||r_R), M2 = r_R \oplus ID_{t_R}.$$

Using GNY formal logic to standardize the above protocol, it can be described as follows.

$$Msg1: T < *Hello;$$

$$Msg2: T < *ID_{t_L};$$

$Msg3 : T < *M1 = r_L \oplus ID_{t_R}$,
$M3 = Sub(r_L||ID_{t_R}, ID_{t_R}||r_R), M2 = r_R \oplus ID_{t_R}$.

6.2 Initialization Hypothesis of the Protocol

The $WKGA-BO$ protocol assumes that R and T represent the entities, that is, R for the reader, and T for the tag. Hypothesis 1-3 represent what the tag and the reader have. Hypothesis 4 represents the trust of the reader and the tag on the freshness of the random number. Hypothesis 5-8 represent ID_{t_L}, ID_{t_R} are shared between the reader and the tag.

$$
\begin{aligned}
&Sub\ 1 : T \ni (ID_{t_R}, ID_{t_L})\\
&Sub\ 2 : R \ni (ID_{i_{t_R}}, ID_{i_{t_L}})\\
&Sub\ 3 : R \ni (r_L, r_R)\\
&Sub\ 4 : R| \equiv \#(r_L, r_R);\\
&Sub\ 5 : T| \equiv R \overset{ID_{t_R}}{\leftrightarrow} T;\\
&Sub\ 6 : R| \equiv T \overset{ID_{t_R}}{\leftrightarrow} R;\\
&Sub\ 7 : T| \equiv R \overset{ID_{t_L}}{\leftrightarrow} T;\\
&Sub\ 8 : R| \equiv T \overset{ID_{t_L}}{\leftrightarrow} R.
\end{aligned}
$$

6.3 Proof Target of the Protocol

There are 3 proof targets in $WKGA-BO$, mainly in the trust of the reader and the tag on the freshness of the interaction information.

The formulas of the targets are as follows.

$$
\begin{aligned}
&Goal\ 1 : T| \equiv R| \sim \#(M1 = r_L \oplus ID_{t_R});\\
&Goal\ 2 : T| \equiv R| \sim \#(M2 = r_R \oplus ID_{t_R});\\
&Goal\ 3 : T| \equiv R| \sim \#(M3 = Sub(r_L||ID_{t_R},\\
&\qquad\qquad ID_{t_R}||r_R)).
\end{aligned}
$$

6.4 Proof Process of the Protocol

The proof of the $WKGA-BO$ protocol is based on the initialization hypothesis. The proof process follows the logical reasoning rules, being-told rules, freshness rules and possession rules in Reference[30].The message interpretation rules follow the written form of the GNY logical reasoning rule in Reference[30], which are represented by T, P, F, I respectively.

This process $Goal\ 2 : T| \equiv R| \sim \#(M2 = r_R \oplus ID_{t_R})$, $Goal\ 3 : T| \equiv R| \sim \#(M3 = Sub(r_L||ID_{t_R}, ID_{t_R}||r_R))$ is similar to the process of the proof target $Goal\ 1 : T| \equiv R| \sim \#(M1 = r_L \oplus ID_{t_R})$. Therefore, this chapter takes the process of the proof target $Goal\ 1 : T| \equiv R| \sim \#(M1 = r_L \oplus ID_{t_R})$ as an example, which is described as follows.

1) Because $Rule\ P1 : \frac{P \leq X}{P \ni X}$ and $Msg3 : T* < \{M1 = r_L \oplus ID_{t_R}\}$, therefore $T \ni \{M1 = r_L \oplus ID_{t_R}\}$.

2) Because $Rule\ F1 : \frac{P|\equiv(x)}{P|\equiv(x,y),p|\equiv\#F(x)}$ and $Sup4 : R| \equiv \#(r_L, r_R)$, therefore $T = \#\{M1 = r_L \oplus ID_{t_R}\}$.

3) Because $Rule\ P2 : \frac{P \ni X, P \ni Y}{P \ni (X,Y), P \ni F(X,Y)}$, $Sup1 : T \ni (ID_{t_R}, ID_{t_L})$ and $Sup3 : R \ni (r_L, r_R)$, therefore $T \ni \{M1 = r_L \oplus ID_{t_R}\}$.

4) Because $Rule\ F10 : \frac{P|\equiv(X), P \ni X}{P|\equiv\#(H(X))}$ and the inferred rule $T = \#\{M1 = r_L \oplus ID_{t_R}\}$, $T \ni \{M1 = r_L \oplus ID_{t_R}\}$, therefore $T| \equiv \#\{M1 = r_L \oplus ID_{t_R}\}$.

5) Because $Rule\ I3 : \frac{P<H(X,<S>)>, P \ni (X,S), P|\equiv\#(X,S)}{P|\equiv Q|\sim(X,S), P|\equiv Q \sim H(X,<S>)}$, $Sup5 : T| \equiv R \overset{ID_{t_R}}{\leftrightarrow} T$, $Sup6 : R| \equiv T \overset{ID_{t_R}}{\leftrightarrow} R$ and $Msg3 : T* < \{M1 = r_L \oplus ID_{t_R}\}$, therefore $T| = R \sim \{M1 = r_L \oplus ID_{t_R}\}$.

6) Because The definition of freshness and the inferred $T = \#\{M1 = r_L \oplus ID_{t_R}\}, T| = R \sim \{M1 = r_L \oplus ID_{t_R}\}$, therefore $Goal\ 1 : T| \equiv R| \sim \#(M1 = r_L \oplus ID_{t_R})$.

The protocol is proved.

7 Performance Analysis

RFID system consists of three parts: tag, reader, and database. In this paper, the shared private key generation process is mainly completed in the tag and the reader, and the reader and the database are viewed as a whole, so the performance analysis is more concerned about the tag. In this section, we analyze the advantages and disadvantages of the WKGA-BO from the following three aspects: the storage space of the tag, the calculation of the tag, the communication cost of $WKGA-BO$.

Table 6: The Single Tag's Shared Private Key Generation Algorithm

Storage space	Calculation	Communication cost
$3l$	$2XOR + 2AND$ $+2OR + 2Sub()$	$6l$

The data in Table 7 is analyzed for the single tag's shared private key generation algorithm. The tag stores three kinds of data structures: ID_{t_R}, ID_{t_L}, Key. We set the length of each data structure is l bit, so the storage space of the tag is $3l$. XOR means bitwise XOR operation; AND means bitwise AND operation; OR means bitwise concatenation operation; $Sub()$ means bitwise substitution operation. The above four operations are bitwise operations, which can share part of the circuit to a certain extent, and they belong to super-lightweight operation, which can achieve the goal of reducing the cost of the tag. A complete communication process

in the single tag's shared private key generation algorithm requires the transmission of the following five data: $Hello, ID_{t_L}, M1, M2, M3$. According to the above analysis, the lengths of $Hello, ID_{t_L}, M1, M2$ are l bits, and the length of $M3$ is $2l$ bits, so the communication cost is $6l$.

Table 7: A Batch of Shared Private Keys Generation Algorithm

Storage space	Calculation	Communication cost
$3l$	$3XOR + 3AND$ $+2OR + 2Sac()$	$7l$

The data in Table 8 is analyzed for a batch of shared private keys generation algorithm. The tag stores three kinds of data structures: $ID_{i_{t_R}}$, $ID_{i_{t_L}}$, Key_i. We set the length of each data structure is l bit, so the storage space of the tag is $3l$. XOR means bitwise XOR operation; AND means bitwise AND operation; OR means bitwise concatenation operation; $Sac()$ means self-assembling cross-bit operation. The above four operations are bitwise operations, which can share part of the circuit to a certain extent, and they belong to super-lightweight operation, which can achieve the goal of reducing the cost of the tag. A complete communication process in a batch of shared private keys generation algorithm requires the transmission of the following five data: $Hello$, $ID_{i_{t_R}}$, $M4, M5, M6$. According to the above analysis, the lengths of $Hello$, $ID_{i_{t_R}}$, $M4, M5$ are l bits, and the length of $M6$ is $2l$ bits. $ID_{i_{t_R}}$ is transmitted twice, so the communication cost is $7l$.

Table 8: Group-based Shared Private Keys Generation

Storage space	Calculation	Communication cost
$3l$	$4XOR + 1AND$ $+1Sac()$	$7l$

The data in Table 9 is analyzed for the group-based shared private keys generation algorithm. The tag stores three kinds of data structures: $ID_{i_{t_R}}$, $ID_{i_{t_L}}$, Key. We set the length of each data structure is l bit, so the storage space of the tag is $3l$. XOR means bitwise XOR operation; AND means bitwise AND operation; $Sac()$ means self-assembling cross-bit operation. The above three operations are bitwise operations, which can share part of the circuit to a certain extent, and they belong to super-lightweight operation, which can achieve the goal of reducing the cost of the tag. A complete communication process in the group-based shared private keys generation algorithm requires the transmission of the following six data: $Hello$, $ID_{i_{t_L}}$, $M7, M8, M9, M10$. According to the above analysis, the lengths of $Hello$, $ID_{i_{t_L}}$, $M7, M8, M9, M10$ are l bits, and the length of $M6$ is $2l$ bits. $ID_{i_{t_L}}$ is transmitted twice, so the communication cost is $7l$.

8 Conclusions

This paper proposes a wireless generation algorithm $WKGA-BO$ for private key in RFID system based on bitwise operation. This algorithm mainly solves the problems that the shared private key between the tag and the reader must be pre-set, and it can not be customized by the user. $WKGA-BO$ can not only be used in the single tag's shared private key generation, but also for group tag's shared private key generation, which makes $WKGA-BO$ more widely used. The security analysis shows that $WKGA-BO$ can resist passive attacks and active attacks. GNY formal logic is used for the rigorous mathematical derivation of $WKGA-BO$. The performance analysis shows that $WKGA-BO$ is able to generate the shared private key on the tag and the reader in RFID systems. In summary, $WKGA-BO$ is suitable for use in existing RFID systems. The next step for the paper is to implement an RFID system that uses $WKGA-BO$, and to research the total number of required gate circuits, a complete communication time and so on, with the combination of theory and practice.

Acknowledgments

This work was supported in part by the Natural Science Foundation of China under Grant 61472090, Grant 61472089 and Grant 61672169, in part by the Science and Technology Project of Guangdong Province under Grant 2015B090906015 and Grant 2017B090906003, in part by the Science and Technology Planning Project of Guangzhou under Grant 201707010492, Grant 201604016003, Grant 201604016067 and Grant 201604016041.

References

[1] T. Agrawal, P. K. Biswas, and A. D. Raoot, "Optimum frame size evaluation framework for efficient tag identification in passive RFID systems," in *2013 IEEE Wireless Power Transfer*, pp. 48–51, 2013.

[2] P. Bellot and M. Dang, "Secret key agreement over a non-authenticated channel," *IEEE Transactions on Information Theory*, vol. 49, no. 4, pp. 48–55, 2003.

[3] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," in *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, pp. 233–271, 1989.

[4] C. Castelluccia and P. Mutaf, "Shake them up! a movement based pairing protocol for cpu-constrained

devices," in *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services*, pp. 51–64, 2005.

[5] C. L. Chen, Y. L. Lai, and C. C. Chen, "RFID ownership transfer authorization systems conforming epc global class-1 generation-2 standards," *International Journal of Network Security*, vol. 13, no. 1, pp. 41–48, 2011.

[6] W. T. Chen, "A feasible and easy-to-implement anticollision algorithm for the epc global uhf class-1 generation-2 RFID protocol," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 485–491, 2014.

[7] Y. C. Chen, W. L. Wang, M. S. Hwang, "RFID authentication protocol for anti-counterfeiting and privacy protection", in *The 9th International Conference on Advanced Communication Technology*, pp. 255–259, 2007.

[8] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[9] R. Doss, W. Zhou, and S. Yu, "Secure RFID tag ownership transfer based on quadratic residues," *IEEE Trans on Information Forensics and Security*, vol. 8, no. 2, pp. 390–401, 2013.

[10] Z. Y. Du, G. A. Zhang, and H. L. Yuan, "Crossover based ultra-lightweight RFID authentication protocol," *Computer Science*, vol. 40, no. 14, pp. 35–42, 2013.

[11] L. Gong, R. Needham, and R. Yahalom, "Reasoning about belief in cryptographic protocols," *IEEE Computer Society Symposium in Security and Privacy*, pp. 234–248, 1990.

[12] B. Jian and D. Liu, "Wireless key generation algorithm for RFID system based on bit operation," *Computer Engineering and Applications*, vol. 53, no. 16, 2017.

[13] Y. Jin and Q. Wu, "RFID lightweight authentication protocol based on prf," *Journal of computer Research and Development*, vol. 51, no. 7, pp. 1506–1514, 2014.

[14] G. Kapoor and S. Piramuthu, "Single RFID tag ownership transfer protocols," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 2, pp. 164–173, 2012.

[15] C. Kuo, M. Luk, and R. Negi, "Message-in-a-bottle: User-friendly and secure key deployment for sensor nodes," in *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, pp. 223–246, 2007.

[16] Y. C. Lai, L. Y. Hsiao, and H. J. Chen, "A novel query tree protocol with bit tracking in RFID tag identification," *IEEE Transactions on Mobile Computing*, vol. 12, no. 10, pp. 2063–2075, 2013.

[17] H. Lalndaluce, A. Perallos, and I. J. G. Zuazola, "A fast RFID identification with low tag complexity," *IEEE Communications Letters*, vol. 17, no. 9, pp. 1704–1706, 2013.

[18] D. W. Liu, J. Ling, and X. Yang, "An improved RFID authentication protocol with backward privacy," *Computer Science*, vol. 43, no. 8, pp. 128–130, 2016.

[19] L. Lu, "Wireless key generation for RFID systems," *Chinese Journal of Computers*, vol. 38, no. 4, pp. 822–832, 2015.

[20] C. S. Ma, "Low cost RFID authentication protocol with forward privacy," *Chinese Journal of Computers*, vol. 34, no. 8, pp. 1388–1398, 2011.

[21] Y. Ma and D. Liu, "Improved mutual authentication with backward security for RFID protocols," *Computer Engineering and Applications*, 2017.

[22] H. Mala, M. Dakhilalian, and M. Shakiba, "Cryptanalysis of mcrypton-a lightweight block cipher for security of RFID tags and sensors," *International Journal of Communication Systems*, vol. 25, no. 4, pp. 415–426, 2012.

[23] D. Moriyama, S. Matsuo, and M. Ohkubo, "Relations among notions of privacy for RFID authentication protocols," *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 97, no. 1, pp. 225–235, 2014.

[24] J. Si, B. Y. Yang, and D. Liu, "Wireless key generation algorithm for RFID system," *Computer Engineering and Design*, vol. 38, no. 9, 2017.

[25] F. Stajano and R. Anderson, "The resurrecting duckling: Security issues for ubiquitous computing," *Computer*, vol. 35, no. 4, pp. 22–26, 2002.

[26] Y. Tian, G. Chen, and J. Li, "A new ultralightweight RFID authentication protocol with permutation," *IEEE Communications Letters*, vol. 16, no. 5, pp. 702–705, 2012.

[27] S. Wang, S. Liu, and D. Chen, "Scalable RFID mutual authentication protocol with backward privacy," *Journal of computer Research and Development*, vol. 5, no. 6, pp. 1276–1284, 2013.

[28] C. H. Wei, M. S. Hwang, A. Y. H. Chin, "A mutual authentication protocol for RFID", *IEEE IT Professional*, vol. 13, no. 2, pp. 20–24, Mar. 2011.

[29] M. H. Yang, "Secure multiple group ownership transfer protocol for mobile RFID," *Electronic Commerce Research and Applications*, vol. 11, no. 4, pp. 361–373, 2012.

[30] Z. Zhang, Y. Liu, and D. Liu, "Based on tag's id wireless key generation for RFID system algorithm," *Application Research of Computers*, vol. 34, no. 1, pp. 261–263, 2017.

Biography

Rui Xie received his B.S. in electrical engineering and automation from Dalian Maritime University in 2000, and the M.Sc. in Computer Science from Guangdong University of Technology in 2003. He is currently a Ph.D Candidates in Guangdong University of Technology. His research interests cover a variety of different topics including netwok security, machine learning, cloud

computing, data mining and their applications.

Jie Ling received his Ph.D degree in computation mathematics from Sun Yat-sen University (China) in June 1998. He is a professor in computer science in Guangdong University of Technology. His current research interest fields include computer applications and intelligent video processing technology.

Dao-wei Liu received a master's degree in School of Computers from Guangdong University of Technology (China) in June 2016. His current research interest fields include information security.