A PUF-based Group Key Transfer Protocol for Bluetooth

Sensen Li, Bin Yu, and Yicai Huang
(Corresponding author: Sensen Li)

Department of Computer Engineering, Zhengzhou Information Science and Technology Institute
Zhengzhou 450001, China
(Email: lss589@163.com)

(Received Dec. 5, 2017; Revised and Accepted Mar. 19, 2018; First Online Mar. 2, 2019)

Abstract

Group key is the basis for ensuring the security of bluetooth broadcast messages. Recently, aiming at establishing group key among resource-constrained devices, Hsu *et al.* and Piao *et al.* respectively proposed a group key transfer protocol, but our analysis demonstrates that the two protocols can neither satisfy the security requirements of bluetooth. Physical Unclonable Function (PUF) is a mapping relationship based on the physical characteristics of a given device, which has a broad application prospect in the field of information security. In this paper, we propose a PUF-based group key transfer protocol for bluetooth, which can establish the shared group key between bluetooth master device and slave devices on the condition of slave devices not storing any secret parameters. The proposed protocol not only resists the traditional attacks such as eavesdropping, tampering and replaying, but also protects the bluetooth devices from replication attack. Compared with related protocols, this protocol has a higher security and obviously decreases the computation, storage and communication overhead.

Keywords: Bluetooth; Group Key Transfer; Physical Unclonable Function; Replication attack; Traditional Attacks

1 Introduction

Bluetooth has become one of the main communication ways of wireless sensors for the features of low cost, low complexity and high reliability. Nowadays, bluetooth is being used in smart home, medical care, indoor positioning and many other areas [6, 20, 26]. With the popularization of its application, the security issues of bluetooth have attracted extensive attention and gradually become the key factor constraining its development in the high security requirements fields, such as finance and military [14].

In the application of sensor networks, the basic communication topology of bluetooth is the piconet consisting of a master and several slavers. Key agreement is a pivotal step for secure information interaction among bluetooth devices. The bluetooth specification [2] implements the establishment of a point-to-point link key between the master and slaver by defining the processes called pairing and binding. However, the specification does not provide a mechanism for establishing the group key. Adversaries can attack the bluetooth broadcast channel by eavesdropping, tampering, replaying, etc. Besides, they are able to capture the slaver, whose self-protection ability is poor, then extract secret parameters from the slaver's storage medium and replicate malicious devices. Consequently, for the purpose of ensuring the security of the data in the broadcast channel, a lightweight cryptographic protocol should be used to establish the group key shared by the master with multiple slavers. And the protocol should meet the link security requirements and protect the device from replication attack.

The traditional group key management protocols can be divided into two categories: Centralized protocols and distributed protocols. The centralized ones require a device with strong computing and storage capabilities, called KGC, to generate and distribute the group key, while the distributed protocols don't have an explicit KGC and the group key is obtained by all group members through negotiation. In the bluetooth piconet, the master usually has abundant resources, while the slavers are usually nodes with simple structure and limited resources. Therefore, the centralized protocols are more suitable for bluetooth and the master can act as KGC.

In recent years, many researchers have studied the group key management in wireless sensor networks. Harn *et al.* [9] pointed out that the traditional centralized protocols [7, 10, 18, 23] and distributed protocols [3, 5, 13, 15, 28] have the problems like high computational complexity and the prolonged time delay of setting up a group key, so they proposed a group key transfer protocol based on (t, n) secret sharing scheme. In Harn's protocol, KGC generates the group key and broadcasts related secret information to group members. When receiving the secret information, each authorized member needs to calculate a t-degree interpolation polynomial to recover the group key. But Nam *et al.* [21] proved that Harn's protocol was unsecured. To improve Harn's protocol, Liu *et al.* [17] proposed a new protocol, which achieved the security at the cost of higher computation overhead. Based on the secret sharing scheme and factoring assumption, Hsu *et al.* [11] proposed an efficient group key transfer protocol for WSNs, whose communication and computation overheads are less than those of Harn's protocol, but unfortunately, this protocol can't achieve the claimed security for the reason that inside members can obtain the secret key shared by another group member with KGC. To fill the gap, Hsu *et al.* [12] improved their former protocol [11] by using the hash function to ensure the confidentiality of the key shared by group member and KGC. However, in this protocol, the KGC needs to perform $(t + 1)$ times hash function, where t is the number of group members, so the computation overhead is too high to be suitable for the resource-constrained bluetooth devices. Piao *et al.* [24] employed a polynomial to implement group key transfer. The protocol is lightweight and simple, but it can be proved that this protocol can't guarantee the forward security and backward security of the group key. In addition, the protocols above are all establishing the group key on the basis of the secret key shared by each group member with KGC, so each device needs to store such secret information in its memory. However, [19, 25] pointed out this storage way can't resist the replication attack on the devices, especially for the resource-constrained devices. By capturing one authorized group member and extracting the point-to-point key from its device memory, attackers can easily recover all the group keys for communications the captured member has participated in.

PUF [22] is a special mapping relationship between the input challenges and the output responses. Similar with using the unique features of human body like fingerprint, iris and so on, the mapping is based on the intrinsic physical characteristics of the device, which can be expressed as hardware fingerprint. With the features of uniqueness, unclonability, unpredictability and lightweight [27], PUF can be applied in authentication, key generation and many other fields. Many researches [1, 4, 8, 16] have probed the application of PUF in resource-constrained devices and significantly improved the security of these devices. However, the existing researches mainly focus on key generation, device authentication and point-to-point key agreement. Up to now, PUF has not been used for establishing the group key.

In this paper, we propose a PUF-based bluetooth group key transfer protocol that obviously decreases the resource overhead and improves the security. In our protocol, PUF is adopted to hide the information related to group key and the resource-constrained slave devices don't need to store any secret parameters, which effectively prevents the replication attack as well as traditional link attacks. The rest of this paper is organized as follows: In the next section, we briefly review the related protocols and then analyze their security weaknesses respectively. In Section 3, we propose our PUF-based group key transfer protocol for bluetooth. In Section 4, we analyze the correctness and security of our protocol. Section 5 provides the performance evaluation of the proposed protocol. Concluding remarks are given in Section 6.

2 Related Protocols and Their Security Analysis

2.1 Analysis of Hsu's Protocol

Hsu *et al.* [11] proposed a group key transfer protocol with low resource overhead, but we found that the protocol couldn't resist the insider attack. This section firstly reviews this protocol briefly, then gives the specific attack method of the malicious inside members.

1) Protocol review: Hsu's protocol consists of three phases: KGC initialization, user registration, group key generation and distribution. We only briefly introduce the most important phase: Group key generation and distribution, the process is as follows.

   Step1. The initiator sends a list of target group members $\{1, \cdots, t\}$to KGC, as a group key transfer request.

   Step2. When receiving the request, KGC broadcasts the group list $\{1, \cdots, t\}$.

   Step3. Each participating member sends a random number $R_i (i = 1, \cdots, t)$ to KGC.

   Step4. KGC randomly selects a group key K_G and a random number R_0. KGC also computes $U_i = (K_G - K_i) \bmod m \ (i = 1, \cdots, t)$ and authentication code $Auth$, where $K_i = (v(x_i), \overrightarrow{r}) = R_0 + R_1 x_i + R_2 {x_i}^2 + \cdots + R_t {x_i}^t$ and x_i is the secret parameter shared by the member i and KGC. Then, KGC broadcasts $\{Auth, R_0, U_i\}$ $(i = 1, \cdots, t)$.

   Step5. Each participating member i, knowing the x_i, is able to compute $K_i = (v(x_i), \overrightarrow{r})$ and recover the group key $K_G = (U_i + K_i) \bmod m$. Then the member i uses the authentication code $Auth$ to check the correctness of K_G.

2) Feasible attack method: The malicious group member *eve* can obtain the secret parameter shared by the member *target* with KGC. The detailed attack process is as follows.

   As the initiator, *eve* firstly sends the group key transfer request$\{eve, target\}$ to KGC, then these three parties, KGC, *eve* and *target*, execute the group key transfer process above. During this process, *eve* can get the parameter U_{target} and have the ability to compute $K_{target} = K_{G1} - U_{target}$. Therefore, *eve* obtains the Equation (1).

$$K_{target} = R_0 + R_{eve} x_{target} + R_{target} x^2_{target}. \quad (1)$$

By repeating the above procedure, *eve* gets the following Equation (2).

$$K'_{target} = R'_0 + R'_{eve}x_{target} + R'_{target}x^2_{target}. \quad (2)$$

Using the public parameters $\{R_0, R'_0, R_{eve}, R'_{eve}, R_{target}, R'_{target}\}$, *eve* can recover the secret x_{target} by executing $(1) \times R'_{target} - (2) \times R_{target}$, as shown in Equation (3).

$$\begin{aligned} x_{target} &= [(K_{target} - R_0) \times R'_{target} - \\ &\quad (K'_{target} - R'_0) \times R_{target}]/ \\ &\quad (R_{eve} \times R'_{target} - R'_{eve} \times R_{target}). \end{aligned} \quad (3)$$

2.2 Analysis of Piao's Protocol

Piao *et al.* [24] used the polynomial to implement the secret distribution of group keys. The implementation process is simple, but our analysis shows that the protocol can't guarantee the forward security and backward security of the group key.

1) Protocol review: Each group member $i(i = 1, \cdots, t)$ firstly establishes the secret KEY_i shared with KGC by registering, and then performs the following steps.

Step 1. KGC randomly selects a group key K_G and generates the related polynomial P, as shown in Equation (4).

$$\begin{aligned} P &= (x - KEY_1)(x - KEY_2)... \\ &\quad (x - KEY_t) + K_G \end{aligned} \quad (4)$$

Then, KGC broadcasts the polynomial P to group members $\{1, \cdots, t\}$.

Step 2. When receiving the polynomial, group member$i(i = 1, \cdots, t)$recovers the group key K_G by using the method shown in Equation (5).

$$\begin{aligned} K_G &= (x - KEY_1)(x - KEY_2) \cdots \\ &\quad (x - KEY_t) + K_G, \text{ where } x = KEY_i. \end{aligned} \quad (5)$$

2) Forward security analysis: Before joining the group, the node w can obtain the polynomialP_1, as shown in Equation (6), by monitoring the public channel. The constant term of this polynomial is $c_1 = (-1)^t \times KEY_1 \times ... \times KEY_t + K_{G1}$.

$$\begin{aligned} P_1 &= (x - KEY_1)(x - KEY_2)...(x - KEY_t) \\ &\quad +K_{G1}, w \notin \{1, \cdots, t\}. \end{aligned} \quad (6)$$

After the node w becomes a group member, the polynomial P_2, as shown in Equation (7), can be obtained during the group key distribution.

$$\begin{aligned} P_2 &= (x - KEY_1)(x - KEY_2)...(x - KEY_t) \\ &\quad \times(x - KEY_w) + K_{G2}. \end{aligned} \quad (7)$$

The constant term of P_2 is $c_2 = (-1)^{t+1} \times KEY_1 \times ... \times KEY_t \times KEY_w + K_{G2}$. Using the secret KEY_w, group member w is able to recover the group key K_{G2}. And Equation (8) shows the method to get the forward group keyK_{G1}, which shouldn't have been known by node w.

$$KG_1 = c_1 + \frac{c_2 - KG_2}{KEY_w} \quad (8)$$

3) Backward security analysis: In the process of group key distribution, group member i obtains the polynomial P_3, as shown in Equation (9).

$$\begin{aligned} P_3 &= (x - KEY_1)(x - KEY_2)...(x - KEY_t) \\ &\quad +K_{G3}, i \in \{1, \cdots, t\}. \end{aligned} \quad (9)$$

The constant term of P_3 is $c_3 = (-1)^t \times KEY_1 \times ... \times KEY_t + K_{G3}$. And group member i uses KEY_i to recover the group key K_{G3}.

After leaving the group, node i can get the polynomial P_4, as shown in Equation (10), by monitoring the channel. And its constant term is $c_4 = (-1)^{t-1} \times KEY_1 \times ... \times KEY_{i-1} \times KEY_{i+1} \times ... \times KEY_t + K_{G4}$.

$$\begin{aligned} P_4 &= (x - KEY_1)...(x - KEY_{i-1}) \\ &\quad \times(x - KEY_{i+1})...(x - KEY_t) \\ &\quad +K_{G4}. \end{aligned} \quad (10)$$

Utilizing the method shown in Equation (11), node i can recover the backward group key KG_4.

$$KG_4 = c_4 + \frac{c_3 - KG_3}{KEY_i} \quad (11)$$

From the above analysis, we can see that Piao's protocol can't guarantee the forward and backward security of the group key, so its security needs to be improved.

2.3 Analysis of Other Protocols

The existing group key transfer protocols all need to store sensitive parameters in device memory: the protocols based on symmetric cryptography need to save symmetric key information, while the ones based on public key cryptography need to save the device's private key.

With a simple structure and limited resources, the bluetooth slave device is usually difficult to achieve self-protection. If attackers capture a slave device, they may extract sensitive information, such as keys or algorithm parameters, using the method shown in Figure 1. Utilizing these information, attackers can not only figure out the group key to decrypt broadcast messages, but also replicate a similar node to forge the identity of the legal device and deliver false information, which poses a tremendous threat to the security of the bluetooth.

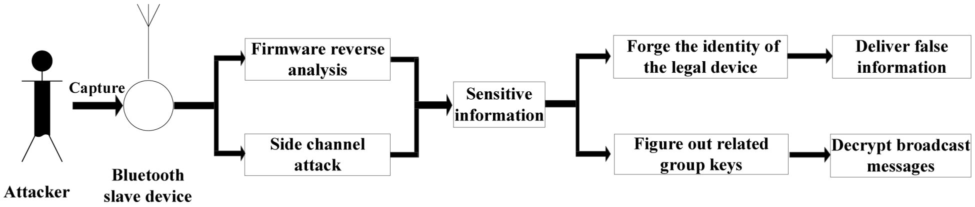
Attacker
Capture
Bluetooth slave device
Firmware reverse analysis
Side channel attack
Sensitive information
Forge the identity of the legal device
Figure out related group keys
Deliver false information
Decrypt broadcast messages

Figure 1: Group key establishment process

3 The Proposed Protocol

This section firstly gives the basic model of the bluetooth network, according to which we propose the attack assumptions. Then, the detailed process of the PUF-based group key transfer protocol is introduced.

3.1 Network Model and Attack Assumptions

The participants of the proposed protocol include a bluetooth master device (*Master*) and multiple bluetooth slave devices (*Slavers*). Playing the role of KGC, *Master*, with strong computing, storage and self-protection ability, is responsible for the generation and distribution of group keys; while *Slaver* is usually an information gathering node with a simple structure and limited resources, which is vulnerable to replication attack for its poor self-protection ability.

According to the characteristics of different devices, we put forward the following attack assumptions.

1) Adversaries can not only attack the network by traditional manners, such as eavesdropping, tampering and replaying, but also capture the bluetooth devices in the open environment.

2) Once the *Slaver* is captured, adversaries can obtain all the secret information in the device's storage medium by replication attack.

3) The *Master* has sufficient self-protection capability to resist replication attack.

3.2 Protocol Details

The proposed group key transfer protocol consists of two processes: initialization and group key establishment. The initialization process accomplishes the selection of basic parameters and the registration of *Slavers*. In group key establishment process, *Master* generates the group key and distributes it to *Slavers* based on the PUF.

1) Initialization: The *Master* randomly chooses two secure primes, p and q, and computes $n = pq$. Here, n is made publicly known. Then, the *Master* randomly selects $C_1, C_2 \in \mathbb{Z}_n^*$ as the challenges of PUF. All computations of the proposed protocol are performed in $\mathbb{Z}_n^*$.

   Each device is required to register to the *Master* when joining the network. And this process is performed in a secure manner, for example, it can be achieved with the help of the network administrator. For the device $Slaver_i$, *Master* uses the PUF of $Slaver_i$ to obtain the unique responses$R_{i,1}$ and $R_{i,2}$, where $R_{i,1} = \mathrm{PUF}_i(C_1)$ and $R_{i,2} = \mathrm{PUF}_i(C_2)$. Then, *Master* stores the tuple $(i, R_{i,1}, R_{i,2})$.

   After the initialization process is completed, $Slaver_i$ destroys its one-time PUF external interface for the purpose that adversaries outside the device chip have no access to challenge-response pairs of the PUF.

2) Group key establishment: The group key establishment process contains five steps and the detailed description is as follows.

   Step1. The initiator, *init*, sends the list of target group members $\{1, 2, \cdots, t\}$, $init \in [1,\ t]$ to *Master*, as a group key establishment request.

   Step2. For each group member $Slaver_i$, $i = 1, \cdots, t$, *Master* randomly selects a secret parameter $x_i \in \mathbb{Z}_n^*$. Then, *Master* broadcasts the message$\{C_1, C_2, i, R_{i,1} \oplus x_i\}$.

   Step3. When receiving the message, $Slaver_i$ uses its PUF to acquire the responses $R_{i,1} = \mathrm{PUF}_i(C_1)$ and $R_{i,2} = \mathrm{PUF}_i(C_2)$, then figures out the secret parameter x_i. After that, $Slaver_i$ generates a random number $y_i \in \mathbb{Z}_n^*$ and sends the message $\{R_{i,2} \oplus y_i\}$ to Master.

   Step4. Knowing the response$R_{i,2}$, *Master* is able to recover the secret y_i generated by each group member. *Master* randomly selects the group key $K_G \in \mathbb{Z}_n^*$ and constructs the t-degree equation $a_1 x + a_2 x^2 + \cdots + a_t x^t = K_G$, whose roots are $\{k_1, k_2, \cdots, k_t\}$ and $k_i = x_i \oplus rev(y_i)$. (Here, $rev(y_i)$ represents the reverse order of the binary sequence y_i and the method to get the equation coefficients $\{a_1, a_2, \cdots, a_t\}$ is introduced in Section 4.1.) Then, *Master* computes the group key authentication code $Auth = \mathrm{H}(a_1||a_2||\cdots||a_t||K_G)$ and broadcasts the message $\{a_1, a_2, \cdots, a_t, Auth\}$.

   Step5. Knowing the secret parameters x_i and y_i, $Slaver_i$ is able to figure out a equation root $k_i = x_i \oplus rev(y_i)$. And the group key

K_G can be recovered by computing $K_G = f(k_i)$, where $f(x) = a_1x + a_2x^2 + \cdots + a_tx^t$. Then, $Slaver_i$ checks the validity of K_G by computing $\mathrm{H}(a_1||a_2||\cdots||a_t||K_G)$ and comparing whether the hash value is equal to $Auth$. Figure 2 shows the process of group key establishment for a group whose members are $\{Slaver_A, Slaver_B, Slaver_C\}$.

After the communication is completed, each group member deletes the group key and other related parameters in the device. In other words, *Slavers* in the open environment don't store any secret parameters.

4 Protocol Analysis

In this section, we analyze the correctness and security of the proposed protocol respectively. Then, the security comparison between this protocol and other related protocols is listed.

4.1 Correctness Analysis

In our protocol, *Master* is required to figure out the proper coefficients $\{a_1, a_2, \cdots, a_t\}$ in order to make the roots of the equation $a_1x + a_2x^2 + \cdots + a_tx^t = K_G$ be $\{k_1, k_2, \cdots, k_t\}$, where $k_i = x_i \oplus rev(y_i)$. It's a key issue to prove that no matter what the values of $\{k_1, k_2, \cdots, k_t\}$ are, the coefficients are surely existed, which is vital to the correctness of the proposed protocol. The detailed proof process is as follows.

By substituting $\{k_1, k_2, \cdots, k_t\}$ into the equation $a_1x + a_2x^2 + \cdots + a_tx^t = K_G$, we can obtain the linear equations, whose unknowns are $\{a_1, a_2, \cdots, a_t\}$, as shown in Equation (12).

$$\begin{cases} k_1a_1 + k_1^2a_2 + \cdots + k_1^ta_t = K_G \\ k_2a_1 + k_2^2a_2 + \cdots + k_2^ta_t = K_G \\ \cdots \\ k_ta_1 + k_t^2a_2 + \cdots + k_t^ta_t = K_G \end{cases} \tag{12}$$

The coefficient matrix of the linear equations is A, as shown in Equation (13).

$$A = \begin{bmatrix} k_1 & k_1^2 & \cdots & k_1^t \\ k_2 & k_2^2 & \cdots & k_2^t \\ \cdots & \cdots & \cdots & \cdots \\ k_t & k_t^2 & \cdots & k_t^t \end{bmatrix} \tag{13}$$

Through calculation, we can see that the determinant of matrix A is $|A| = (k_1k_2\cdots k_t)\prod_{1\leq j<i\leq n}(k_i - k_j)$. On account that x_i and y_i are randomly generated and $k_i = x_i \oplus rev(y_i)$, it is reasonable to think that when $i \neq j$, $k_i \neq k_j$ and then $|A| \neq 0$. According to the *Cramer Rule*, when $|A| \neq 0$, the linear Equation (12) has the unique solution $\{a_1, a_2, \cdots, a_t\}$.

For the reason that $\{k_1, k_2, \cdots, k_t\}$ are the roots of the equation $a_1x + a_2x^2 + \cdots + a_tx^t = K_G$, we can conclude this equation is equivalent to Equation (14). And the parameters $\{a_1, a_2, \cdots, a_t\}$ can be obtained by expanding Equation (14).

$$a_t(x - k_1)(x - k_2)\cdots(x - k_t) = 0. \tag{14}$$

When receiving the $\{a_1, a_2, \cdots, a_t\}$, each group member $Slaverr_i$ is able to recover the group key $K_G = f(k_i)$ by substituting $k_i = x_i \oplus rev(y_i)$ into the function $f(x) = a_1x + a_2x^2 + \cdots + a_tx^t$.

4.2 Security Analysis

The security of the proposed protocol depends on the confidentiality of PUF's challenge-response pairs (CRPs), which can be guaranteed by the unclonability and unpredictability of PUF. That is to say, the following security conditions are true.

1) The unclonability of PUF: For a given PUF, it's infeasible to construct a PUF' by physical manners enabling $PUF'(c) = PUF(c)$ for any challenge signal c.

2) The unpredictability of PUF: Given a CRPs set $L = \{(c_i, PUF(c_i))|i = 1, 2, \cdots l\}$, the probability to predict the response $PUF(c_x)$ is negligible, where c_x is a random challenge signal and $(c_x, PUF(c_x)) \notin L$.

Based on the above security conditions, we prove the security of the proposed protocol by the following two theorems.

Theorem 1. *The proposed protocol can guarantee the freshness, confidentiality, authentication, forward security and backward security of the group key.*

1) Key freshness. When receiving the group key establishment request from the initiator, *Master* randomly selects the group key K_G and secretly distributes K_G to group members by constructing the equation $a_1x + a_2x^2 + \cdots + a_tx^t = K_G$. The roots of the equation are $\{k_1, k_2, \cdots, k_t\}$ and $k_i = x_i \oplus rev(y_i)$, where x_i and y_i are the random parameters selected by *Master* and $Slaver_i$ respectively. Therefore, using random numbers as the fresh factor, the proposed protocol can ensure the freshness of the group key.

2) Key confidentiality. In the process of group key distribution, the parameters transmitted in the public channel include $\{C_1, C_2, R_{i,1} \oplus x_i, R_{i,2} \oplus y_i, a_1, a_2, \cdots, a_t, Auth\}$. Because of the unclonability and unpredictability of PUF, the attacker can't figure out the corresponding response signals R_1 and R_2 as well as the equation's root $k_i = x_i \oplus rev(y_i)$. In addition, due to the one-way nature of the hash function, the attacker can't obtain any secret information from the authentication code $Auth$. For the function $f(x) = a_1x + a_2x^2 + \cdots + a_tx^t$, it is impracticable to calculate $K_G = f(k_i)$ only by using the coefficients $\{a_1, a_2, \cdots, a_t\}$. In conclusion, the group key in the protocol is confidential.

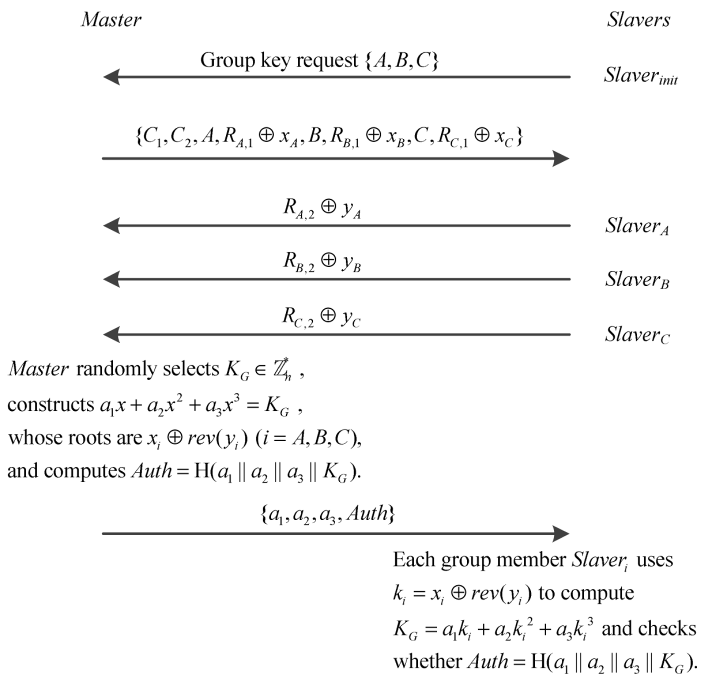
Master
Slavers
Group key request $\{A, B, C\}$
$Slaver_{init}$
$\{C_1, C_2, A, R_{A,1} \oplus x_A, B, R_{B,1} \oplus x_B, C, R_{C,1} \oplus x_C\}$
$R_{A,2} \oplus y_A$
$Slaver_A$
$R_{B,2} \oplus y_B$
$Slaver_B$
$R_{C,2} \oplus y_C$
$Slaver_C$
Master randomly selects $K_G \in \mathbb{Z}_n$,
constructs $a_1x + a_2x^2 + a_3x^3 = K_G$,
whose roots are $x_i \oplus rev(y_i)$ $(i = A, B, C)$,
and computes $Auth = \mathrm{H}(a_1 \| a_2 \| a_3 \| K_G)$.
$\{a_1, a_2, a_3, Auth\}$
Each group member $Slaver_i$ uses
$k_i = x_i \oplus rev(y_i)$ to compute
$K_G = a_1k_i + a_2k_i^2 + a_3k_i^3$ and checks
whether $Auth = \mathrm{H}(a_1 \| a_2 \| a_3 \| K_G)$.

Figure 2: Group key establishment process

3) Key authentication. Group members use the authentication code $Auth$, which is the hash value of equation coefficients $\{a_1, a_2, \cdots, a_t\}$ and the group key K_G, to judge the validity of the group key. The attackers outside the group can't correctly forge the authentication code for the freshness and confidentiality of K_G. Any group member, knowing the group key, also can't forge the authentication code without being detected, for the reason that the coefficients $\{a_1, a_2, \cdots, a_t\}$ are related to the secrets shared between each group member and *Master*. Consequently, as long as the calculated hash value $\mathrm{H}(a_1||a_2||\cdots||a_t||K_G)$ is equal to the received $Auth$, group members can believe K_G is generated by *Master*.

4) Forward security and backward security. Since the group key is randomly generated by *Master* and the parameters of the equation are fresh, group keys generated at different times have nothing to do with each other. Therefore, the unauthorized device can't infer other group keys through a group key. In other words, the protocol can guarantee the forward security and backward security of the group key.

Theorem 2. *While resisting the attackers outside the group, the proposed protocol provides protection against the insider attack, which is initiated by the malicious inside group member.*

Proof. According to *Theorems 1*, the attackers outside the group can't obtain the group key by eavesdropping the bluetooth channel, so the protocol can resist the attacks from malicious outside devices. Different from the attackers outside the group, the malicious inside attackers are authorized to know the group keys and their attack attempt is to recover the CRPs of another member' s PUF. □

In order to facilitate the representation, we assume that the malicious device in the group is $Slaver_{eve}$ and its attack target is $Slaver_{target}$. $Slaver_{eve}$ can obtain the polynomial $a_1k_{target} + a_2k_{target}^2 = KG \pmod{n}$, where $k_{target} = x_{target} \oplus rev(y_{target})$, by sending the group key establishment request $\{eve, target\}$ to *Master*. And sending the same request again, $Slaver_{eve}$ will get a similar but different polynomial $a_1'k'_{target} + a_2'(k'_{target})^2 = KG' \pmod{n}$. Since all the parameters in the polynomial are fresh, there is no correlation between the polynomials. In other words, $Slaver_{eve}$ can only get the secret parameter k_{target} by solving the polynomial $a_1k_{target} + a_2k_{target}^2 = KG \pmod{n}$. But Harn *et al.* [9] pointed out that this is an intractable problem due to the *Factoring Assumption*. For our protocol, even if $Slaver_{eve}$ has the ability to solve the factorization problem and figure out k_{target}, he still obtains nothing about the CRPs of $Slaver_{target}$, $(C_1, R_{target,1})$ and $(C_2, R_{target,2})$, since $k_{target} = x_{target} \oplus rev(y_{target})$. Therefore, the proposed protocol provides protection against the insider attack while resisting the attackers outside the group.

For the reason that the bluetooth slave devices don't need to store any secret parameters and the PUF has unclonability and unpredictability, the proposed protocol can not only resist the traditional attacks such as eavesdropping, tampering and replaying, but also effectively prevent the possible replication attack on the slave devices. The security comparison between the proposed protocol and other related protocols is shown in Table 1.

5 Performance Evaluation

This section firstly analyzes the performance of the proposed group key transfer protocol from three aspects, computation, communication and storage overhead. Then, we compare our protocol with Liu's protocol [17], which is more secure than other existing protocols. The proposed protocol consists of two processes: initialization and group key establishment. This section mainly analyzes the resource overhead of group key establishment process, since the former process only needs to perform one time while the latter process performs as long as *Master* has received the "group key establishment request".

In the bluetooth network, the master device, *Master*, usually has strong computation, communication and storage capabilities, while the slave devices, *Slavers*, only possess limited resources. Therefore, it is more important to consider reducing the resource overhead of the slave devices when designing the protocol.

For the convenience of description, we assume that the slave devices in the network are $\{1, 2, \cdots, m\}$, the group members are $\{1, 2, \cdots, t\}(t \leq m)$, the length of each parameter in $\mathbb{Z}_n^*$ is $|n|$ and the length of the hash is $|H|$.

5.1 Computation Overhead

We use T_M, T_I and T_H, respectively, to represent the time required to perform modular multiplication, modular inversion and hash. Compared to T_M, T_I and T_H, the time required for other operations, such as modular addition and subtraction, can be ignored [11].

In the proposed protocol, *Master* needs to perform $\frac{1}{2} \times t \times (t+1)$ times modular multiplication and one hash operation, so its computation overhead is $\frac{1}{2} \times t \times (t+1) \times T_M + t \times T_H$, while the computation overhead of $Slave_i(i = 1, 2, \cdots, t)$ is $(2t-1) \times T_M + T_H$ for performing $(2t-1)$ times modular multiplication and one hash operation. In the same way, we can get that, in Liu's protocol, the computation overhead of *Master* is $t \times (t+1) \times t \times (T_M + T_I) + (t+1) \times T_H$ and the computation overhead of $Slave_i$ is $(t+1) \times t \times (T_M + T_I) + 2 \times T_H$.

Table 2 shows the comparison of computation overhead between the proposed protocol and Liu's protocol. It can be seen that the proposed protocol obviously reduces the computation overhead of devices, include *Master* and *Slavers*.

5.2 Communication Overhead

The communication overhead is measured using the length of the messages sent by the device in group key establishment process. In the proposed protocol, the communication overheads of *Master* and $Slave_i(i = 1, 2, ..., t)$ are approximately $(2t+2)|n| + |H|$ and $|n|$, respectively. In Liu's protocol, *Master*'s communication overhead is about $2t|n| + |H|$ and $Slaver_i$'s is about $|n|$.

The communication overhead of each protocol is shown in Table 3. The overall communication overhead of the proposed protocol is almost equal to Liu's protocol. And in the two protocols, the overheads of resource-constrained devices, *Slavers*, are identical.

5.3 Storage Overhead

In the proposed protocol, *Master* needs to store the challenge signals of PUF, $\{C_1, C_2\}$, and the response signals of $Slave_j(j = 1, 2, \cdots, m)$, $\{R_{j,1}, R_{j,2}\}$, while *Slavers* don't need to store any parameter. In other words, the storage overhead of *Master* is about $2|n| + 2m|n|$ and $Slaver_j$'s overhead is 0. In Liu's protocol, *Master*'s storage overhead is $2m|n|$ and $Slaver_j$'s is $2|n|$.

Compared with Liu's protocol, the proposed protocol significantly reduces the total storage overhead of the network and the overheads of resource-constrained devices are lower.

6 Conclusions

In this paper, we analyze the security of the existing group key transfer protocols when applied to the bluetooth network and put forward several feasible attack methods respectively. As a remedy, we have proposed a PUF-based group key transfer protocol for bluetooth. The security of the proposed protocol is based on the unclonability and unpredictability of PUF. Compared with related protocols, this protocol significantly reduces the resource overhead of the device and its security is higher.

References

[1] M. Aman, K. C. Chua, and B. Sikdar, "Mutual authentication in iot systems using physical unclonable functions," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1327–1340, 2017.

[2] Bluetooth SIG, *Specification of the Bluetooth System: Core Package Version 4.0*, Technical Report Bluetooth Core v4.0, Dec. 2009.

[3] E. Bresson, O. Chevassut, and D. Pointcheval, "Provably secure authenticated group diffie-hellman key exchange," *Acm Transactions on Information and System Security Journal*, vol. 10, no. 3, pp. 255–264, 2007.

[4] U. Chatterjee, R. S. Chakraborty, and D. Mukhopadhyay, "A puf-based secure communication protocol for IOT," *Acm Transactions on Embedded Computing Systems*, vol. 16, no. 67, pp. 1–25, 2016.

[5] J. C. Cheng and C. S. Laih, "Conference key agreement protocol with non-interactive fault-tolerance over broadcast network," *International Journal of Information Security*, vol. 8, no. 1, pp. 37–48, 2009.

[6] J. J. V. Diaz, A. B. R. Gonzalez, and M. R. Wilby, "Bluetooth traffic monitoring systems for travel time estimation on freeways," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 1, pp. 123–132, 2016.

[7] M. Eltoweissy, M. H. Heydari, L. Morales, and I. H. Sudborough, "Combinatorial optimization of group key management," *Journal of Network and Systems Management*, vol. 12, no. 1, pp. 33–50, 2004.

Table 1: Security comparison

Scheme	Resisting Outsider Attack	Resisting Insider Attack	Resisting Replication Attack	Forward Security	Backward Security
Harn's protocol [9]	√	×	×	√	√
Liu's protocol [17]	√	√	×	√	√
Hsu's protocol [11]	√	×	×	√	√
Piao's protocol [24]	√	√	×	×	×
Our protocol	√	√	√	√	√

Table 2: Comparison of computation overhead

Protocol	*Master*	*Slaver*
Our protocol	$\frac{1}{2} \times t \times (t+1) \times T_M + T_H$	$(2t-1) \times T_M + T_H$
Liu's protocol	$t\times(t+1)\times t\times(T_M+T_I)+(t+1)\times T_H$	$(t+1)\times t\times(T_M+T_I)+2\times T_H$

Table 3: Comparison of communication overhead

Protocol	*Master*	*Slaver*
Our protocol	$(2t+2)\|n\|+\|H\|$	$\|n\|$
Liu's protocol	$2t\|n\|+\|H\|$	$\|n\|$

Table 4: Comparison of communication overhead

Protocol	*Master*	*Slaver*	Totally
Our protocol	$2\|n\|+2m\|n\|$	0	$2\|n\|+2m\|n\|$
Liu's protocol	$2m\|n\|$	$2\|n\|$	$4m\|n\|$

[8] Y. B. Guo, Z. N. Zhang, and K. W. Yang, "Authenticated key exchange protocol based on physical unclonable function system in wireless sensor networks," *International Journal of Advancements in Computing Technology*, vol. 4, no. 23, pp. 300–308, 2012.

[9] L. Harn and C. Lin, "Authenticated group key transfer protocol based on secret sharing," *IEEE Transactions on Computers*, vol. 59, no. 6, pp. 842–846, 2010.

[10] H. Harney, C. Muckenhirn, and T. Rivers, *Group Key Management Protocol (GKMP) Architecture*, RFC 2094, July 1997.

[11] C. F. Hsu, L. Harn, T. He, and M. Zhang, "Efficient group key transfer protocol for WSNs," *IEEE Sensors Journal*, vol. 16, no. 11, pp. 4515–4520, 2016.

[12] C. F. Hsu, L. Harn, Y. Mu, M. Zhang, and X. Zhu, "Computation-efficient key establishment in wireless group communications," *Wireless Networks*, vol. 23, no. 1, pp. 1–9, 2016.

[13] K. H. Huang, Y. F. Chung, H. H. Lee, F. Lai, and T. S. Chen, "A conference key agreement protocol with fault-tolerant capability," *Computer Standards and Interfaces*, vol. 31, no. 2, pp. 401–405, 2009.

[14] M. S. Hwang, C. C. Lee, J. Z. Lee, and C. C. Yang, "A secure protocol for bluetooth piconets using elliptic curve cryptography", *Telecommunication Systems*, vol. 29, no. 3, pp. 165-180, 2005.

[15] J. Katz and M. Yung, "Scalable protocols for authenticated group key exchange," *Journal of Cryptology*, vol. 20, no. 1, pp. 85–113, 2003.

[16] Y. S. Lee, H. J. Lee, and E. Alasaarela, "Mutual authentication in wireless body sensor networks (WBSN) based on physical unclonable function (PUF)," in *International Wireless Communications and Mobile Computing Conference (IWCMC'13)*, pp. 1314–1318, July 2013.

[17] Y. Liu, C. Cheng, J. Cao, and T. Jiang, "An improved authenticated group key transfer protocol based on secret sharing," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2335–2336, 2013.

[18] J. W. Lo, S. C. Lin, and M. S. Hwang, "A parallel password-authenticated key exchange protocol for wireless environments," *Information Technology and Control*, vol. 39, no. 2, pp. 146–151, 2010.

[19] C. Marchand, L. Bossuet, U. Mureddu, N. Bochard, and A. Cherkaoui, "Implementation and characterization of a physical unclonable function for IOT: A case study with the TERO-PUF," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 1, pp. 97–109, 2018.

[20] G. Mokhtari, Q. Zhang, G. Nourbakhsh, S. Ball, and M. Karunanithi, "Bluesound: A new resident identification sensor Xusing ultrasound array and ble technology for smart home platform," *IEEE Sensors Journal*, vol. 17, no. 5, pp. 1503–1512, 2017.

[21] J. Nam, M. Kim, J. Paik, W. Jeon, and B. Lee, "Cryptanalysis of a group key transfer protocol based on secret sharing," in *Future Generation Information Technology - Third International Conference*, pp. 309–315, Dec. 2011.

[22] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, 2002.

[23] A. Perrig, D. Song, and J. D. Tygar, "Elk, a new protocol for efficient large-group key distribution,"

in *Proceedings of IEEE Symposium on Security and Privacy*, pp. 247–262, May 2001.

[24] Y. Piao, J. U. Kim, U. Tariq, and M. Hong, "Polynomial-based key management for secure intra-group and inter-group communication," *Computers and Mathematics with Applications*, vol. 65, no. 9, pp. 1300–1309, 2013.

[25] S. Skorobogatov, "Flash memory 'bumping' attacks," in *International Conference on Cryptographic Hardware and Embedded Systems (CHES'10)*, pp. 158–172, Aug. 2010.

[26] M. Singh and N. Jain, "Performance and evaluation of smartphone based wireless blood pressure monitoring system using bluetooth," *IEEE Sensors Journal*, vol. 16, no. 23, pp. 8322–8328, 2016.

[27] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Design Automation Conference (DAC'07)*, pp. 9–14, June 2007.

[28] C. C. Yang, T. Y. Chang, and M. S. Hwang, "A new anonymous conference key distribution system based on the elliptic curve discrete logarithm problem," *Computer Standards and Interfaces*, vol. 25, no. 2, pp. 141–145, 2003.

Biography

Sensen Li is currently an Assistant in Zhengzhou Information Science and Technology Institute, China. His research interests include wireless sensor networks, information security and bluetooth.

Bin Yu is currently a Professor with Zhengzhou Information Science and Technology Institute, China. His research interests include information security, wireless sensor networks, embedded systems and visual cryptography.

Yicai Huang is currently a Lecturer with Zhengzhou Information Science and Technology Institute, China. His research interests include wireless sensor networks, information security and bluetooth.