Bluetooth Low Energy Security Vulnerability and Improvement Method

Giwon Kwon, Jeehyeong Kim, Jaewon Noh, Sunghyun Cho
Department of Computer Science and Engineering, Hanyang University, Ansan, Republic of Korea
{giwonkwon, whiteje, wodnjs1451, chopro}@hanyang.ac.kr

Abstract

In this paper, we introduce a security vulnerability of Bluetooth Low Energy(BLE) and propose an improved security method to solve it. BLE is used to transmit and receive data between nodes such as sensors, wearable devices in the Internet of Things (IoT) environment. However, there is a possibility that an encryption key for communicating between nodes is cracked because of the security vulnerability of BLE. The security vulnerability of BLE is that the length of the Temporary Key (TK) to generate the encryption key is too short. It is a fatal problem in terms of a data security. We describe how an attacker can crack the encryption key through the vulnerability. To solve this problem, we propose an improved security method that increases the length of the TK. In the conventional BLE, the encryption key can be cracked within 20 seconds. But, it may take 600,000,000,000 years when the proposed method is used. Therefore, the proposed method can ensure that it is infeasible to crack the encryption key.

Keywords: Bluetooth Low Energy, Internet of Things, Vulnerability

1. Introduction

An Interest for IoT [1-2] has been increasing more and more recently. In the IoT environment, various nodes are used to collect a data and each node uses a wireless communication technology such as Bluetooth, Wi-Fi, and, ZigBee for transmitting and receiving the data. Among these wireless communication technologies, Bluetooth Low Energy (BLE) is commonly used for the IoT environment because BLE is a suitable technology for transmitting and receiving a data using low energy wirelessly [3-8]. However, BLE has a security vulnerability that is abused to hijack a data.

BLE has the pairing process to generate an encryption key for a connection. There is a key called the TK that are used to generate the encryption key in the pairing process. In a certain situation an attacker cracks the encryption key because the TK has a short length.

To solve the above vulnerability, we propose an improved security method. In the proposed method, the length of the TK can increase by repeating a step related to generate the TK. The proposed method is computational secure because it may take 600,000,000,000 years to crack the encryption key when the proposed method is used. So, it ensures a secure data communication between nodes. Also, the proposed method can be implemented through an amendment of BLE standard for supporting a backward compatibility.

2. Bluetooth Low Energy Security Procedure and Vulnerability.

According to the Bluetooth core specification version 4.2 [9], Bluetooth provides a Security Manager (SM) for secure data transmission. SM defines the pairing process, a key distribution method, and a security tool for Bluetooth security.

2.1 Important keys and functions

In BLE, the following keys are used.

- Temporary Key (TK)

A 128-bit temporary key is used to generate the STK in the pairing process.

- Short Term Key (STK)

A 128-bit temporary key is used to encrypt a connection after pairing.

- Long Term Key (LTK)

A 128-bit key is used to encrypt the connection after the transport specific key distribution step is end.

The following functions are also used.

- Function c1

This function is used to generate the confirmation value.

- Function s1

This function is used to generate the STK.

2.2 Pairing Process

Pairing is performed to generate an encryption key that encrypts a connection between Bluetooth devices. Pairing consists of four steps. Fig. 1 shows the pairing process in BLE.

978-1-5090-2743-9/16/$31.00 ©2016 IEEE

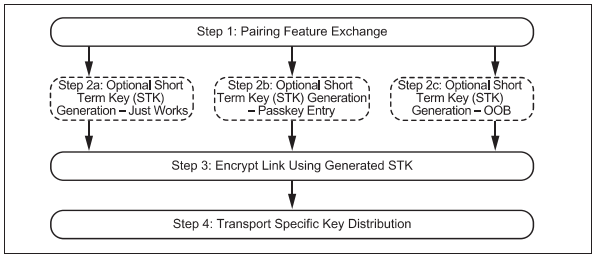

Fig. 1. Pairing process in BLE

In step 1 (pairing feature exchange), devices exchange the authentication requirements and IO capabilities in order to determine which method is used in STK generation of step 2. The followings are available methods in step 2:

- Just Works

The Just works is a method for devices that do not have a keyboard. A headset can be an example of this case.

- Passkey entry

The Passkey entry method uses six digits for TK. After a device generates six digits and represents on a display, a user inputs the displayed digits in another device using a keyboard on it.

- Out of Band(OOB)

The OOB method uses an interface other than Bluetooth for exchanging pairing information. This method is more secure than the passkey entry and just works methods. However, both devices must have an interface to be compatible with each other.

In step 3, devices encrypt a communication using the generated STK. Finally, the last step is performed to distribute transport specific keys for a reconnection in the future.

2.3 Security Procedure

This section describes the procedures of each step mentioned above.

The pairing feature exchange consists of following procedures.

1. A master sends a pairing information such as IO capability, maximum encryption key size to a slave through a PairingRequest packet.
2. The slave that has received a request from the master also sends the pairing information to the master through a PairingResponse packet.
3. Devices determine a method that will be used in the STK generation step.

Fig. 2 shows a STK generation step using the passkey entry method. It consists of following procedures.

1.1 A master generates a random number named as a Mrand and sets a TK as a user input passkey.

1.2 A slave also generates a random number named as a Srand and sets a TK as a user input passkey.

2.1 The master generates a confirm value named a Mconfirm using a c1 function.

2.2 The slave also generates a confirm value named a Sconfirm using a c1 function.

3.1 The master sends a Mconfirm value to the slave through a PairingConfirm packet.

3.2 The slave that has received the Mconfirm value sends the Sconfirm value to the master through the PairingConfirm packet.

4. After the master receives the PairingConfirm packet from the slave, it sends the Mrand value to the slave through a PairingRandom packet.
5. The slave calculates a confirm value using the received Mrand to compare whether it matches the received Mconfirm value or not. If the values match, then slave sends the Srand value to the master through the PairingRandom packet.
6. The master calculates a confirm value using the received Srand and compare whether it matches the received Mconfirm value or not.
7. Each side generates the STK using the s1 function with the TK, the Mrand, and Srand.
8. A link is encrypted by the AES encryption algorithm [10] with the generated STK.

In the transport specific key distribution step, based on the exchanged information in the first step, the LTK and other necessary keys are distributed. The slave sends keys to the master and then the master sends keys to the slave.

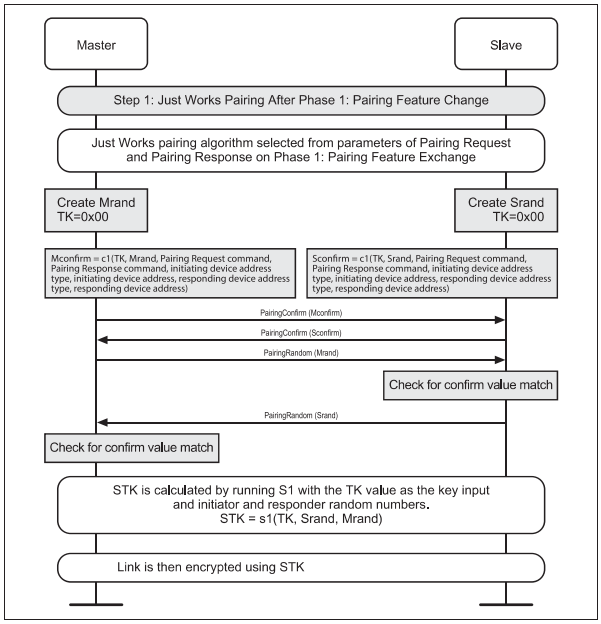

Fig. 2. STK generation step using the passkey entry method

2.4 Vulnerability

In the pairing process, the TK is used to generate the STK. In fact, an attacker never knows the TK because it is an input value of user not transmitted through a packet. However, the length of the TK is too short and this means that the TK is predictable. So it can cause a fatal security vulnerability of BLE. The Just works method always uses predefined value (0x00) as the TK. Although the Passkey entry method is used, it uses up to 6 digits and this has approximately 20 bits of entropy. So, it is easy to find the TK using the brute force method within only a brief time. If a device has successfully generated the STK without an attack, a connection can be maintained safe through using AES encryption algorithm. However, if an attacker tries to eavesdrop from the beginning of the pairing process and captures all of the transmitted packets, then the attacker can find the TK. This is possible because all of the information is in the transmitted packets to generate the STK except the TK. So the attacker can find the TK through calculating a Mconfirm or a Sconfirm and comparing with transmitted values like the followings.

1. Assigning a random value to a TK and calculating the Mconfirm or the Sconfirm using the c1 function.
2. Compairing the calculated value with a transmitted value by the master or the slave.

If the TK can be found, then the STK can be calculated. It means that the encrypted link for the transport specific key distribution step can be decrypted. This can cause the leakage of a data.

3. Proposed Method

In BLE, the security vulnerability in pairing process results from the short length of the TK. Thus we propose an improved security method to increase the length of the TK. The proposed method is only applicable to the following environment.

1. The passkey entry is used.
2. A device represents the TK value on its display and the other device inputs the TK value that is represented on the other side device.

The proposed method also requires an amendment of the present packet format. In the proposed method, we define a security level field which has four level in the PairingRequest and the PairingResponse packet like Table 1. We use 2 bits for new defined field using reserved 3 bits in these packets.

Table 1: Security level flag

Security level flag	Length of the TK
00	6 digits
01	12 digits
10	18 digits
11	24 digits

We also need a new type of packet. So, we define the new packet called as PasskeyRequest by utilizing the reserved code in the SMP command code. It has two fields and each field is like the followings.

1. Code

This field indicates the type of packet and its value is 0x0F.

2. Phase

This field indicates the phase number that an input device requests.

The proposed method follows the next procedures. A master transmits requiring security level through the security level field in the PairingRequest packet. Also, a slave transmits requiring security level through the security level field in the PairingResponse packet. If each device requires different security level, the lower security level is used. In the proposed method, each device has IO capabilities for using passkey entry method. If the determined security level is 1, it proceeds similarly to the conventional passkey entry methods. If security level is greater than 1, each device performs the following procedures.

1. Set the TK to 0x00.
2. After a device which inputs digits (request device) sets phase field of PasskeyRequest packet to 00 indicating the first phase, the request device transmit the packet to a device which represents digits on the display (response device)
3. The response device which has received the PasskeyRequest packet displays the requested phase that consists of digits on the its display.
4. The request device inputs digits that displayed on the response device. After each device concatenates the digits to the TK, it checks whether the security level is satisfied or not.
5. If the security level is satisfied, each device proceeds remaining processes. If not, the request device sets the phase field of PasskeyRequest packet to value indicating the next phase and transmits to the response device.
6. Repeat 2-5 process until the security level is satisfied.

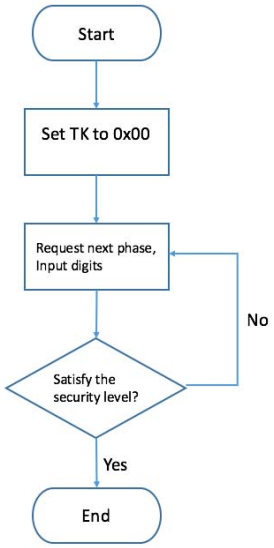

Fig. 3. The procedures of the proposed method

These procedures are the same as Fig. 3. The length of the TK is up to 24 digits when the proposed method is used. In the conventional BLE protocol, the TK is only up to 6 digits and this means that it can be predictable by comparing just 1,000,000 times. Based on an experiment, we prove that It takes less than 20 seconds in a desktop computer with 3.4 GHz CPU. However, the time to find the TK increases exponentially according to the length of the TK. Table 2 shows the calculating time for finding the TK according to its length. If the length of TK increases to 12, 18, and 24 digits through the proposed method, then the calculating time increases to 220 days, 632,900 years, and 600,000,000,000 years. Actually, finding of the TK becomes unfeasible when the length of the key is increasing only to 12 digits. It means that the proposed method ensures a secure data communication.

Table 2. Comparison of calculation time according to the length of the TK

Length of the TK	Calculation time
6 digits	20 sec
12 digits	220 days
18 digits	632,900 years
24 digits	600,000,000,000 years

4. Conclusion

BLE has a security vulnerability that TK is predictable because its length is too short. In this paper, based on the current BLE standard, we describe the security vulnerability. To solve this problem, we propose an improved security method. In the proposed method, it is possible to solve the security vulnerability through increasing the size of TK by inputting TK repeatedly. If the length of TK increases through the proposed method, the time for finding TK increases to 600,000,000,000 years. Thus, finding of TK becomes unfeasible. Also, the proposed method can be implemented through an amendment of BLE standard for supporting a backward compatibility. Therefore, it is expected that the proposed method can complement the security method for systems and services based on BLE.

Acknowledgement

This work was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (No. NRF-2015R1D1A1A01059473)

References

[1] Ashton, Kevin. "That 'internet of things' thing." *RFiD Journal* 22.7, 2009, pp. 97-114.

[2] Gubbi, Jayavardhana, et al. "Internet of Things (IoT): A vision, architectural elements, and future directions." *Future Generation Computer Systems* 29.7, 2013, pp. 1645-1660.

[3] Chang, Kuor-Hsin. "Bluetooth: a viable solution for IoT?[Industry Perspectives]." *Wireless Communications*, IEEE, 21.6, 2014, pp. 6-7.

[4] Galeev, Mikhail. "Bluetooth 4.0: an introduction to bluetooth low energy-part I." *EE Times*, Design, 2011.

[5] Nieminen, Johanna, et al. "Networking solutions for connecting bluetooth low energy enabled machines to the internet of things." *Network*, IEEE 28.6, 2014, pp. 83-90.

[6] Bisdikian, Chatschik. "An overview of the Bluetooth wireless technology." *IEEE Commun Mag* 39.12, 2001, pp. 86-94.

[7] Kamath, Sandeep, and Joakim Lindh. "Measuring bluetooth low energy power consumption." *Texas instruments application note AN092*, Dallas, 2010.

[8] Siekkinen, Matti, et al. "How low energy is bluetooth low energy? comparative measurements with zigbee/802.15. 4." *Wireless Communications and Networking Conference Workshops (WCNCW)*, IEEE, 2012, pp. 232-237.

[9] Bluetooth SIG. (2014). *Bluetooth Core Specification Version 4.2*. [Online]. Available: http://bluetooth.com.

[10] Announcing the Advanced Encryption Standard (AES), *FIPS PUB 197*, 2001.