Seamless and Secure Bluetooth LE Connection Migration

Syed Rafiul Hussain[1], Shagufta Mehnaz[1], Shahriar Nirjon[2], Elisa Bertino[1]
[1]Purdue University, [2] UNC Chapel Hill
hussain1@purdue.edu, smehnaz@purdue.edu, nirjon@cs.unc.edu, bertino@purdue.edu

ABSTRACT

At present, Bluetooth Low Energy (BLE) is dominantly used in commercially available Internet of Things (IoT) devices – such as smart watches, fitness trackers, and smart appliances. Compared to classic Bluetooth, BLE has been simplified in many ways that include its connection establishment, data exchange, and encryption processes. Unfortunately, this simplification comes at a cost. For example, only a star topology is supported in BLE environments and a peripheral (an IoT device) can communicate with only one gateway (e.g. a smartphone, or a BLE hub) at a set time. When a peripheral goes out of range, it loses connectivity to a gateway, and cannot connect and seamlessly communicate with another gateway without user interventions. In other words, BLE connections do not get automatically migrated or handed-off to another gateway. In this paper, we propose a system which brings seamless connectivity to BLE-capable mobile IoT devices in an environment that consists of a network of gateways. Our framework ensures that unmodified, commercial off-the-shelf BLE devices seamlessly and securely connect to a nearby gateway without any user intervention.

Keywords

BLE; IoT; Connection migration

1. INTRODUCTION

The Internet of Things (IoT) has entered the commercial market much faster than expected. The IoT industries predict that the total number of 'smart things' will be more than 30 billion by the year 2020. In a typical scenario, an IoT device connects to a gateway (e.g., a smartphone or a smart hub) over a low-power wireless network, and the gateway enables its access to the Internet. Because the connection process between an IoT device and a gateway requires active engagement of a user, *seamless connectivity of mobile IoT devices in a network of gateways* is still not happening. Ideally, an IoT device should be able to seamlessly communicate with a nearby gateway, without requiring an end-user to enter pins and passwords every time it moves near a different gateway in the same trusted network environment.

There are a number of wireless protocols such as Bluetooth LE (BLE), ZigBee, NFC, that have been used in different IoT communication scenarios. Among these, BLE is the most popular choice because of its simplicity, openness, and a promised battery life of multiple years. The BLE protocol allows multiple devices ('peripherals') to attach themselves to a single gateway (the 'central'), but it restricts the mobility of the peripherals outside and into the range of a gateway. Moving the gateway along with a mobile IoT device seems like an option, but it is not always feasible as it causes disconnections of other IoT devices that are static or moving in a different direction from the gateway.

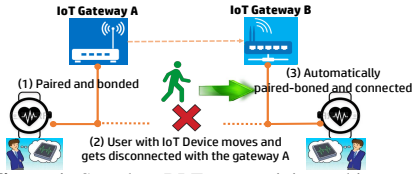


Figure 1: Seamless BLE connectivity architecture.

In order to ensure continuous BLE connectivity, Zachariah et al. [5] proposed an architecture where an IoT device may connect to multiple gateways located at different places. However, establishing a distinct connection with every gateway requires a peripheral to reset and broadcast advertising signals separately for all the gateways. Even if connections with multiple gateways is made possible by hacking [1], it comes at the cost of disconnecting the device from its previous gateway and then connecting to a new one. This incurs significant CPU, memory, energy, and bandwidth overhead in resource constrained IoT devices as even a single connection establishment requires advertisements, discovery, pairing and bonding [4], and several mutual agreements in different layers of BLE protocol stack. In addition, the process requires repeated manual interventions that disrupt the ongoing communication between a device and a remote service [2]. Because of these practical issues, we argue that an IoT device should be able to seamlessly communicate with different gateways [5] without requiring to create a separate connection with each of them.

In this paper, we propose a system which enables seamless BLE connection migration for mobile IoT devices in a network of BLE gateways. The contributions of this paper are the following:

- We propose a framework that ensures seamless communication between an unmodified, BLE-enabled mobile IoT device and a remote service in a network of BLE gateways, without requiring pairing-bonding and connections to individual gateways.
- We propose two approaches – *full stack cloning* and *partial stack cloning* for capturing a snapshot of connection states at the current gateway and then transferring and updating them at the next gateway during BLE connection migration.
- We propose a gateway selection mechanism for transferring the connection state to the most suitable gateway when an IoT device requires to migrate its connection.

2. USAGE SCENARIOS

- *In Hospitals:* Patients wearing BLE devices in hospitals can be

CODASPY'17 March 22-24, 2017, Scottsdale, AZ, USA

ACM ISBN 978-1-4503-4523-1/17/03.
DOI: http://dx.doi.org/10.1145/3029806.3029840

localized and monitored using a network of gateways deployed at different locations in the hospital.

- *In Airports:* Upon arrival at the airport, passengers (and baggage) who are equipped with BLE beacons can voluntarily report their location and status to the deployed gateways from anywhere within the airport, and in return, they get personalized services and notifications.
- *In Theme Parks:* With the help of a BLE-enabled wristband on the child and static gateways deployed at different locations inside a theme park, parents can monitor and locate their missing kids via their mobile phone.

3. BLE BACKGROUND

Roles of BLE Device: A BLE device assumes either a *peripheral* or a *central* role. A peripheral, typically an IoT device such as a heart rate monitor, a blood pressure monitor, a smart lock, or a smart watch, comes with limited capabilities and contains advertisement information. A central device such as an access point, a personal computer, or a smartphone, scans for BLE advertisements, receives an advertisement, and initiates a connection.

Modes of Communication: Two modes of communication are available: *broadcast* and *connected* modes. The broadcast mode enables a peripheral to send data to any other device listening for transmissions. If two devices need to exchange data they can use the *connected* mode.

Pairing: In *connected* mode, if two devices want to exchange data securely, they perform a *pairing* process which results into a number of keys shared between the peripheral and the central.

Bonding: It is the process of storing the keys created during pairing for use in subsequent connections.

Privacy: BLE can use *Random Device Addressing* for privacy of connections and prevent 'tracking' based on the assumption that eavesdropping did not occur during pairing process.

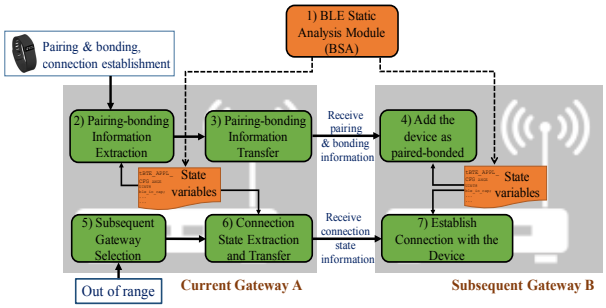


Figure 2: Workflow of connection migration

4. SYSTEM DESIGN

4.1 Threat model

We consider a strong threat model where only IoT gateways and IoT devices are parts of a trusted computing base. Also, we consider that the adversaries have capabilities of injecting unauthenticated packets, modifying legitimate packets, and sniffing messages.

4.2 Workflow

Figure 2 depicts the workflow of connection migration process.

Identify Connection State Information: In an off-line, one-time step, we analyze the BLE source code to identify the set of variables required for both pairing-bonding and connection information transfers. The *BLE Static Analysis (BSA)* module performs this static analysis on BLE source code and the identified set of variables (that are required for connection transfers) are stored in each of the gateways.

Extract Pairing-Bonding Information: A BLE central, serving as a gateway scans for peripheral devices that broadcast advertisement packets so that the central can connect with the peripherals and receive the desired GATT services. We name the gateway to which the IoT device is currently connected as the *current* gateway. In order to ensure secure data transfer, the current gateway initiates pairing and bonding procedures as shown in Figure 3(a). After creating connection, the current gateway extracts the pairing-bonding related information.

We instrument the BLE source code to obtain the runtime values of the pairing-bonding variables for a connected IoT device (as shown in steps (2) in Figure 2). The current gateway stores the extracted information into memory and sends them to subsequent gateways. The runtime for this extraction module is distributed across different layers of BLE protocol stack.

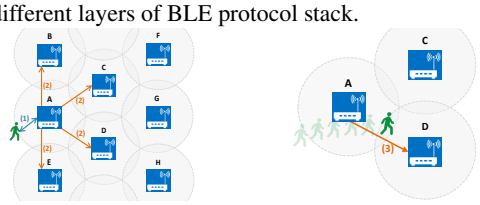


Figure 3: a) Pairing and bonding with gateway `A` and sharing of pairing-bonding information with gateways `B`, `C`, and `D`. (b) Subsequent gateway selection and connection transfer as the user with IoT device moves from gateway A to gateway D.

Disseminate Pairing-Bonding Information: The current gateway disseminates the pairing-bonding information to a set of gateways that are candidates for the subsequent gateway (the next gateway to which the IoT device may connect). This pairing-bonding information consists of both the bonded device's information as well as a subset of state variables. In Figure 3(a), the gateways that are in the vicinity of gateway `A` are the gateways `B`, `C`, `D`, and `E`. Therefore, the candidate gateways are `B`, `C`, `D`, and `E` but not the gateways `F`, `G`, or `H`.

Add as a Bonded Device: Upon reception of the pairing-bonding information, the candidate gateways `B`, `C`, `D`, and `E` store these information mapped with the Bluetooth device address of that IoT device so that whenever that device needs service from these gateways, they do not have to execute the pairing-bonding procedures. Note that the candidate gateways do not initiate connection at this stage since they do not have connection state information.

Subsequent Gateway Selection: If an IoT device moves during or after connection establishment, the current gateway or the IoT service providing cloud system is able to estimate the device's moving direction [3]. We leverage this mechanism and examines a device's locations at recent timestamps to infer the moving direction. Location information of IoT devices can also be obtained using existing indoor and outdoor localization techniques. Analyzing the movement direction and speed of the IoT devices, the current gateway or the service provider selects the subsequent gateway among the candidate gateways to whom the connection information will be transferred. As shown in Figure 3 (b), the current gateway `A` transfers connection information to the subsequent gateway `D`.

Extract and Transfer Connection State: The current gateway identifies the current state (or snapshot) of the connection and transfers the required state variables to the subsequent gateway so that the subsequent gateway can reconstruct the connection state with the same peripheral. The extraction of runtime values of connection state variables follows the same procedure as the extraction of pairing-bonding information.

Establish Connection with Subsequent Gateway: Upon reception of the connection state information, the subsequent gateway creates required objects related to connection, updates the connection state variables, and stores the connection information into gateway's non-volatile memory (NVRAM). As a result, the peripheral gets seamlessly connected to the subsequent gateway.

4.3 BLE Stack Cloning

Our system provides two modes for connection state extraction: *full stack cloning*, and *partial stack cloning*. Full stack cloning refers to cloning states of all the layers of Bluetooth stack starting from the application layer down to the link layer whereas partial stack cloning refers to the cloning of Bluetooth stack starting from the application layer down to the L2CAP layer.

4.4 Secure Connection Information Sharing

While sharing the pairing-bonding and connection information, the receiver gateways can be categorized into two groups: *trusted* and *untrusted* gateways.

Trusted Gateway: If the receiver gateways belong to the same cluster of gateways as the current gateway, the receiver gateways do not need to further authenticate themselves. They already share a secret group key with which they encrypt the data and then distribute the data in encrypted form securely. This secret group key can be shared between gateways through WiFi or 4G/LTE communication network and thus do not require any change in the existing Bluetooth protocol. An example of such data transfer is following:

$$Enc\_D \leftarrow E_{K_{grp}}(D||nonce), D||nonce \leftarrow D_{K_{grp}}(Enc\_D) \quad (1)$$

where D is the data to transfer, and K_{grp} is the symmetric group key. In Eqn. 1, the current gateway encrypts the data using K_{grp} and transfers Enc_D to the receiver gateway(s). A trusted receiver gateway has knowledge of K_{grp}, and thus decrypts Enc_D to D as shown in Eqn. 1.

Untrusted Gateway: If the subsequent gateway to which the BLE connection is going to be migrated is not already trusted, the current gateway needs to verify the public key certificate of the subsequent gateway. Upon certificate validation, the current gateway use the public key cryptography protocol as the following to share the bonding and connection information securely.

$$Enc\_D \leftarrow E_{PK_{rcv}}(D||nonce), D||nonce \leftarrow D_{SK_{rcv}}(Enc\_D) \quad (2)$$

where D is the data to transfer, and PK_{rcv} is the public key of the receiver gateway. In Eqn. 2, the current gateway encrypts the data using PK_{rcv} and transfers Enc_D to the receiver gateway. The receiver gateway has knowledge of the corresponding secret key, SK_{rcv}, and thus decrypts Enc_D to D as shown in Eqn. 2.

5. EVALUATION

Experimental Setup We use five Nexus 5 phones as gateways (i.e., BLE centrals), one Nexus 6 phone and one Alcatel Onetouch tablet as IoT devices. Nexus 5 phones have only the BLE central feature whereas Nexus 6 and Alcatel Onetouch tablet have both the BLE central and the BLE peripheral capabilities. We use the *nRF Connect* application downloaded from the Google Play store for BLE peripherals. For the gateways, we have developed a custom application. We use the heart rate monitoring service that periodically sends heart rate measurement in a single BLE packet of size 20-bytes every second. Each data point reported in the experiment is obtained by taking the average of at least five runs.

Migration Success Rate: We found every connection migration request successful when IoT devices are both static and moving at different speeds.

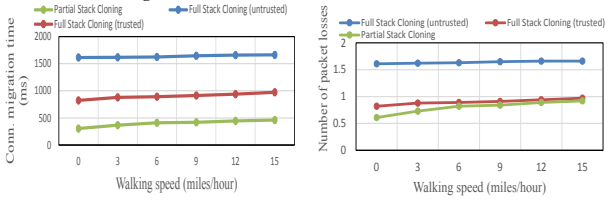
Extra bytes required for connection migration: In both *partial stack cloning* and *full stack cloning*, the current gateway sends a 512-bytes of blob containing the bonding related information to each of its neighbors. However, for *full stack cloning*, the current gateways sends a 2048-bytes of blob containing values of all the connection related variables to the next gateway.

Time required for adding a peripheral as a bonded device: To add the peripheral as a bonded device requires the IoT gateway needs to load the device information, e.g., device address, device type, address type, and keys from the main memory and then store them into the gateway's NVRAM for using in future communications. Table 1 shows the mean time required by IoT gateways of different device types to load a peripheral.

Gateway	Loading Time (ms)	Storing Time (ms)	Total Time (ms)
Nexus 5	40.5	19.1	60.4
Nexus 6	36.7	17.4	54.1
Alcatel OneTouch	43.2	20.3	63.5

Table 1: Time required for adding a peripheral as a bonded device.

Time required for connection migration: Figure 4(a) shows the connection migration time increases with the increase of users speed as there are more packet losses associated with increased mobility of users. Also, the connection migration time for *partial stack cloning* is smaller than that of the *full stack cloning* because creating a new connection between the subsequent gateway and the IoT device does not require any cryptographic operation in case of *partial stack cloning*.



(a) (b)

Figure 4: a) Time required for connection migration. (b) Number of packet losses when data packets are sent with 1 second time interval.

Packet loss: Fig. 4(b) shows the number of packet losses when BLE packets are sent from the IoT device to the gateway every after 1 second. *Full stack cloning* with untrusted gateway causes upto 2 packet losses which is about 2X of the other scenarios.

5.1 Security Analysis

Adversaries cannot inject or modify packets. Our system ensures that the gateways always perform secure communication with BLE devices through the long term keys established by pairing-bonding procedure.Thus maliciously injected or modified BLE packets by adversaries are always identified.

Adversaries cannot derive paring-bonding and connection information. The gateways use non-deterministic encryption to securely transfer the pairing-bonding information. Therefore, adversaries cannot derive the long term keys and the connection related values to impersonate a legitimate gateway.

6. CONCLUSION

In this paper, we focus on the problem of IoT devices being unable to connect to multiple gateways seamlessly and thus propose a framework that ensures seamless and secure communication between a mobile IoT device and a remote service in a network of BLE gateway environment.

7. REFERENCES

[1] W. Albazrqaoe, J. Huang, and G. Xing. Practical bluetooth traffic sniffing: Systems and privacy implications. In *MobiSys, 16*.

[2] P. A. Kodeswaran, R. Kokku, S. Sen, and M. Srivatsa. Idea: A system for efficient failure management in smart iot environments. In *MobiSys '16*.

[3] Y. Tao, C. Faloutsos, D. Papadias, and B. Liu. Prediction and indexing of moving objects with unknown motion patterns. In *SIGMOD '04*.

[4] J. L. Yan Michalevsky, Suman Nath. Mashable: Mobile applications of secret handshakes over bluetooth le. In *MobiCom*, 2016.

[5] T. Zachariah, N. Klugman, B. Campbell, J. Adkins, N. Jackson, and P. Dutta. The internet of things has a gateway problem. In *HotMobile'15*.