International Journal of Networking and Computing – `www.ijnc.org`, ISSN 2185-2847
Volume 10, Number 1, pages 25–43, January 2020

An Adaptive Connection-Establishment Timeout Configuration Method for Bluetooth MANETs in Control Packet Loss Environments

Temma OHTANI†, Eitaro KOHNO†, Akifumi NOMASAKI‡, and Yoshiaki KAKUDA†

†Graduate School of Information Sciences, Hiroshima City University,
‡Faculty of Information Sciences, Hiroshima City University
†‡3-4-1, Ozuka-Higashi, Asaminami-Ku, Hiroshima, 731-3194, Japan

Received: February 15, 2019
Revised: May 6, 2019
Revised: July 31, 2019
Accepted: September 10, 2019
Communicated by Yuji Yamato

Abstract

Bluetooth MANETs, which consist of Bluetooth-enabled terminals, are a prospective methodology for mobile ad hoc networks (MANETs). Since Bluetooth is a connection-oriented and a low-power-consumption communication method, terminals must execute time-consuming connection establishment procedures in advance. We have to solve the following two problems for Bluetooth MANETs: (1) since terminals move within fields, terminals must establish their connection within a limited time. In addition, (2) since Bluetooth has a shorter range than other technologies, established connections are easily disrupted. In order to solve problem (1), a low-latency connection establishment method has been proposed. However, there is no countermeasure for problem (2). Therefore, the possibility of rapid re-establishment for connections ought to be investigated.

In this paper, we have proposed a new adaptive connection-establishment timeout configuration method for environments with control packet loss. Our proposed method employs an adjustment mechanism for control packet loss. We have also implemented our proposed method on Raspberry Pi in order to evaluate our proposed method. To design our proposed method, we have conducted preliminary experiments to investigate the effects on varied connection-establishment timeouts. As a result, we have found that the minimum connection-establishment timeout is 2 seconds. At the same time, we have found that connection establishment failure rates increased as connection-establishment timeouts became shorter than 2 seconds. Especially, when connection-establishment timeouts were less than 2 seconds, the latency of the connection establishment became extremely long. This is due to the presence of a race condition between the processing time of control packets and the connection-establishment timeouts. From these results, we have set an adjustable interval of connection-establishment timeouts between 2 to 5 seconds. 5 seconds is the native default value of Bluetooth's connection-establishment timeout. We have also conducted experiments of our proposed method in varying packet loss rate scenarios, and have confirmed the effectiveness of our proposed method.

Keywords: Bluetooth MANETs, Adaptive connection-establishment timeout configuration, Control packet loss

1 Introduction

Mobile ad hoc networks (hereinafter, referred to as MANETs) [15] [3] [6] are a prospective network technology for disaster response. MANETs also can be used for normal situations and MANET-based applications have been developed recently [10]. In disaster response scenarios, a terminal must use a low-power communication protocol. Bluetooth [1] is one of the prospective solutions to fulfill this requirement. We have researched Bluetooth MANETs which consist of Bluetooth-enabled terminals for several years [10]. In papers [12] and [11], communications among user-side terminals such as smartphones and Raspberry Pi-based terminals, utilizing Bluetooth MANETs for communications, have been proposed. As we mentioned above, while Bluetooth is a prospective communication technology for MANETs, there are some important issues that need to be addressed: (1) latency of communication establishment and (2) countermeasures for frequently occurring disruptions. So far, some solutions to these issues that use a combination of Classic Bluetooth and Bluetooth Low Energy [8] have been proposed. Some of our solutions had no control packet-loss under ideal conditions. Terminals can establish a connection with a two-second latency in a two-terminal environment. In real networks, communication between terminals can be disrupted by environmental factors such as interference and obstacles. Furthermore, in order to implement Bluetooth MANETs using real terminals, we have to consider the difference in both processing time of each terminal and the environmental factors. These differences can cause variations in the number of failures of connection-establishment procedures.

In this paper, we propose a new adaptive connection-establishment timeout configuration method for control packet loss-environments. Our proposed method employs an adjustment mechanism to counter control packet loss. In order to design our proposed method, we conducted preliminary experiments to investigate the effects on varied connection-establishment timeouts in advance. From the results, we set adjustable intervals of connection-establishment timeouts. We also conducted experiments using our proposed method in varying packet loss rate scenarios. We conducted experiments to confirm the adaptability to varying packet loss rates of our proposed method through a Raspberry Pi-based application. Furthermore, since terminals have mobility, terminals may drift apart from their corresponding terminal's radio area during the connection establishment procedure. Since our existing connection establishment method of Bluetooth MANETs utilizes a combination of Classic Bluetooth and Bluetooth Low Energy, the difference between those radio ranges can easily cause control packet loss. Control packet loss leads to connection establishment failure in Bluetooth MANETs. Connection establishment failure, then results in a longer lentency as terminals must wait until they time out before re-establishing a connection.

The rest of the paper is organized as follows: in Section 2, we explain the existing proposals of Bluetooth MANETs; In Section 3, we describe the application on the Raspberry Pi-based system; In Section 4, we show the results of the preliminary experiments; In Section 5, we describe our proposed method and experiments; In Section 6, we discuss the effectiveness of our proposed method by using experimental results; In Section 7, we summarize our paper.

2 Bluetooth MANETs and application development

2.1 Bluetooth and its characteristics

Bluetooth is a communication protocol for personal area networks between 10 and 100 meters wide [1]. Bluetooth has several versions such as Classic Bluetooth (hereinafter, referred to as "Classic") [1] and Bluetooth Low Energy (hereinafter, referred to as "BLE") [5]. Classic is the conventional standard and BLE is a newer standard since Bluetooth ver. 4.0 [4]. While both Classic and BLE are referred to as Bluetooth, which use the 2.4 (GHz) band, they are different technologies. Thus, there are several differences. We summarized the parameters between Classic and BLE in Table 1.

Table 1: Specifications of Classic and BLE

Item	Value	
	Classic	BLE
Frequency (GHz)	2.4	
Number of channels	79	40
Standard maximum speed (Mbps)	1	1
Effective maximum speed (Mbps)	0.7	0.27
Maximum transmission power (mW)	100	10
Pairing	Required	Option
Broadcast when disconnected	impossible	possible
Maximum packet size (Byte)	1021	47

2.1.1 Classic Bluetooth (Classic)

Classic provides a connection-oriented bidirectional communication method. In Classic, terminals must connect in advance of data sharing. The bandwidth and payload size of Classic are greater than those of BLE (Table 1).

In Classic, a terminal has exclusive states. In paper [14], the states Discovering and Discoverable have been proposed. A terminal in the Discovering state can discover its surrounding terminals which are in the Discoverable state. Likewise, a terminal in the Discoverable state can be discovered by other terminals in the Discovering state. A terminal cannot be in the Discovering and the Discoverable state, simultaneously. In Classic, terminals exchange their IEEE 802-2014 standard-based fixed and unique public device addresses [2].

2.1.2 Bluetooth Low Energy (BLE)

BLE provides both connection-oriented bidirectional and broadcast-based unidirectional communication methods. Just as in Classic, a BLE terminal has its states. However, the role and behavior of its states are quite different from Classic. In this paper, our proposed method utilizes the BLE-Broadcast method. In BLE-Broadcasts, a terminal with the state Advertising can broadcast advertising packets to surrounding terminals. In BLE-Broadcast, terminals that are in the state of Advertising will not use terminal addresses.

2.2 Classic Bluetooth MANETs

Classic Bluetooth MANETs are one type of Bluetooth MANETs which consist of only Classic-enabled terminals. Fig. 1 shows the typical sequence of connection establishment of Classic Bluetooth MANETs. In Classic Bluetooth MANETs, terminals establish their connection as follows:

1. A terminal (terminal A as shown in Fig. 1) discovers Discoverable terminal(s) which are located in the surrounding terminal's radio area (terminal B as shown in Fig. 1).

2. Terminal A registers terminal B in terminal A's adjacent terminal list.

3. When terminal A enters the Discoverable state, it refers to its own adjacent terminal list and sends a connection establishment request packet to terminal B.

4. When terminal B accepts the request and sends back a connection establishment acknowledgment packet to terminal A, the connection between terminals A and B is established.

Table 2: Latency of Bluetooth MANET [9]

	Classic Bluetooth MANETs	BLE-BAFE
Latency (sec)	23.6	3.00

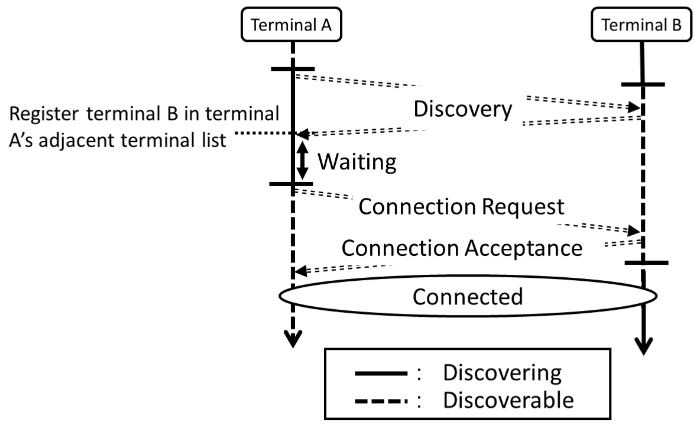


Figure 1: Connection establishment sequence of Classic Bluetooth MANETs

In Classic, a terminal cannot have the Discovering and Discoverable states simultaneously. In Classic Bluetooth MANETs, we have introduced an autonomous switching mechanism with random waiting time in order to mitigate the collision of connection request timings. As shown in Table 2, however, the connection-establishment latency of Classic Bluetooth MANETs is almost 8 times longer than that of the below-mentioned BLE-Broadcast assisted fast connection establishment method(BLE-BAFE), which is described in Section 2.3.

2.3 BLE-Broadcast assisted fast connection establishment method for Bluetooth MANETs

In paper [9], BLE-Broadcast assisted fast connection establishment method (hereinafter, BLE-BAFE) has been proposed. In BLE-BAFE, a terminal initiates the connection establishment process by sending a BLE-based advertising packet. In BLE, advertising packets contain information such as a terminal's fixed device address to utilize the Classic-based connection establishment procedure. Unlike Classic, BLE can have different terminal states, Advertising and Scanning, simultaneously. BLE-BAFE also replaces the time-consuming Classic-based control packet procedure with BLE-Broadcast. Furthermore, paper [9] shows the proposed BLE-BAFE-based connection establishment timing arbitration method is used to solve deadlock issues. Therefore, BLE-BAFE can establish connections faster than that of Classic Bluetooth MANETs.

So far, several fast connection establishment methods have been proposed. While they are quite effective, some issues have remained. Two problematic issues are the temporal disruption of an established connection and control packet loss.

As we described in Section 2.1, these two technologies are quite different from each other. In BLE-BAFE, when a terminal discovers other terminal(s) at the boundary area of Classic's radio distance, the reachability of connection establishment request packet(s) of Classic will become intermittent. A preliminary investigation of the connectivity of BLE-BAFE through Raspberry Pi-based implementation has been conducted. The results show us that the connection establishment request packet could become lost due to the long distances in real terminal-based experiments. In Bluetooth MANETs, since terminals have mobility, the reachability of connection establishment packet(s) will become unreliable, too. When the acknowledgement packet isn't received by a terminal, it waits for its predetermined connection-establishment timeout before resending a new connection request packet. While BLE-BAFE without modification of the connecton-establishment timeout has been proposed, it may cause too long of a wait time. We have to develop countermeasures to address reachability and mobility issues.

2.4 Application development using Raspberry Pi

The original development of the Bluetooth-MANET-based application was developed on Android OS-based terminals. At that time, the Android API was used. While this is quite effective in expanding the portability of the application, all configurable parameters which are described in the core specification of Bluetooth vol.2 have not been able to be utilized. Through the Android API, we can only use configurable parameters for scanning and advertising in BLE. Moreover, the Android API provides only the coarse-grained interval and transmission power values for scanning and advertising such as LOW, MEDIUM, and HIGH. In addition, parameters for Classic have not been able to be configured through the application.

Conversely, Raspberry Pi with Linux OS allows us a more flexible parameter design than the Android API. Furthermore, there is no limitation for configurable parameters. Thus, protocol design through Raspberry Pi provides us a more flexible development environment.

3 The Application on Raspberry Pi

We have designed and implemented the communication system for Bluetooth MANETs as an application on a Raspberry Pi 3 which is a Linux OS-based terminal (hereinafter, referred to as our proposed system) [12]. Our proposed system is interoperable with Android OS-based applications and has been implemented by using open source software. Fig. 2 shows the relationship between the modules of our proposed system. Our proposed system consists of the following 5 main modules which work concurrently:

Advertising packet transmission processing
This module periodically sends advertising packets to its surrounding terminals. These advertising packets inform the surrounding terminals of the sender terminal's state.

Adjacent terminal discovery processing
A terminal which receives an advertising packet refers to its own list which records the state of adjacent terminals (hereinafter, referred to as an adjacent terminal-state list). When the terminal finds a sender terminal entry in the adjacent terminal-state list, it already has established a connection with it and therefore does not need to initiate a new connection. When an advertising packet is received from an adjacent terminal that is not on the adjacent terminal-state list, the connection establishment procedure is initiated. When the terminal completes the connection establishment process, the module starts over.

Summary vector transmission processing
This module handles the Summary Vector (hereinafter, referred to as SV) of a terminal. In our proposed system, SV is the same as Epidemic Routing in paper [16], and has a set of data packet IDs. In order to generate the terminal's SV, this module refers to the below-mentioned data packet list. This module also periodically performs the procedures.

Data packet transmission processing
When a user of a terminal inputs data (e.g. text messages) to be transferred through our proposed system, this module generates data packets and appends them to a packet list. The packet list also includes received data packets from adjacent terminals. This module refers to the adjacent terminal-state list and sends data packets to connection established terminals.

Data packet reception processing
When a terminal receives a data packet(s) from other terminals, it forwards the data packet(s). As we proposed in paper [12], our system autonomously switches between two types of data

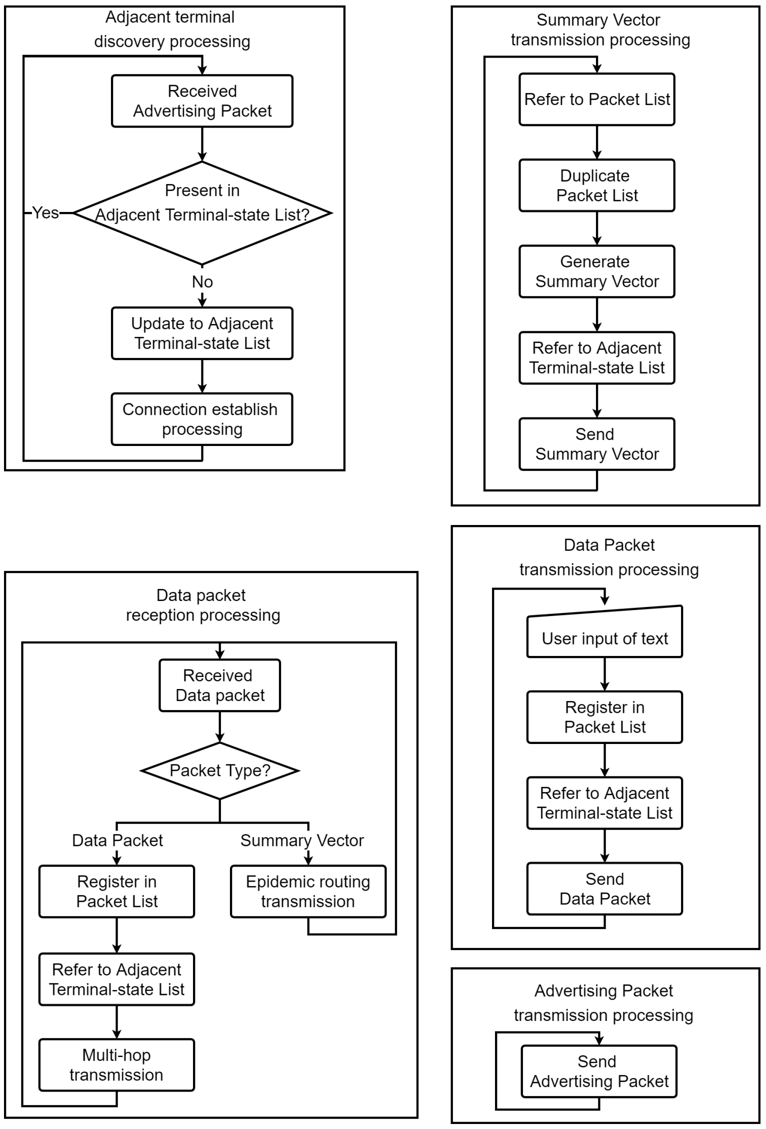

Adjacent terminal discovery processing
Received Advertising Packet
Present in Adjacent Terminal-state List?
Yes
No
Update to Adjacent Terminal-state List
Connection establish processing
Summary Vector transmission processing
Refer to Packet List
Duplicate Packet List
Generate Summary Vector
Refer to Adjacent Terminal-state List
Send Summary Vector
Data packet reception processing
Received Data packet
Packet Type?
Data Packet
Summary Vector
Register in Packet List
Epidemic routing transmission
Refer to Adjacent Terminal-state List
Multi-hop transmission
Data Packet transmission processing
User input of text
Register in Packet List
Refer to Adjacent Terminal-state List
Send Data Packet
Advertising Packet transmission processing
Send Advertising Packet

Figure 2: Procedures of modules in our proposed system

forwarding schemes: multi-hop data packet forwarding with waiting time (Multi-hop transmission as shown in Fig. 2) and Epidemic-routing-oriented store-carry-forward-based data packet forwarding (Epidemic routing transmission as shown in Fig. 2).

In our proposed system, mutual exclusion has been introduced to protect the packet list and the terminal-state list from the simultaneous access of concurrently executed modules.

4 Preliminary experiments [13]

4.1 Overview

In order to determine the configuration connection-establishment timeout interval for use by our proposed method, we have to investigate the relationship between the connection-establishment latency and the connection-establishment timeout through preliminary experiments. To do this, we have implemented a Raspberry Pi 3-based application with our proposed method [9]. In the experiments,

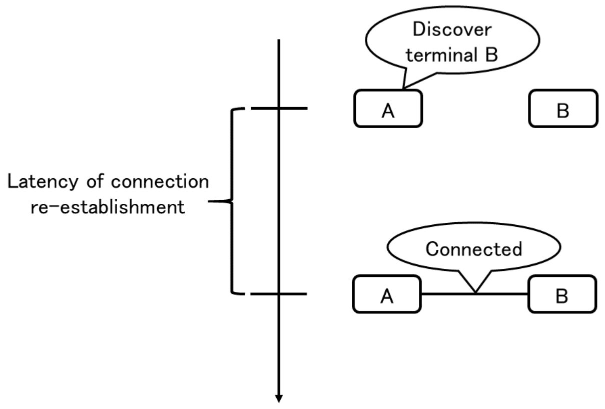


Figure 3: Latency of connection re-establishment

we have measured the latency of connection re-establishment and the connection-establishment time-out. In experiments, we have prepared 5 varying connection-establishment timeout configurations. We define the latency for re-establishing connections as the elapsed time between the discovery of a terminal and the successful establishment of a connection. (Fig. 3)

4.2 Configuration of experiments

We placed two Raspberry Pi 3 Model B terminals (Raspbian 9.1 OS and Bluetooth ver.4.1) within communication range of each other. The time of the experiments was 30 minutes. The connection-establishment timeout was 1, 2, 3, 4 and 5 seconds, respectively. The 5-second connection-establishment timeout is the default setting. We describe the method as shown in paper [9] to measure the latency for re-establishing connections by using Fig. 4. Fig. 4 shows the connection establishment sequence between terminals A and B. In our proposed method, terminals A and B establish their connection as follows:

1. When terminal A discovers terminal B, we start the measurement timer in A.
2. Subsequently, terminal A sends its connection establishment request packet and terminal B discards the packet. This action mimics packet loss.
3. Terminal A waits until connection-establishment timeout occurs.
4. After sequence 3, terminal A re-discovers terminal B.
5. Terminal A re-sends its connection establishment request packet, and terminal B accepts the packet. It then sends back its connection establishment acceptance packet to terminal A.
6. When terminal A receives the acceptance packet, it completes the connection establishment sequence. At this time, we stop the measurement timer.

We have implemented a predetermined random time to disrupt connections after established connections to mimic an environment of varying connectivity in Bluetooth MANETs. In our experiments, we set the predetermined time between 15 to 20 seconds. This value was determined through our field experiments using Bluetooth MANETs in the field.

4.3 Experimental results

Fig. 5 shows the results of experiments for the latency of connection re-establishment vs varying connection-establishment timeout configurations.

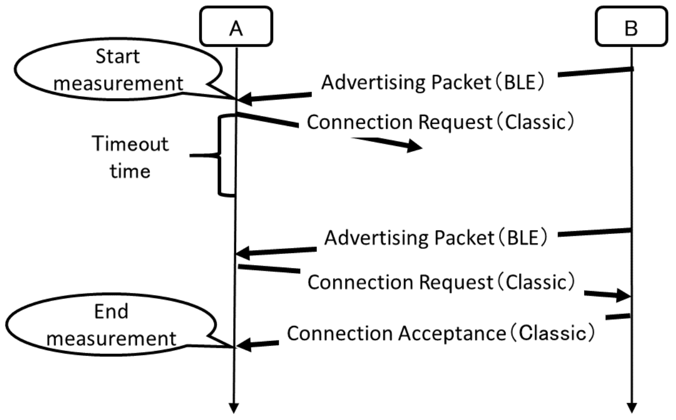


Figure 4: Connection establishment sequence of BLE-BAFE

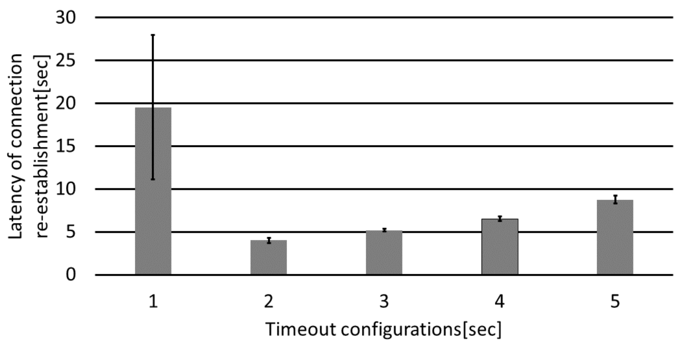


Figure 5: Results of experiments

As we mentioned in Section 4.2, the experimental configurations mimic control packet loss for some reason or another in the connection establishment procedure. Note that the experimental configurations of this section are different from those of paper [8].

The vertical and horizontal axes are the latency of connection re-establishment and connection-establishment timeout setting, respectively. As shown in Fig. 5, the latency of the re-establishment decreases incrementally between the default settings of 5 seconds and 2 seconds. When the connection-establishment timeout configuration is set at 1 second, terminal B could occasionally process the request before terminal A timed out. However, more often them not, terminal A timed out while terminal B was processing terminal A's request. This resulted in terminal A trying another re-establishment request and terminal B responding too late to terminal A's original request. This scenario compounded itself and resulted in a race-condition [13].

We also have measured the number of connection-establishment failures in the aforementioned experiments per varied connection-establishment timeout configurations. Fig. 6 shows the results of the average number of connection-establishment failures in 30 trials. The duration of one trial was 30 minutes. From the research in paper [7], we have learned that a connection-establishment success rate of 90% is acceptable. This is because the throughput of Wi-Fi does not decrease much in a most 10% packet loss environment. However in environments above 10% packet loss, we see dramatically reduced throughput.

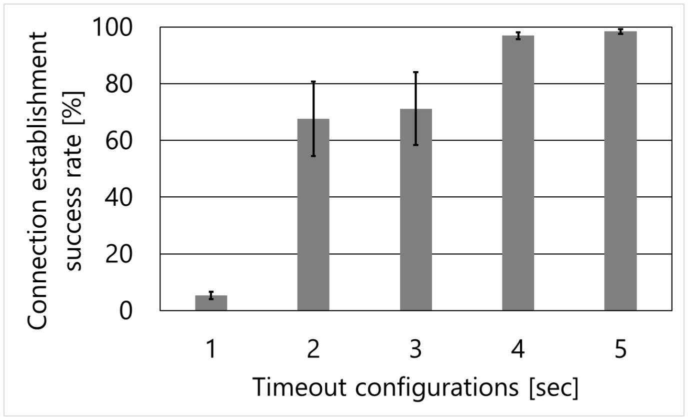


Figure 6: Results of connection establishment success rate per timeout

5 Proposed method

5.1 Overview

As we mentioned in Section 1, we propose a new adaptive connection-establishment timeout configuration method in environments prone to (control) packet loss. As we mentioned in Section 4.3, in real terminals such as Raspberry Pi, when a terminal detects connection-establishment failures, they cannot identify the cause of the failure whether that is simple packet loss or a "race-condition." Thus, our proposed method comprehensively handles varying the "race-condition" and simple packet loss. In order to do this, we have employed a measurement process for connection-establishment latency to adapt to the environmental (temporal) change. In our proposed method, we employed an adjustment mechanism for connection-establishment timeout in Bluetooth.

In addition, we designed our proposed method for real terminal-based application. In general, real terminals such as Raspberry Pi cannot configure precise timeout values such as in the order of microseconds. Furthermore, the real terminal has a limited amount of memory and computational power. Therefore we designed our proposed method to be light-weight.

Our proposed method employs the following adaptive connection-establishment timeout configuration mechanisms: a terminal records its connection-establishment latency and calculates its average value using the 5 most recent times to configure the latest connection-established timeout. When the terminal can establish a connection, it configures a new shorter connection-establishment timeout that uses the average of the 5 most recent times. The minimum connection-establishment timeout is two seconds. If the terminal's connection-establishment procedure fails, it configures a new longer connection-establishment timeout that is the average of the 5 most recent times. The maximum connection-establishment timeout is five seconds, the default value of the connection-establishment timeout of Bluetooth.

5.2 Sequence

We explain the sequence of connection-establishment timeout for our proposed method as shown in Fig. 7.

1. A terminal judges whether the latest connection-establishment procedure was a success or a failure.

2. When the connection-establishment procedure is completed successfully, the terminal measures the latency. Here, let x_t be the measured latency of the terminal. Then the terminal records the modified latency X_t. When x_t is smaller than the minimum connection-establishment timeout (2), the terminal records the minimum connection-establishment timeout (2) as the

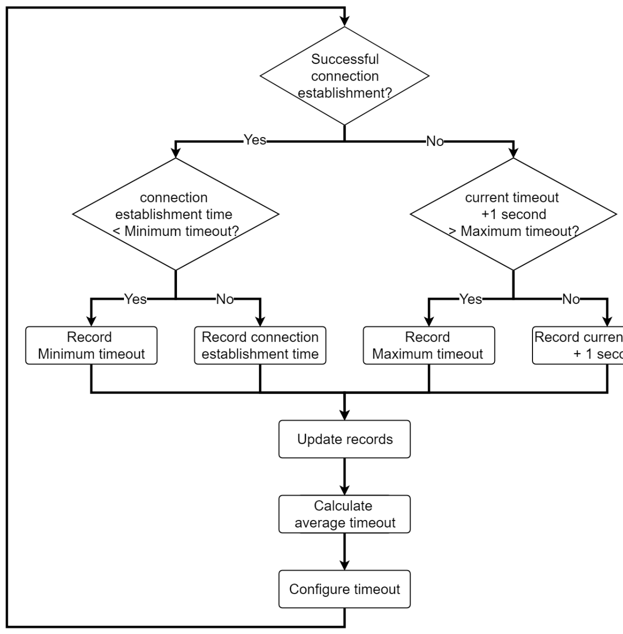

Successful connection establishment?
Yes
No
connection establishment time < Minimum timeout?
current timeout +1 second > Maximum timeout?
Yes
No
Yes
No
Record Minimum timeout
Record connection establishment time
Record Maximum timeout
Record current timeout + 1 second
Update records
Calculate average timeout
Configure timeout

Figure 7: Sequence of our proposed method

new connection-establishment timeout ($X_t = 2$). When x_t is larger than or equal to the minimum connection-establishment timeout (2), the terminal records the measured connection-establishment timeout ($X_t = x_t$).

When the connection-establishment procedure fails, the terminal measures the latency.

When $x_t + 1$ is larger than the maximum connection-establishment timeout (5), the terminal records the maximum connection-establishment timeout (5) as the new connection-establishment latency ($X_t = 5$). When $x_t + 1$ is smaller than or equal to the maximum connection-establishment timeout (5), the terminal records the measured connection-establishment latency ($X_t = x_t + 1$) as the new connection-establishment timeout. The reason for the modified latency $X_t = x_t + 1$ is in general cases to gradually increase the terminal's new timeout value when the connection-establishment procedure fails.

3. The terminal then re-calculates a new average connection-establishment timeout by using the 5 most recent latencies. The terminal calculates its new timeout value TO_t by the following equation (1).

$$TO_t = \frac{1}{5}\sum_{i=0}^{4} X_{t-i} \tag{1}$$

4. The terminal updates its new connection-establishment timeout (TO_t).

5.3 Experiments

5.3.1 Overview

In order to confirm the effect of our proposed method in varying (control) packet loss environments, we have implemented our proposed method with an adaptive connection-establishment timeout configuration on Raspberry Pi 3 devices. In the experiments, we have conducted varying control packet loss scenarios and measured the connection-establishment timeout. We also have measured the number of the connection-establishment failures in each packet loss interval.

5.3.2 Configuration of experiments

In the experiments, we prepared two Raspberry Pi 3 terminals and placed them adjacent to each other. We also conducted 4 20-minute experiments. Within each 20-minute trial, we changed the control packet loss rate as follows: from the start of the trial to 5 minutes: 0%; from 5-10 minutes: 50%; from 10-15 minutes: 0%; from 15-20 minutes: 90%. In the experiments, the connection-establishment timeout is an individually measured value and the number of failures is the total number of the measured values in each segment period. The specifications of the devices (Raspberry Pi 3) were the same as in Section 4.2. In addition, the configuration of the disruptions in the experiments were the same as Section 4.2.

5.3.3 Results

The (upper) graphs in Figs. 8(a), 9(a), 10(a) and 11(a) show the experimental results on both packet loss rate associated with connection-establishment requests and the connection-establishment timeout. The x- and y-axes show the timeout and the elapsed time, respectively. The solid lines and dashed lines show the measured connection-establishment timeout and the average timeout of each segment period of packet loss configurations. In Figs. 8, 9, 10 and 11, while the experimental configurations are the same, the measured results are different. Since we have introduced real Raspberry Pi-based terminals, the measurement values were different due to the inter-individual variation of them. In these experiments, we have displayed individual trials in Figs. 8, 9, 10 and 11 to confirm the effects on varying packet loss rate environments. The (lower) graphs in Figs. 8(b), 9(b), 10(b) and 11(b) show the number of failed connection-establishment procedures in each segment period. The x- and y-axes show the number of failed connection-establishment procedures and the elapsed time. In this experiments, the number of failed connection-establishment procedures shows the total number of failures in each segment period. For example, from the 0-5 minutes segment in Fig. 8(b), two connection-establishment failures occur in 0% control packet loss environments. From Figs. 8 to 11, our proposed method can adapt the connection-establishment timeout to match the varying number of failed connection-establishment procedures. In addition, the connection-establishment timeout increased as the number of failed connection-establishment procedures increased.

5.4 Experiments on the success rate of connection-establishment procedures

5.4.1 Overview and configuration

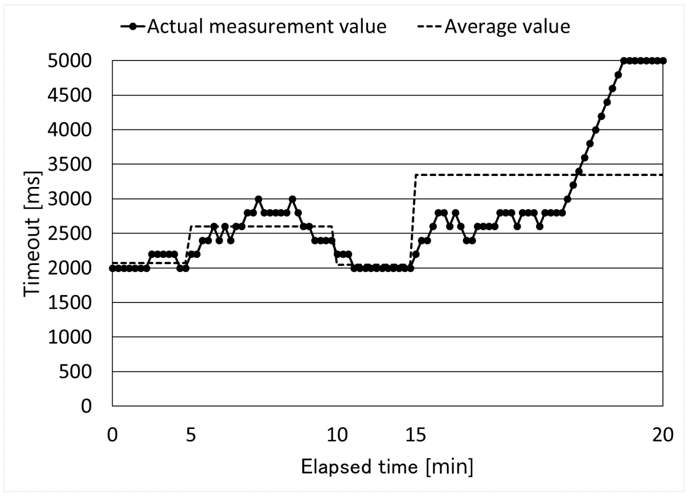
In order to confirm the effects on the connection-establishment timeout, we have conducted experiments on the success rate of connection-establishment procedures. We have prepared the following three types of methods: the 2-second fixed timeout method, the 5-second fixed timeout method, and our proposed method.

In the experiments, we placed two Raspberry Pi 3 terminals adjacent to each other. The duration of the experiments were 30 minutes at every rate of the connection-establishment request packet configuration. We prepared the varied rate connection-establishment request packet configurations as 0, 20, 50, and 90 percent. The hardware specifications of the Raspberry Pi 3 were the same as shown in Section 4.2. In addition, the configuration of the disruption of the established connection was the same as shown in Section 4.2.

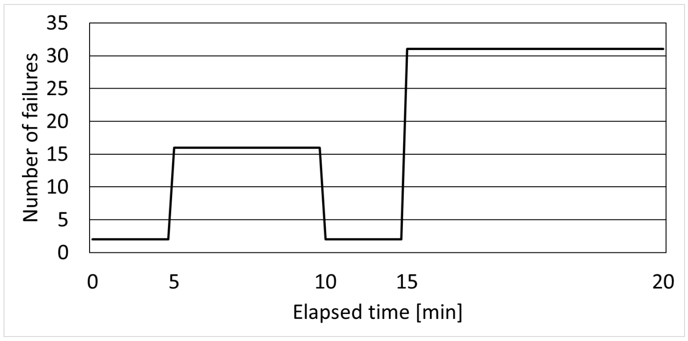
5.4.2 Results

Fig. 12 shows the results of experiments for the connection establishment success rate. Fig. 12 shows the success rate of the connection-establishment procedure at each packet loss rate configuration of the connection-establishment. The x- and y-axes show the success rate of the connection-establishment procedures and the packet loss rate of the connection-establishment requests.

As shown in Fig. 12, the success rate of connection-establishment procedures at the 0% packet loss rate of connection-establishment request was not 100% but 90% with the 2-second fixed time-

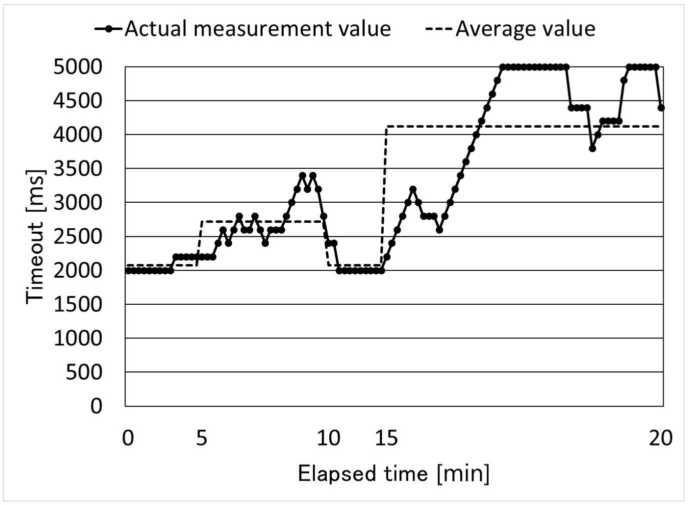


(a) Timeout

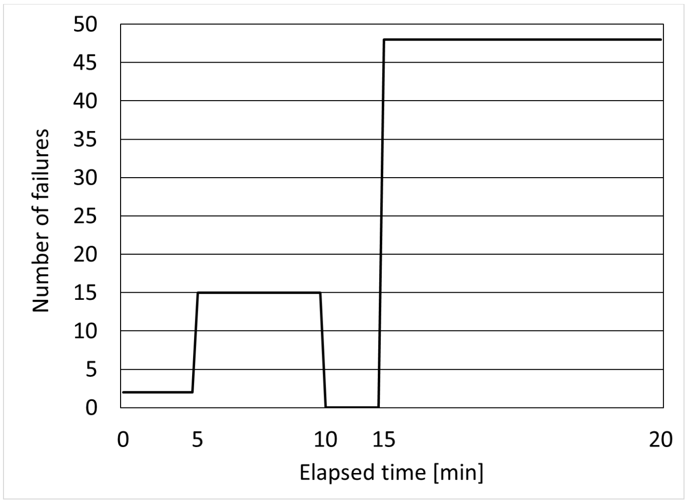


(b) Number of failures

Figure 8: Results of experiments for adaptability(1)

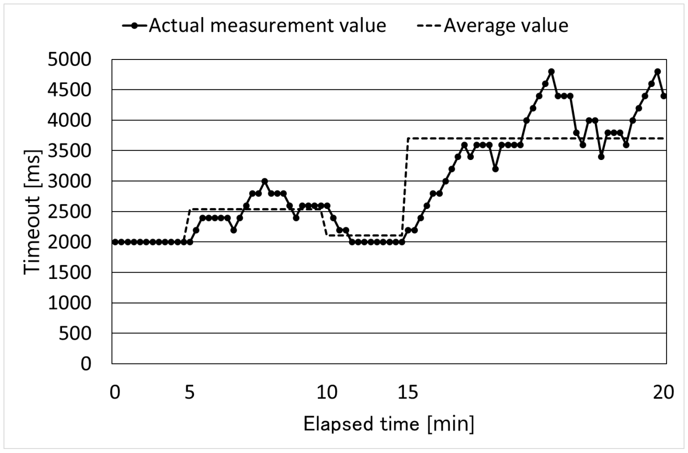


(a) Timeout

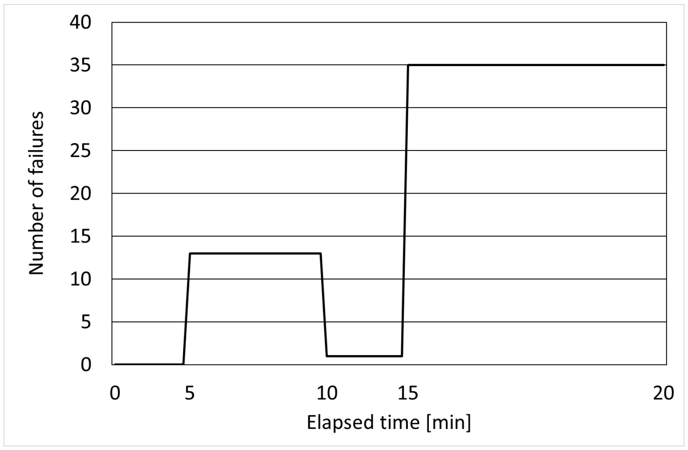


(b) Number of failures

Figure 9: Results of experiments for adaptability(2)

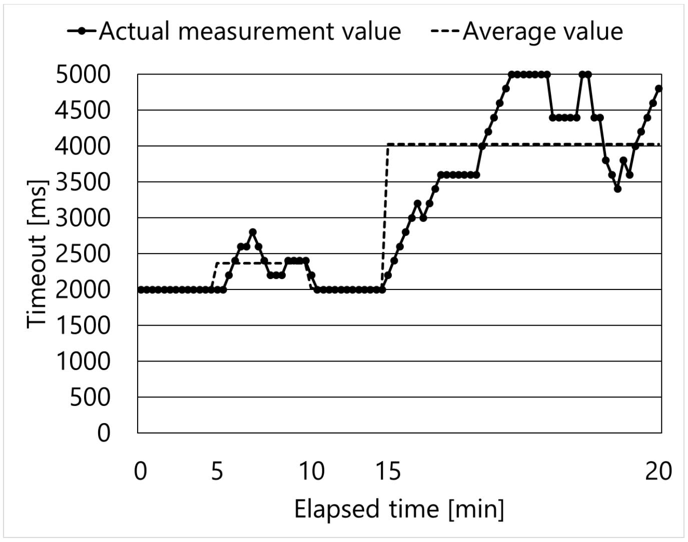


(a) Timeout

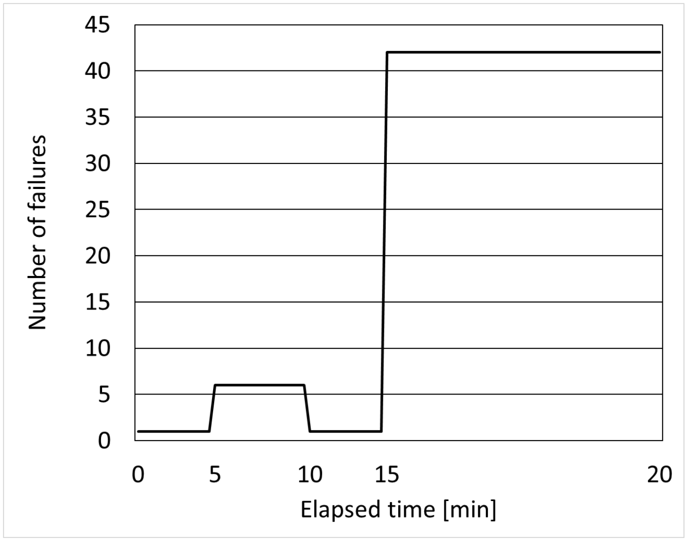


(b) Number of failures

Figure 10: Results of experiments for adaptability(3)



(a) Timeout



(b) Number of failures

Figure 11: Results of experiments for adaptability(4)

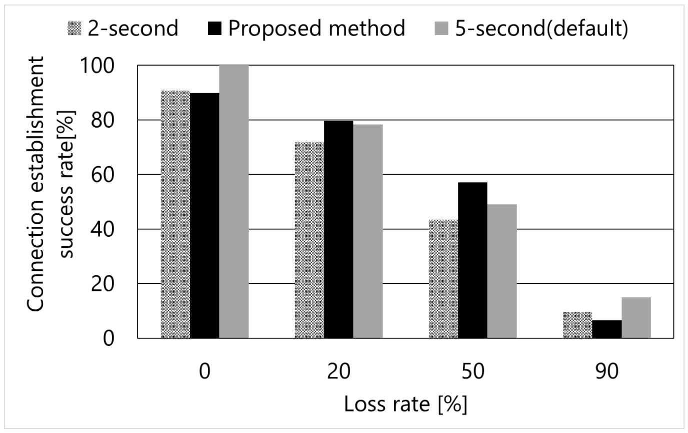


Figure 12: Results of experiments for connection establishment success rate

out method and our proposed method. At the 20% and 50% packet loss rate of the connection-establishment request, the success rate of our proposed method was higher than the 2-second fixed and the 5-second fixed timeout methods. At a 90% packet loss rate of connection-establishment requests, the success rate of our proposed method was slightly lower than that of the 2-second fixed timeout method and was lower than the success rate of the 5-second fixed timeout method. In our proposed method, the connection-establishment timeout value increases as the connection-establishment procedure fails. As a result, the frequency of the connection-establishment procedure for our proposed method became less than that of the 2-second fixed timeout method and became more than that of the 5-second fixed method. The gap in the success rate between the 2-second fixed timeout method and our proposed method was caused by a small number of outlier experiments.

Paper [7] shows that while throughput does not decrease much in almost 10% packet loss environments in Wi-Fi, throughput decreases 50% in 20% packet loss environments. Therefore, the 20-90% packet loss environments are higher packet loss scenarios for Bluetooth. Furthermore, since Bluetooth has the adaptive frequency hopping (AFH) mechanism [5], Bluetooth-enabled terminals will invoke the AFH mechanism to counter higher packet loss scenario in real situations.

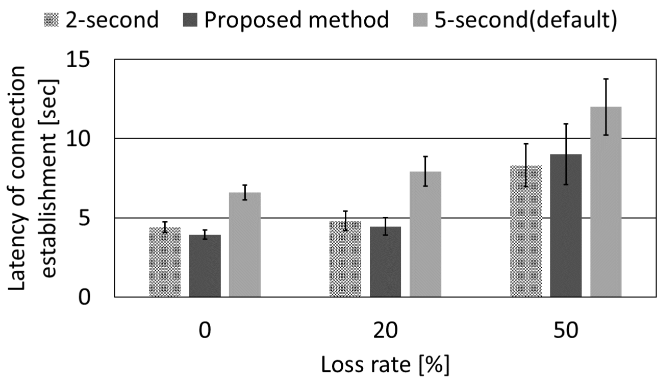
5.5 Experiments on the latency of connection-establishment

5.5.1 Overview and configuration

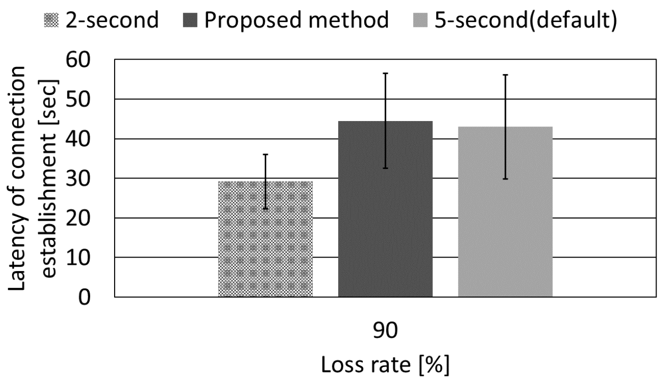
In order to confirm the effect on the latency of connection-establishment of our proposed method, we have conducted experiments on the latency of connection-establishment. In the experiments, we define the connection-establishment latency as the waiting time between disconnection and connection re-establishment. In the experiments, we placed two Raspberry Pi 3 terminals adjacent to each other. We utilized the 2-second fixed timeout method, the 5-second fixed timeout method, and our proposed method. The duration of the experiments were 30 minutes at every rate of the connection-establishment request packet configuration in each method. We set the varied rate connection-establishment request packet configurations at 0, 20, 50, and 90 percent. The hardware specifications of the Raspberry Pi 3s were the same as shown in Section 4.2. In addition, the configuration of the disruption of the established connection was the same as shown in Section 4.2.

5.5.2 Result

Fig. 13 shows the latency of connection-establishment at each packet loss rate configuration of the connection-establishment. The x- and y-axes show the latency of connection-establishment and the packet loss rate of the connection-establishment requests. As shown in Fig. 13(a), at the 0%, 20% and 50% packet loss rate of the connection-establishment request, the latency of our proposed method was the same as the 2-second fixed timeout method. The results show that our proposed method



(a) Packet loss rate 0 - 50%



(b) Packet loss rate 90%

Figure 13: Results of experiments for the latency of connection establishment

can improve the performance of the connection-establishment procedures compared to that of the 5-second fixed timeout method. As shown in Fig. 13(b), at the 90% packet loss rate of connection-establishment requests, our proposed method's latency was longer than that of the 2-second fixed timeout method.

6 Discussion

From Figs. 8 to 13, we discuss the effect of our proposed method. As we described in Sections 5.1 and 5.2, the purpose and targets of our proposed system is to realize fast connection-establishment for Bluetooth MANETs with control packet loss environments. From Fig. 13, the latency of connection-establishment is shorter or almost equal to the 2-second fixed timeout method in constant control packet loss rate environments between 0 to 50%. From Figs. 8 to 11, our proposed method displays adaptability to varying packet loss environments. From Fig. 12, the connection-establishment success rate of our proposed method, however, was almost 10% inferior to the 5-second fixed timeout method. For the reason, that a "race-condition" can occur in 0% packet loss environments. This degradation of the connection-establishment success rate indicates that terminals perform excessive multiple connection-establishment processes. Thus, the terminals may consume more power. Furthermore, from Figs. 12 and 13, both the connection establishment success rate and the latency of connection establishment were inferior to 2-second and 5-second fixed timeout methods. The reason for these results is that our proposed method causes frequent measurement procedures in 90% packet loss environments. Moreover, our proposed method gradually increases the timeout value when the connection-establishment procedures failed. As future work, we have to develop countermeasures to these drawbacks.

7 Conclusion

In this paper, we have proposed a connection-establishment timeout adjustment method to increase the success rate of the connection establishment in BLE-broadcast assisted fast connection establishment for Bluetooth MANETs. We also have confirmed the effectiveness of our proposed method through real terminal-based experiments using Raspberry Pi.

From preliminary experiments, we have confirmed that the adjustment of the connection-establishment timeout is effective. Furthermore, we have found a configurable range of the connection-establishment timeout. In addition, we also confirmed that our proposed method with adaptable mechanisms could be effective at the 20% and 50% connection-establishment packet loss rate. We also have conducted experiments on the latency of connection-establishment and discussed the effectiveness of our proposed method in various aspects. As a result, our proposed method can reduce the latency of connection-establishment. At the same time, we found that our proposed method has drawbacks in severe packet loss environments such as 90% packet loss. As future work, we have to develop countermeasures to these drawbacks.

Acknowledgment

This research is supported by Ministry of Internal Affairs and Communications, Japan, Strategic Information and Communications R&D Promotion Programme (SCOPE), JSPS KAKENHI Grant Number (C) (No. 17K00130, and 17K00131).

References

[1] IEEE standard for information technology– local and metropolitan area networks– specific requirements– part 15.1a: Wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPAN). *IEEE Std 802.15.1-2005 (Revision of IEEE Std 802.15.1-2002)*, pages 1–700, June 2005.

[2] IEEE standard for local and metropolitan area networks: Overview and architecture. *IEEE Std 802-2014 (Revision to IEEE Std 802-2001)*, pages 1–74, June 2014.

[3] Stefano Basagni, Marco Conti, Silvia Giordano, and Ivan Stojmenović. Mobile Ad Hoc Networking, John Wiley & Sons., Aug. 2004.

[4] Bluetooth SIG. Specification of the Bluetooth system – Covered Core Package version: 4.0 –. Dec. 2013. https://www.bluetooth.com/ja-jp/specifications/bluetooth-core-specification/legacy-specifications (2018-01-21).

[5] Bluetooth SIG. Specification of the Bluetooth system – Covered Core Package version: 4.1 –. Dec. 2013. https://www.bluetooth.com/ja-jp/specifications/bluetooth-core-specification/legacy-specifications (2018-01-21).

[6] C. Siva Ram Murthy and B. S. Manoj. *Ad Hoc Wireless Networks – Architectures and Protocols –*. Prentice Hall, 2004.

[7] Hiroyasu OBATA and Kohei IZUI and Ryo HAMAMOTO and Chisa TAKANO and Kenji ISHIDA. Invited Lecture A Study on Experimental Evaluation and Modeling of Transmission Rate over Multi-Rate Wireless LAN. *Technical Report of IEICE in Japanese*, 116(45):53–58, 2016.

[8] Nobuhiro Kajikawa, Yuya Minami, Eitaro Kohno, and Yoshiaki Kakuda. On availability and energy consumption of the fast connection establishment method by using Bluetooth Classic and Bluetooth Low Energy. In *Proc. Fourth International Symposium on Computing and Networking (CANDAR 2016), 9th International Workshop on Autonomous Self-Organizing Networks (ASON 2016)*, pages 286–290, Higashi-Hiroshima, Japan, Nov. 2016.

[9] Yuya Minami, Nobuhiro Kajikawa, Ryohei Saka, Yuma Nakao, Eitaro Kohno, and Yoshiaki Kakuda. Arbitration-based Deadlock Mitigation Mechanism for Fast Connection Establishment in Autonomous Self-organized Bluetooth MANETs. In *Proc. 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovations, at The 17th International Workshop on Assurance in Distributed Systems and Networks (ADSN 2018)*, pages 1611–1616, Guangzhou, China, Oct. 2018.

[10] Yuya Minami, Yuya Kitaura, Eitaro Kohno, Shinji Inoue, Tomoyuki Ohta, and Yoshiaki Kakuda. Implementation and evaluation of dual-purpose normal and disaster situations system based on delay and disruption tolerant Bluetooth MANETs. *International Journal of Communications, Network and System Sciences*, 8(9):342–357, Sep. 2015.

[11] Masahiro Nishi, Koichi Shin, Eitaro Kohno, Shinji Inoue, Tomoyuki Ohta, Kenji Ishida, Eiji Utsunomiya, and Yoshiaki Kakuda. A proposal of grass-roots information delivery systems for protecting oneself from land disasters. *Technical Report of IEICE*, 116(250):29–34, Oct. 2016. (in Japanese).

[12] Temma Ohtani, Nobuhiro Kajikawa, Yuma Nakao, Eitaro Kohno, and Yoshiaki Kakuda. Implementation and verification of interoperable bluetooth manet-enabled communication system with between Android- and Raspberry Pi-based terminals. The 2018 IEICE General Conference, March 2018. (in Japanese).

[13] Temma Ohtani, Eitaro Kohno, and Yoshiaki Kakuda. On relationship between timeout and latency of connection re-establishment for control packetloss scenario in Bluetooth MANETs. In *Proc. 2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW), at the 11th International Workshop on Autonomous Self-Organizing Networks (ASON)*, pages 42–46, November 2018.

[14] Koji Taketa and Yoshiaki Kakuda. A method for effective discovery and connection in Bluetooth MANET consisting of Android terminals. In *Proc. 8th International Conference on Broadband Communications and Biomedical Applications (IB2COM 2013)*, pages 78–83, Guilin, China, Dec. 2013.

[15] Chai-Keong Toh. Ad hoc mobile wireless networks, Prentice Hall, 2002.

[16] Amin Vahdat and Devid Becker. Epidemic routing for partially connected ad hoc networks. Duke Tech. Report. CS-2000-06, July 2000.