Sub-GHz Band Communication Bridge Connecting TCP/IP Network

Satoshi Kodama[1], Rei Nakagawa[2,*]

[1]Research Institute for Science and Technology, Tokyo University of Science, 2641 Yamazaki, Noda, Chiba Prefecture, Japan.

[2]Graduate School of Informatics and Engineering, University of Electro-Communication, 1-5-1 Chofugaoka, Chofu, Tokyo, Japan.

Abstract

Recently, low power wide area (LPWA) communication devices have attracted attention in the construction of wide area sensor networks. A type of LPWA device called a sub-GHz band communication device allows a 1-hop wide area communication system to have improved power consumption. However, a sub-GHz band communication system is not scalable due to the lack of compatibility with heterogeneous communication devices, such as WiFi and LTE. In this paper, we propose a sub-GHz band communication bridge that builds a cross-over communication system connecting a conventional TCP/IP network and LPWA network. The communication bridge enables scalable TCP/IP network services (e.g., HTTP) by transmitting an ethernet frame over an LPWA network. Through experiments using IM920, a dedicated device for 920 MHz band communication, we evaluated the performance and behavior of communication through TCP applications over a TCP/IP network and LPWA network connected by the proposed communication bridge.

Keywords: IoT, LPWA, Heterogeneous Networks, Software-Defined Network, Communication Bridge.

Received on 05 July 2019, accepted on 18 July 2019, published on 29 July 2019

Copyright © 2019 Satoshi Kodama and Rei Nakagawa, licensed to EAI. This is an open access article distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/3.0/), which permits unlimited use, distribution and reproduction in any medium so long as the original work is properly cited.

doi: 10.4108/eai.13-7-2018.163921

*Corresponding author. Email: J6316627@gmail.com

1. Introduction

The Internet of Things (IoT) has led to diverse communication requirements. For example, personal applications involving IoT have become common, such as the management of physical conditions by uploading data to the cloud using a wearable device. In addition, network infrastructure with IoT allows for smart cities, which automatically collect various consumption states, such as electric power, gas, and water [1]. Thus, with the advent of the IoT era, a data aggregation system for various sensor devices is essential.

To enable wireless sensor networks (WSNs) with wide area coverage and low cost devices, low power wide area (LPWA) communication devices have emerged in the past several years and have experienced rapid growth [2]. A sub-GHz band communication device, one of the most common LPWA devices, uses a frequency band lower than 1 GHz for communication, and provides wide-range and low-power communication. Because sub-GHz band communication meets IoT requirements, studies have developed WSNs using sub-GHz band communication devices [3]. Although communication standards have emerged for sub-GHz band communication, each communication standard requires a communication device with a dedicated data transfer method and lacks compatibility with heterogeneous communication devices, such as WiFi and LTE [8].

The aim of this study is to achieve cross-over communication that connects an LPWA network and heterogeneous network and makes it possible to use the benefits of both networks. In this paper, we propose a sub-GHz band communication bridge with a TCP/IP network. Specifically, the proposed communication bridge provides software-defined architecture that enables seamless communication over sub-GHz band communication and

TCP/IP communication. By using the communication bridge, a client can use a conventional TCP application over the bridge that connects an LPWA network and TCP/IP network.

The design of the communication bridge is based on a packet processing factory, which has been proposed in previous studies [4], [5] (see Section 2.2) to leverage the programmability of a software-defined network (SDN). We implemented the communication bridge using IM920, a dedicated device for 920 MHz band communication (see Section 2.1), to connect a TCP/IP network (see Sections 3 and 4) and sub-GHz band network. Through experiments using IM920, we evaluated the performance of communication in terms of the packet loss rate and round trip time (RTT). In addition, we evaluated the communication behavior using an HTTP application over a TCP/IP and sub-GHz network connected by the proposed communication bridge (see Section 5). Conclusions of the study are provided in Section 6.

2. Related Work

In this section, we present a description of IM920 as a sub-GHz band communication device. We then describe the packet processing factory, which is the basis of the design of the communication bridge.

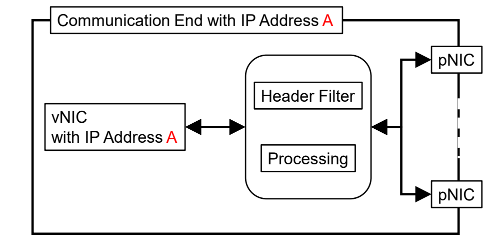
2.1. IM920

IM920 is an LPWA communication device developed by Interplan Co. Ltd. that enables many-to-many 920 MHz band communication [6]. Specifically, IM920 is a long-range communication device with a small bitstream encoded to American Standard Code for Information Interchange (ASCII) [7]. Several dedicated commands perform the communication control of IM920 (e.g., power consumption setting, data transmission) via a microcontroller unit capable of USB connection. The dedicated command area is composed of a command portion (4 bytes) and bitstream portion (variable length up to 64 bytes). For instance, an IM920 sender inputs “TXDA ABCD,” and the IM920 receiver outputs “ABCD.” The previous command is equivalent to "TXDA 0x410x420x430x44" expressed in an ASCII hex bitstream. In addition, IM920 has two types of communication ranges: a fast mode (up to 400 m) and long-range mode (up to 7 km). Because the fast mode has a shorter RTT than the long-range mode, the throughput of the former is higher than that of the latter. In this study, we use the fast mode for the evaluation of the proposed communication bridge.

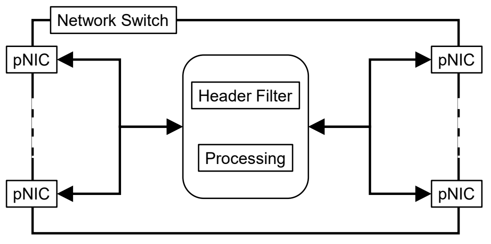
Several studies have used IM920 to construct a WSN [9]-[12]. In a previous study [11], we confirmed the effectiveness of using the proposed communication bridge to send sensor data and text messages collected from a WSN to a UDP/IP server. In this paper, we examine the utility for more general TCP applications. In addition, a proxy server device has been developed to connect heterogeneous networks [13], where sensor data from a WSN are accumulated to be transmitted over a conventional 3G network. Thus, the development of LPWA devices that are compatible with heterogeneous networks is an important topic.

2.2. Packet processing factory

The design of the proposed communication bridge is based on the packet processing factory proposed in previous studies [4], [5]. The packet processing factory handles communication traffic using communication sockets in the C programming language. In addition, it operates on a communication host to capture, modify, and release traffic from the sockets on equipped communication devices. The packet processing factory has two simple architectures: a communication end and a network switch. Figure 1 presents the two architectures of the software bridge.



(a) Communication end of packet processing factory



(b) Network switch of packet processing factory

Figure 1. Two simple architectures of the packet processing factory.

Figure 1 (a) displays the communication end of the packet processing factory. In Figure 1 (a), virtual network interface cards (vNIC) are the SDN interface for the communication end. vNIC receives/sends traffic from/to the host application by assigning the communication end’s IP address. The traffic is processed according to the header information (e.g., ethernet header, IP header, TCP header in TCP/IP ethernet frame) between the sockets on the vNIC and physical network interface cards (pNICs). Figure 1 (b) illustrates the network switch of the packet processing factory. In Figure 1 (b), the relay traffic is processed between the sockets on the pNICs and is a process of the communication end architecture.

3. Sub-GHz Band Communication Bridge Connecting TCP/IP Network

3.1. Overview

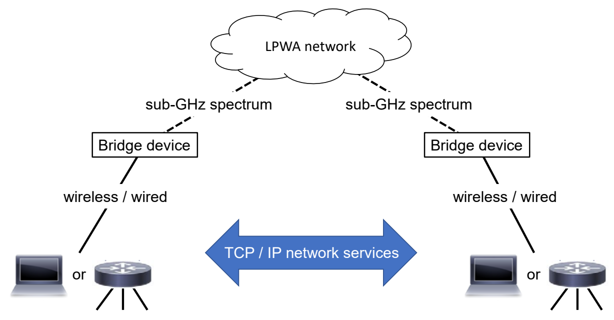


Figure 2. Overview of network system with sub-GHz band communication bridge.

In this section, we introduce the concept of a network system using the proposed communication bridge. The network system consists of a TCP/IP network and LPWA network connected by the communication bridge. The communication bridge splits traffic from the TCP/IP network into single/multiple traffic for the LPWA network. In addition, the traffic from the LPWA network is combined with an ethernet frame by the communication bridge. In this way, the TCP/IP communication ends can use conventional network applications via an LPWA network without any changes to the required equipment.

Figure 2 presents an overview of the network system with the sub-GHz band communication bridge connecting the TCP/IP network. The communication bridge devices displayed in Figure 2 equip two types of network interface cards: TCP/IP network interface cards (NICs; e.g., wired ethernet card, WiFi base unit) and sub-GHz band communication devices. TCP/IP NICs handle the ethernet frame in TCP/IP, while sub-GHz band communication devices handle the bitstream in the LPWA network. This network system provides a physical layer in which both TCP/IP NICs and sub-GHz band communication devices operate in a compatible way to deliver data from the upper TCP/IP application layers on the communication ends. For example, when a PC downloads HTML code/page(s) on a server through the HTTP protocol, HTTP traffic composed of single/multiple ethernet frames is delivered between PCs. The relay bridge devices split an ethernet frame into single/multiple LPWA bitstreams for LPWA outbound traffic and combine single/multiple LPWA bitstreams into an ethernet frame for LPWA inbound traffic.

3.1. Packet processing in a bridge device

Because a bridge device must split/combine traffic from TCP/IP NICs or sub-GHz band communication devices, we apply the concept of the packet processing factory to the bridge devices. The packet processing factory represents the traffic split/combine process through communication between the sockets on TCP/IP NICs and sub-GHz band communication devices. Figure 3 displays the traffic flow in a bridge device. The bridge device connects the TCP/IP NIC and sub-GHz sender device and receiver device. The sub-GHz sender device communicates with another bridge device's receiver device for outbound communication from TCP/IP NIC. Similarly, the receiver sub-GHz device handles inbound traffic to the TCP/IP NIC.

Packet processing takes place between the TCP/IP NIC and sender/receiver sub-GHz device. For outbound traffic from the TCP/IP NIC, the bridge device first splits an ethernet frame into multiple bitstreams for sub-GHz communication. Second, each bitstream is assigned control bits of fixed length for data combination. The control bits represent the sequence number in the bitstream composing the ethernet frame over sub-GHz band communication. The bridge device finally transmits each bitstream in order according to the index in the control bits through the sub-GHz sender device. For inbound traffic from the sub-GHz receiver device, the bridge device receives multiple bitstreams with control bits. If the index of the bitstream is final index the bridge device removes the control bits from each bitstream and combines multiple bitstreams into an ethernet frame according to the sequence number in the control bits of each bitstream. The bridge device finally transmits the ethernet through the TCP/IP NIC.

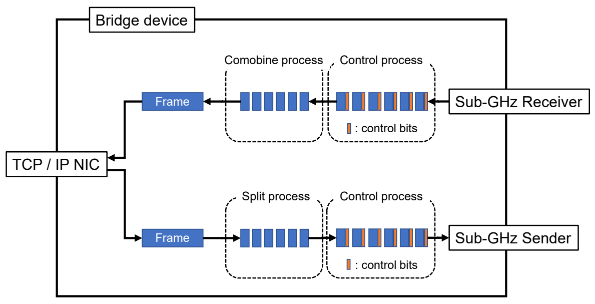


Figure 3. Traffic flow in the communication bridge.

4. Implementation

We implemented a network system with the proposed communication bridge described in Section 3 by developing dedicated C software on Linux using real devices. Figure 4 illustrates the implementation of the network system with the proposed communication bridge using IM920 with two PCs. Each PC equips a sender IM920 and receiver IM920 for sub-GHz band communication, and each IM920 uses the fast mode as the communication type presented in Figure 4. This implementation enables a 920-MHz communication bridge connecting a TCP/IP network between two PCs through each equipped vNIC. To install the combine and split processes in the two PCs, we used communication end architecture with the packet processing factory.

Figure 5 illustrates the internal work on a PC for the implementation of the communication bridge using IM920. First, the transmission of outbound traffic from vNIC (Multi-thread process 2 bound with the communication socket of IM920 Sender in Figure 5) and the reception of inbound traffic to vNIC (Multi-thread process 1 bound with the communication socket of IM920 Receiver in Figure 5) operate independently using multi-threaded programming. Then, to deliver the bitstream encoded by ASCII over IM920 communication, each multi-threaded processes performs base64 encoding/decoding for the split/combine process. Therefore, the size of an encoded ethernet frame bound for the IM920 is 4/3 times the size of the original ethernet frame according to the base64 encoding method [14]. In addition, the size of a bitstream, which is the unit of communication traffic between the IM920s, consists of a data payload of 60 bytes and control bits of 4 bytes in the form of a sequence number field.

Table 1. Specifications of PCs for implementation.

	Specifications
OS	CentOS 7 with Linux 3.10.0-693.el7.x86_64
Memory	DDR4 8 Gbytes
CPU	Intel Core i7-8550U

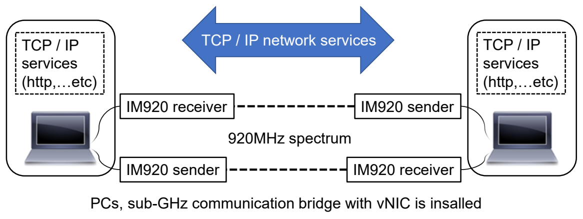

TCP / IP services (http,…etc)
TCP / IP network services
TCP / IP services (http,…etc)
IM920 receiver
IM920 sender
920MHz spectrum
IM920 sender
IM920 receiver
PCs, sub-GHz communication bridge with vNIC is insalled

Figure 4. Implementation of proposed communication bridge using IM920 with two PCs.

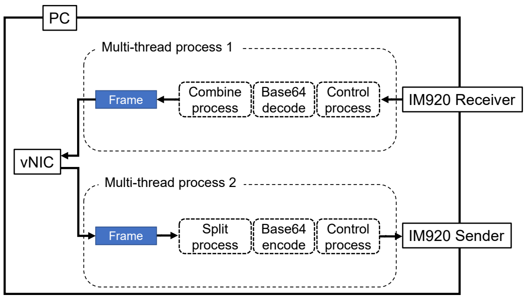


Figure 5. Internal processes on a PC for the implementation of the communication bridge using IM920.

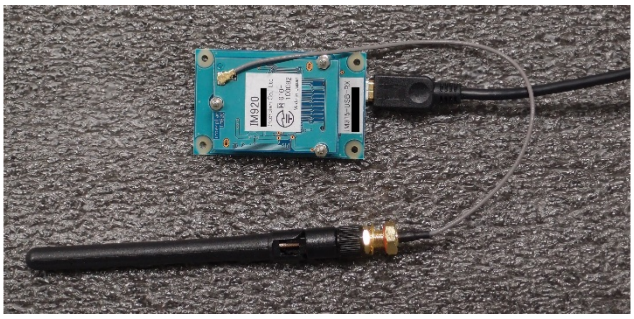


Figure 6. IM920 used in implementation.

5. Evaluation

In this section, we evaluate the communication performance of the implementation described in Section 4. Then, we evaluate the behavior of the communication through conventional HTTP download using the implementation. It should be noted that the distance between PCs was fixed at a sufficiently small length for a small experiment.

5.1. Evaluation of communication performance in terms of packet loss rate and RTT

In this subsection, we evaluate the communication performance in terms of the packet loss rate and RTT upon executing the ping command 1,000 times on a Linux machine. It is assumed that the time interval between each ping command is sufficiently large.

Table 2 presents a comparison of the packet loss rate according to the size of the ping packet (which equals the size of the ethernet frame minus 14 bytes) in the range of 64 bytes to 1,500 bytes. It can be seen that the packet loss rate increases with an increase in the size of the ping packet. Because a large ping packet has a large amount of communication traffic between IM920s and increases the probability of bit error, it can cause incomplete data communication due to bit error and can increase the packet loss rate.

Table 2. Comparison of packet loss rate according to the size of the ethernet frame in the range of 64 bytes to 1,500 bytes.

Size of ping packet (bytes)	Packet loss rate (%)
64	9
128	6
256	12
512	9
1,024	21
1,500	27

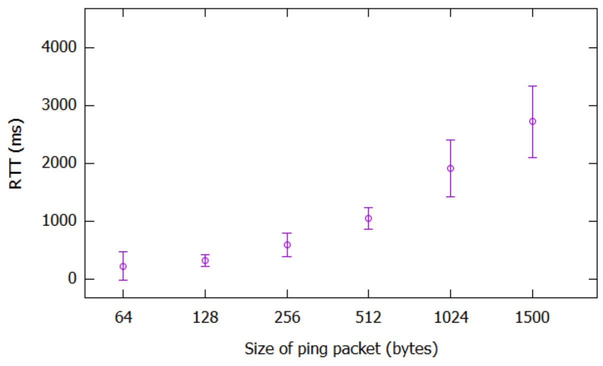


Figure 7. Comparison of average round trip time (error bars representing standard deviation) according to the size of the ethernet frame in the range of 64 bytes to 1,514 bytes.

Figure 7 presents a comparison of the average RTT with error bars from the minimum RTT to the maximum RTT according to the size of the ethernet frame. As in the evaluation of the packet loss rate, the RTT increases with an increase in packet size. However, the standard deviation of the RTT is unstable regardless of the packet size.

5.2. Evaluation of communication behavior through conventional HTTP download

In this subsection, we provide an evaluation of the communication behavior through conventional HTTP download using the proposed implementation. We prepared a scenario in which a PC downloaded a file with text size in the range of 1 kb to 10 kb from an Apache HTTP server on another PC using the curl command on Linux.

Table 3. Comparison of average HTTP download time according to the size of the download in the range of 1,000 bytes to 10,000 bytes in 100 download trials.

Size of download (bytes)	Connection time (s)	Download time (s)
1,000	0.27	2.07
2,000	0.35	4
3,000	0.54	6.57
4,000	0.16	9.74
5,000	0.18	14.83
6,000	0.2	30.55
7,000	0.18	31.81
8,000	0.18	67.17
9,000	0.26	110.34
10,000	0.23	125.43

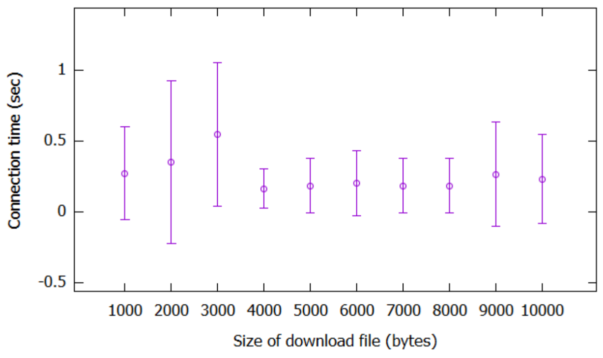


Figure 8. Comparison of average TCP connection time (error bars representing standard deviation) according to the size of the download in the range of 1,000 bytes to 10,000 bytes.

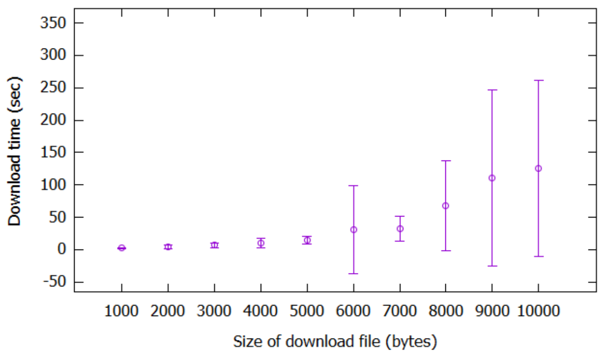
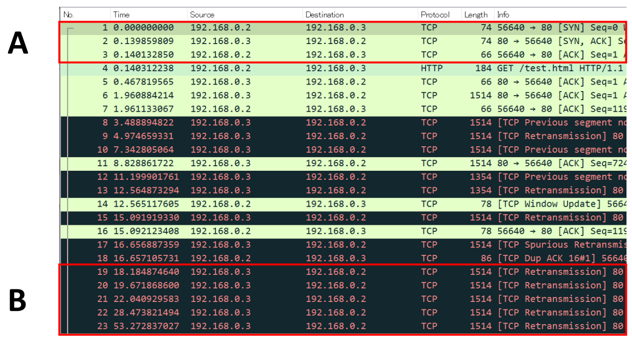


Figure 9. Comparison of average HTTP download time (error bars representing standard deviation) according to the size of the download in the range of 1,000 bytes to 10,000 bytes.



```
No.  Time          Source       Destination  Protocol Length Info
   1 0.000000000   192.168.0.2  192.168.0.3  TCP        74 56640 → 80 [SYN] Seq=0
   2 0.139859809   192.168.0.3  192.168.0.2  TCP        74 80 → 56640 [SYN, ACK] S
   3 0.140132850   192.168.0.2  192.168.0.3  TCP        66 56640 → 80 [ACK] Seq=1
   4 0.140312238   192.168.0.2  192.168.0.3  HTTP      184 GET /test.html HTTP/1.1
   5 0.467819565   192.168.0.3  192.168.0.2  TCP        66 80 → 56640 [ACK] Seq=1
   6 1.960884214   192.168.0.3  192.168.0.2  TCP      1514 80 → 56640 [ACK] Seq=1
   7 1.961133067   192.168.0.2  192.168.0.3  TCP        66 56640 → 80 [ACK] Seq=11
   8 3.488894822   192.168.0.3  192.168.0.2  TCP      1514 [TCP Previous segment n
   9 4.974659331   192.168.0.3  192.168.0.2  TCP      1514 [TCP Retransmission] 80
  10 7.342805064   192.168.0.3  192.168.0.2  TCP      1514 [TCP Previous segment n
  11 8.828861722   192.168.0.3  192.168.0.2  TCP      1514 80 → 56640 [ACK] Seq=72
  12 11.199901761  192.168.0.3  192.168.0.2  TCP      1354 [TCP Previous segment n
  13 12.564873294  192.168.0.3  192.168.0.2  TCP      1354 [TCP Retransmission] 80
  14 12.565117605  192.168.0.2  192.168.0.3  TCP        78 [TCP Window Update] 566
  15 15.091919330  192.168.0.3  192.168.0.2  TCP      1514 [TCP Retransmission] 80
  16 15.092123408  192.168.0.2  192.168.0.3  TCP        78 56640 → 80 [ACK] Seq=11
  17 16.656887359  192.168.0.3  192.168.0.2  TCP      1514 [TCP Spurious Retransmi
  18 16.657105731  192.168.0.2  192.168.0.3  TCP        86 [TCP Dup ACK 16#1] 5664
  19 18.184874640  192.168.0.3  192.168.0.2  TCP      1514 [TCP Retransmission] 80
  20 19.671868600  192.168.0.3  192.168.0.2  TCP      1514 [TCP Retransmission] 80
  21 22.040929583  192.168.0.3  192.168.0.2  TCP      1514 [TCP Retransmission] 80
  22 28.473821494  192.168.0.3  192.168.0.2  TCP      1514 [TCP Retransmission] 80
  23 53.272837027  192.168.0.3  192.168.0.2  TCP      1514 [TCP Retransmission] 80
```

Figure 10. Behavior of TCP communication through file download capture by Wireshark.

Table 3 presents a comparison of the average TCP connection times and HTTP download times according to the size of the download file in the range of 1,000 bytes to 10,000 bytes in 100 download trials. The TCP connection time represents the time length from the start of the connection on TCP to the end of the three-way handshake. The HTTP download time represents the time length from the start of the connection on TCP to the end of the download. Figures 8 and 9 also present a comparison of the average TCP connection time and HTTP download time, respectively, with error bars representing the standard deviation.

Table 3 and Figure 8 reveal that the connection time is generally unstable. Because the traffic in the TCP connection is constant regardless of the download size, we consider that the unstable results are affected by TCP retransmission. However, the results are relatively stable in the range of 4,000 bytes to 8,000 bytes. We consider that the stable results operate without being affected by TCP retransmission. Table 3 and Figure 9 also demonstrate that the download time increases as the download size increases. According to the evaluation of RTT in Section 5.1, we conclude that the download time increases with an increase in traffic between IM920s. In addition, despite the increase in download size, the download time is almost identical for download sizes from 6,000 bytes to 7,000 bytes. We believe that this behavior is also caused by TCP retransmission.

Figure 10 illustrates the behavior of TCP communication through the file download capture by Wireshark. We can see that the TCP three-way handshake is successful (part A in Figure 10). The ideal TCP connection time is thus approximately 0.14 seconds. According to this result, it can be concluded that a very small TCP segment behaves practically because its RTT is acceptable for conventional TCP communication. However, according to Table 3 and Figure 8, when a communication loss occurs between IM920s, a large delay due to TCP retransmission is observed.

In addition, when multiple TCP segments are transmitted to the client by TCP, many full-sized retransmission (1,514 bytes) segments are transmitted by TCP retransmission control (part B in Figure 10). As illustrated in Figure 7, this result reveals that full-sized TCP segments have a very large RTT, which causes redundant retransmission with conventional TCP. The redundant retransmission causes a fatal delay for conventional TCP in a range of 6,000 bytes to 10,000 bytes in Figure 9, and further increases the possibility of bit error between IM920s.

6. Conclusion

In this paper, we propose a communication bridge connecting a TCP/IP network and LPWA network. A network system with the proposed communication bridge allows the TCP/IP communication ends to leverage the merits of the LPWA network without any changes to the equipment. Evaluation of an implementation using IM920 demonstrated that very small TCP segment communication is effective in conventional TCP. In contrast, sequential full-sized (MTU-sized) TCP segment communication is not practical in conventional TCP due to the very large RTT. In this paper, we also observed the need to handle TCP retransmission more effectively.

7. Future Work

In future work, we plan to improve the communication performance in TCP by avoiding redundant transmission of TCP retransmission segment in LPWA device. In addition, we plan to introduce TCP-like control to guarantee the communication quality between LPWA devices.

References

[1] SigFox, https://www.sigfox.com/en/sigfox-story.
[2] LoRa Alliance, "LPWA Technologies Unlock New IoT Market Potential, Machina Research," Nov, 2015.
[3] LoRa Alliance, LoRaWAN What is it? Technical Marketing Workgroup 1.0," 2015.
[4] S. Kodama and R. Nakagawa, "Packet Tagging System Using Adaptive MSS Clamping on TCP," in Proc. of IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 2018.
[5] S. Kodama, R. Nakagawa, N. Takahashi, "TCP Multi-Stream Data Transfer using Multiple Network Interface Cards," in Proc. of 11th Computer Science and Electronic Engineering, 2019
[6] Interplan co. ltd, https://www.interplan.co.jp/.
[7] Interplan co. ltd, "IM920 wireless communication device manual (software)," available at: https://www.interplan.co.jp/support/solution/IM315/manual/IM920_SW_manual.pdf.
[8] R. S. Sinha, Y. Wei, S. Hwang, "A survey on LPWA technology: LoRa and NB-IoT," ICT Express, vol. 3, no. 1, pp.14-21, 2017.
[9] X. Qi, Z. Wen, K. Yu, K. Murata, K. Shibata, T. Sato, "Content-Oriented Disaster Network Utilizing Named Node

Routing and Field Experiment Evaluation," vol. E102D, no. 5, pp.988-997, 2019.

[10] Y. Fukushima et.al., "Application of Electric Double Layer Capacitor and Water Level Sensor to Rice Field Monitoring System," in Proc. of IEEE SENSORS, 2018.

[11] S. Kodama, S. Nagao, "Application of Existing Network Methods in Low Power Long Range Wireless Communication, International Journal of P2P Network Trends and Technology," vol. 10, no. 1, pp.1-9, 2020.

[12] T. Tsunoda and K. Nimura, "Information Harvesting Method by an Energy Harvesting Device with Multiple Microcontroller," in Proc. of IEEE COMPSAC, 2018.

[13] Interplan co. ltd, "IX920-3G," available at: https://www.interplan.co.jp/solution/wireless/ix920-3g.php

[14] S. Josefsson, "The Base16, Base32, and Base64 Data Encodings," RFC 4648, 2006.