LONTAR KOMPUTER VOL. 10, NO. 3 DECEMBER 2019 **p-ISSN 2088-1541**
DOI : 10.24843/LKJITI.2019.v10.i03.p05 **e-ISSN 2541-5832**
Accredited B by RISTEKDIKTI Decree No. 51/E/KPT/2017

Data Transmission Method on a Room Condition Telemonitoring Application (IoT)

Midriem Mirdanies

Research Center for Electrical Power and Mechatronics, Indonesian Institute of Sciences (LIPI)
Komp LIPI Bandung, Jl. Sangkuriang, Gd. 20. Lt. 2, Bandung 40135, Indonesia
midr001@lipi.go.id

Abstract

Data transmission system in IoT applications such as telemonitoring, using cable or wireless, requires a robust data transmission method. In this paper, a room condition telemonitoring application has been made where all data are read from raspberry pi, and sent to a pc server every minute using LAN and Wifi media, then update to a webserver. The data are sent using the method proposed in this paper which using the UDP mechanism with several improvements. Experiments have been done using LAN and Wifi, and it is found that the data can be received perfectly, although there are interferences from communication media. The average processing time on a raspberry pi is 19 ms, while the average processing time on the pc server is 5 us. In addition, the average processing speed of the transmission data from raspberry pi to a pc server until ack is received by raspberry pi using LAN media is 0.04 second, while using Wifi media is 0.08 seconds. Based on the experiments, it was found that the proposed method was successfully implemented in the room condition telemonitoring.

Keywords: *data transmission method, telemonitoring, IoT, raspberry pi, pc server*

1. Introduction

The Government of Indonesia is preparing to the Industry 4.0 [1]. Industry 4.0 consists of several elements such as Internet of Things (IoT), Cloud Computing, big data, smart sensors, and so on. The concept of Industry 4.0 is that all parts from producers to consumers are connected and able to communicate including data processing devices and sensors in it. Data transmission system in the Industry 4.0, especially in Internet of Things (IoT) applications such as telemonitoring, telecontrol, or communication between users or other devices using cable or wireless, require data transmission methods that are efficient, secure, and ensure the data can be received by a receiver.

Telemonitoring system is a system that can monitor a module, objects, an activity, and so on remotely. The telemonitoring system has been implemented in many fields. A lot of paper had been discusses about the telemonitoring system, i.e. Bouchemal et al. [2] which discusses telemonitoring systems that allow medical staff members to access patient data through ubiquitous devices such as laptops, smartphones, or tablets. Ling et al. [3] have discussed a telemonitoring system that allows doctors to check the temperature of patients remotely via XBee communication media. In addition, Mirdanies [4] has also discussed the optimization of telemonitoring systems from the display of robotic camera that have been made before so the processing time is become faster than before using multi-thread method. Many papers that discuss telemonitoring systems, including previously mentioned, have not discussed the detail aspects of the data transmission that is whether the data can be received by a receiver, safe, and not modified by other.

Whereas many papers have discuss about the data transmission, i.e. Elhoseny et al. [5] which discusses the security aspects of medical data transmission by encrypting it, and then hiding it in an image. Whereas Kumar et al. [6] have discussed a reliable communication in IoT Embedded with wireless sensors specifically regarding checking the originality of data and how to correct errors that occurring. Hwang et al. [7] has also discussed the design and implementation of a reliable message transfers based on the MQTT protocol so the messages which sent by the sender can be received quickly by the receiver without a long delay.

Mirdanies et al. [8] also discussed the method of data communication between microcontrollers or microprocessors that can transfer 32-bit integer or float data using only 8 digital pins without using other additional communication media, so it is simple and can be used on minimum system microcontroller. Some existing papers, including that have been previously mentioned, usually only discuss one or a few aspects, and do not discuss all aspects of the data transmission.

In this paper, a room condition telemonitoring application has been created, in which data of temperature, humidity, light condition, and motion detection are read from three sensors using raspberry pi, and send to a pc server using Local Area Network (LAN) or wifi media, then pc server will update the data to the webserver. Data from sensors on raspberry pi are sent to the pc server every minute using the method proposed in this paper, which is using the User Datagram Protocol (UDP) mechanism with several improvements. UDP mechanism is used in this paper because it is faster, simpler and more efficient than Transmission Control Protocol (TCP), however this mechanism requires additional steps in the data transmission method to ensures that data is received, safe, and does not change, so in this paper, there are some improvements, i.e. using the stop and wait arq mechanism, where the format of one block of data (frame) is a integration from four data that are read from three sensors at the same time with the addition of identifiers, encryption and decryption method using a combination of Caesar cipher and Rail Fence cipher methods, and checking the originality of the data using Cyclic Redundancy Check (CRC) method.

2. Research Methods

2.1. The room condition telemonitoring application

The diagram of the room condition telemonitoring application used in this paper can be seen in Figure 1.a and Figure 1.b. On LAN communication media, IP address of the raspberry pi (client) is 192.168.233.31 and the pc server (virtualization) is 192.168.233.112, whereas on the Wifi communication media (access name: LIPI), IP address of the raspberry pi (client) is 172.31.39.234 and pc server (virtualization) is 172.31.39.233. The port is 4000 in both LAN and Wifi media.

The communication media used for communication between pc server (virtualization) and pc web server (windows) is a LAN with the IP address of the pc web server is 192.168.233.77. The sensors used in this paper consist of a temperature and humidity sensor (DHT11), a motion sensor (PIR HC-SR501), and a light sensor (LDR) [9] which can be seen in Figure 2.a, Figure 2.b, and Figure 2.c.

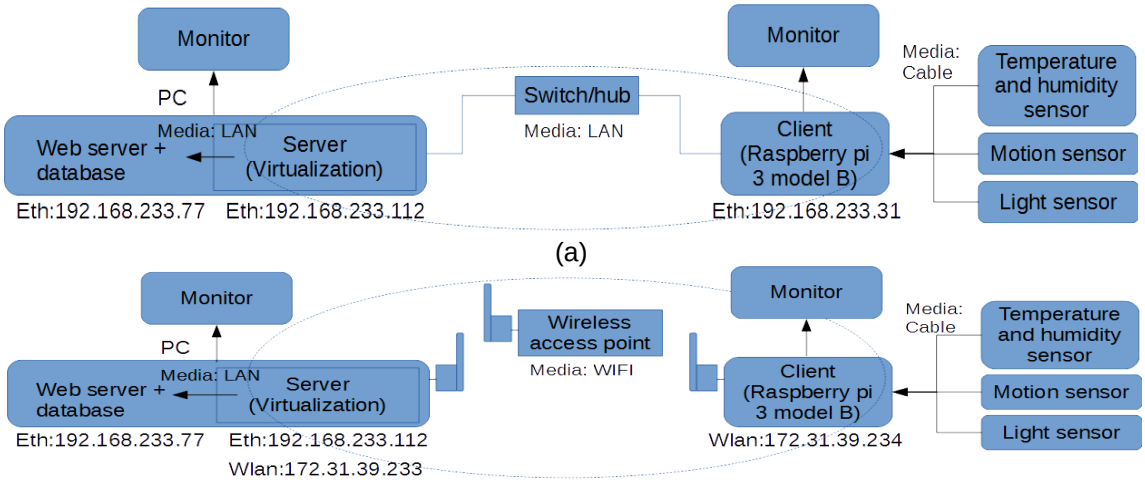


Figure 1. The diagram of the room condition telemonitoring application, where the data communication media between raspberry pi and pc server uses: (a) LAN; (b) Wifi (access name: LIPI)

(a) (b) (c)

Figure 2. The sensors consists of: (a) a temperature and humidity sensor (DHT11); (b) a motion sensor (PIR HC-SR501); (c) a light sensor (LDR)

All sensors are connected to Raspberry Pi 3 model B as shown in Figure 3.

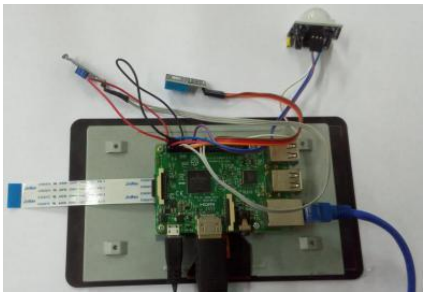


Figure 3. Raspberry pi 3 model B which is connected with three sensors

Figure 4. HP Pavilion 500 PC

Raspberry pi 3 model B is used to read data from sensors, process, and send it to a pc server, where the specifications of the raspberry pi 3 are a 1.2GHz Broadcom BCM2837 64bit CPU, 1GB RAM, 1GB RAM, wireless BCM43438, and LAN 100 Base Ethernet. The pins connection of raspberry pi connected to the three sensors can be seen in Table 1. Whereas the pc server uses hp pavilion 500 pc with core i7-4790 cpu @ 3.6 ghz processor, 8gb ram, realtek pcie lan family controller, and wIfi edimax ac600 wireless lan usb adapter. The display of the pc can be seen in Figure 4.

Table 1. The pins on the Raspberry Pi 3 model B are connected to three sensors

Sensors	Pin (GPIO)	Info
DHT11	7 (7)	Data
	1	3.3 VDC
	39	Ground
PIR HC-SR501	11 (0)	Data
	4	5 VDC
	14	Ground
LDR	5 (9)	Data
	17	3.3 VDC
	20	Ground

Unit of data temperature is ℃, humidity is %, while light conditions is 0 if bright, and 1 if dark, whereas motion detection is 0 if no motion is detected, and 1 if motion is detected.

The PC uses the Windows 8.1 operating system to process web server and database using XAMPP (apache, php, and mysql), while the Oracle VM Virtualbox 6.0 is installed on the PC with the Ubuntu 18.04 operating system, and the resources are 3 cores processor, 3 GB RAM, and connections using 2 bridged adapters, i.e. the LAN Realtek PCIe GBE family controller and Wifi Edimax ac600 wireless LAN USB adapter which used as a server that receives data from the client (raspberry pi), processes, and send it to a webserver (windows 8.1), so it can be accessed by a web browser.

The experiments of data transmission in this paper is limited only between the client (raspberry pi) and pc server (virtualization) using LAN and Wifi media, where the LAN connection are connected via a switch / hub and Wifi connections via an access point as seen in Figure 5.

(a) (b)

Figure 5. Data communication media uses: (a) Switch / hub (TPLink); (b) Access point (TPLink).

2.2. Data transmission method

The data transmission method proposed in this paper uses the UDP mechanism with several improvements. In OSI or TCP / IP layers, UDP is in the transport layer, where in the OSI layers is at the 4th layer while the TCP / IP layers are at the 3rd layer [10]. UDP mechanism is a mechanism that is suitable for real time data transmission applications such as video streaming and so on, but this mechanism requires additional steps in the data transmission method to ensures that data is received, safe, and does not change. Several of the improvements made in this paper are, first, using the stop and wait arq mechanism, second, the format of one data block (frame) is an integration of four data from three sensors at once with the addition of identifiers, third, data encryption and decryption using the combination of the caesar cipher and rail fence cipher methods, and fourth, checking the originality of the data using the crc method. The stop and wait arq mechanism is used to ensure that the data sent can be received by the receiver, the proposing of data integration is to make the transmission time efficient because all data can be sent at once, checking the originality of the data intends to check the originality of the data received so that the data received is the same as the data sent without any changes, while the encryption and decryption data intends to secure the data, so unwanted person cannot know the original data (plaintext) that was sent although the data was successfully hacked during the transmission. The stop and wait arq mechanism can be seen in Figure 6.

The stop and wait mechanism is used in this paper because the method is simple, this method is suitable to use in an application where the data transmission time is not frequent as in the room condition telemonitoring application which transmission time only every minute. In this paper, the waiting time used is 5 seconds, so if in 5 seconds no ack is received by the raspberry pi either because the frame was not sent or the ack was not received then the frame will be sent again to ensure that the data sent has been received by the pc server (virtualization).

In Raspberry pi, data from the three sensors are read and integrate, with an identifier is added for each data. There are 4 data used in this paper, i.e. temperature, humidity, light condition, and motion detection, with the format: aw, bx, cy, dz, where a, b, c, and d are identifiers, while w, x, y, and z are the value which can be an integer or a float.

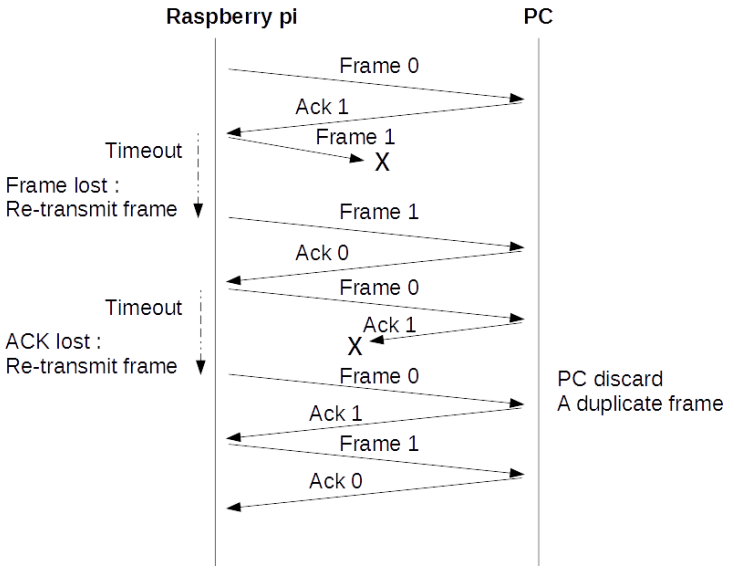


Figure 6. Stop and wait arq.

The encryption and decryption method [11] used in this paper is a combination of the caesar cipher and rail fence cipher methods, the purpose is to make it more difficult for unwanted person to decrypt the data, but the processing time is still fast, so it does not require large cpu resources. In the encryption process, first, the original data (plaintext) is encrypted using the caesar cipher method using Equation 1.

$$E_n(x) = (x+n) \bmod 26 \quad (1)$$

Where x is the data to be encrypted and n is the number of shifts. For example the data is "abcdefghijklm" and the shift is 3, so the result of the encryption is "defghijklmnop". Second, the result of encryption using the caesar cipher will then be encrypted again using the rail fence cipher method with the number of rails is 3 as shown in Figure 7.

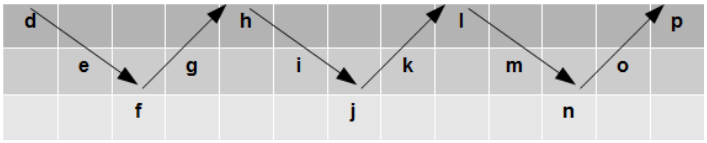


Figure 7. Rail fence cipher.

In Figure 7, the encryption data is read from left to right in each line, so the final data encryption is "dhlpegikmnfjn". In the decryption process, first, the data is decrypted using the rail fence cipher method as in Figure 7, but the way to read it is following the direction of the arrow from left to right, so the data decryption is "defghijklmnop". Second, the data is decrypted again using the caesar cipher method as seen in Equation 2.

$$D_n(x) = (x-n) \bmod 26 \quad (2)$$

Where x is the data to be decrypted and n is the number of shifts. So that the final data decryption is the same as the original data, that is "abcdefghijklm".

Crc method is an efficient method of checking originality of data and can detect more number of errors compared to other methods such as parity check [12]. This method is a type of Backward Error Correction (BEC) which is suitable for use in conditions where the possibility of error is small [13]. Example of the crc method can be seen in [14]. The crc method works by dividing data with certain polynomials until a remainder is obtained. This remainder is the crc value of the data. The crc method used in this paper is 16 bits. The originality of the data can be

checked in pc server using two ways, first, dividing data + crc with the same polynomial, if the remainder = 0 then the data is correct. Second, dividing the data with the same polynomial, if the remainder = crc then the data is correct. The method used in this paper is the second method. In Figure 8, you can see the one phase of the data blocks (frame) transmission between the client (raspberry pi) and pc server (virtualization)

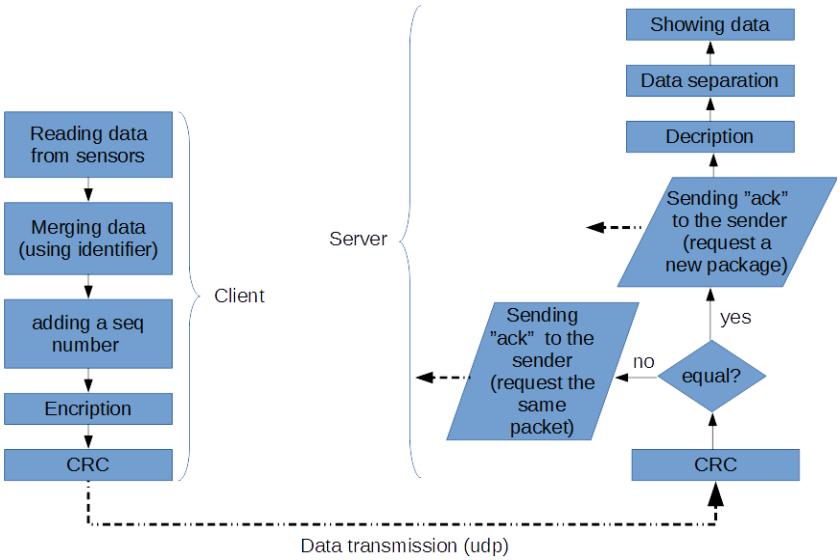


Figure 8. The diagram of one phase of the data transmission between the client (raspberry pi) and pc server (virtualization).

In Figure 8, you can see the sequence of processes on the client (raspberry pi) and pc server (virtualization). The value of sequence and ack numbers are only two, either 0 or 1, it can also be seen in Figure 6. In the raspberry pi, before the data block (frame) is sent, all data from the sensor is read and integrate using an identifier, then the sequence number is added, then data encryption, and finally the crc value is calculated. On the pc server, after the frame is received, the originality of the data will be checked using crc method, if it does not match, then the pc server will send a request to resend the same frame, if appropriate then the pc server will send the next frame request, after that, a decryption of the data is performed, and separating the data based on identifier, finally, data will be displayed and updated to the web server. The application program has been created using the C programming language with IDE Code::Blocks to implement the method in this paper.

Some examples of programming syntax that have been made in this paper are: void read_temperature(); is a procedure to read temperature and humidity data from the DHT11 sensor. Output data is stored on the global variable dht11_dat. Whereas int digitalRead(LDR); and int digitalRead(MOTION); are function to read a light condition and a motion detection. Whereas, snprintf [15] is using to integrate all data. Frame data type are formed data types that contain no sequence, ack, crc, and data. void crf_encrypt(char *text, int key, char *text_encrypt); and void crf_decrypt(char *cipherText, int key, char *text); are used to encrypt and decrypt the data in this paper. void crcInit(void); is used to initialize the crc data, while crc crcFast(unsigned char const message[], int nBytes); is used to calculate crc values. The source code of crc used in this paper is based on Barr [14] with several changes. int update_to_webserver (float suhu, float kelembaban, int cahaya, int gerakan); is used to upload data received by the pc server to the web server. Experiments of the processing time and the data transmission time using lan and wifi media have been measured using gettimeofday() from #include <sys/time.h> header [16].

LONTAR KOMPUTER VOL. 10, NO. 3 DECEMBER 2019 p-ISSN 2088-1541
DOI : 10.24843/LKJITI.2019.v10.i03.p05 e-ISSN 2541-5832
Accredited B by RISTEKDIKTI Decree No. 51/E/KPT/2017

3. Result and Discussion

3.1. The data transmission experiments

The data transmission experiments from the raspberry pi to the pc server (virtualization) had been done in the room condition telemonitoring application using LAN and Wifi communication media. Hardware of the room condition telemonitoring application can be seen in Figure 9.

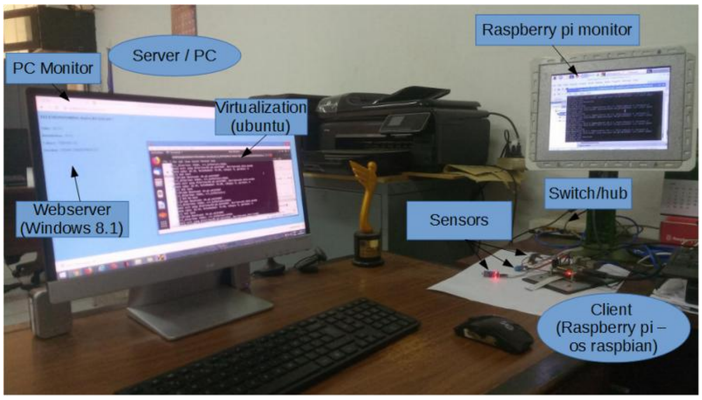
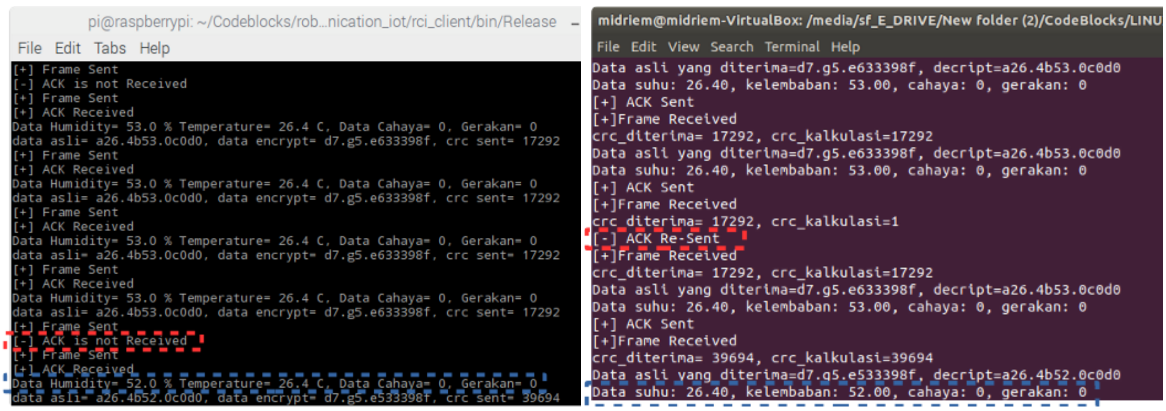


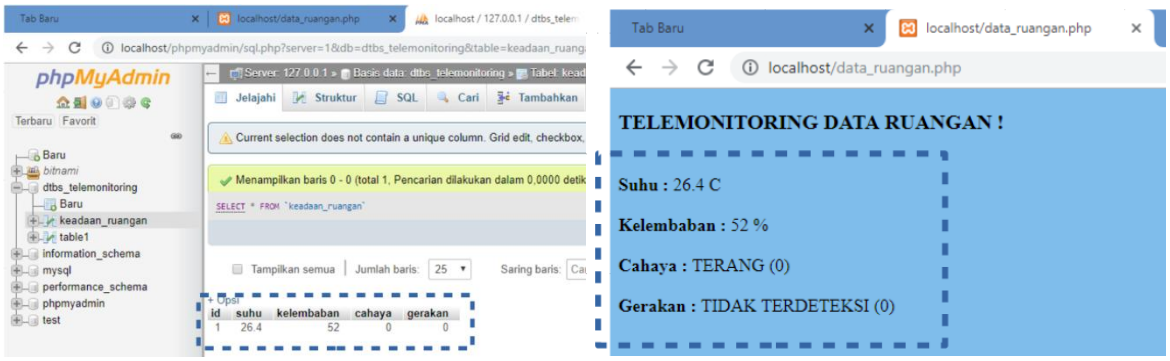
Figure 9. Hardware of the room condition telemonitoring application

The data transmission experiments had been done 300 times which divided into 3 x 100 experiments. In the experiments, it was found that there was a interference from the communication media, in addition other interference simulations had been added to ensure that this method could work properly and reliably. From 300 times experiments, it was found that all data sent by raspberry pi can be received perfectly by the pc server. An display example of the data transmission can be seen in Figure 10 and the detail of the data transmission experiments can be seen in Figure 13. In Figure 10, it can be seen an example of data display i.e. temperature, humidity, light condition, and motion detection in raspberry pi, pc server (virtualization), mysql, and a web browser in a box with a blue dotted line that shows the same value, the temperature = 26.4 ℃, humidity = 52%, light = 0 (brightness), and movement = 0 (no movement detected). The experiments show that the data sent can be received perfectly. Based on Figure 10, it can also be seen that the encrypted data is d7.g5.e533398f, the original data (plaintext) and the decryption result are the same, is a26.4b52.0c0d0, and the results of the crc calculation sent and received are the same, is 39694. Furthermore, in Table 2, it can be seen a few examples of the results obtained from each process starting from reading data from the sensors, integration data, encryption / decryption, until the crc calculation. Other experiments had been done to measure the speed of the processing time from integration data until the data (frame) is ready to be sent on the raspberry pi, and the processing time from the frame is received until the data is ready to be displayed on the pc server to see the performance of the proposed method which can be seen in Figure 11 and Figure 12.



(a)

(b)



(c) (d)

Figure 10. An display example of the data transmission software on: (a) client / raspberry pi; (b) pc server (virtualization); (c) pc web server (mysql); (d) pc web server (web browser)

Table 2. A few examples of reading data from the sensors, integration data, encryption / decryption, and the crc calculation

Temperature	Humidity	Light condition	Motion detection	Data blok (plainteks/ decryption)	Encryption	Crc calculation
26.4	52.0	0	0	a26.4b52.0c0d0	d7.g5.e533398f	39694
25.9	49.0	0	0	a25.9b49.0c0d0	d2.g5.e233387f	58762
26.5	52.0	1	0	a26.5b52.0c1d0	d8.g5.e534398f	15752
26.5	52.0	0	1	a26.5b52.0c0d1	d8.g5.e533498f	2929
26.30	52.0	1	1	a26.3b52.0c1d1	d6.g5.e534498f	59540

Experiments to measure the speed of the processing time in Figure 11 and Figure 12 were carried out 100 times. In Figure 13, it can be seen that the average processing time from integration data until the data (frame) is ready to be sent on the raspberry pi is 0.019 seconds (19 ms), and in Figure 12, it can be seen that the average processing time from the frame is received until the data is ready to be displayed on the pc server is 0.000005 seconds (5 us). Based on the experiments, it can be seen that the processing time required to process this method is fast, so the proposed method can be implemented properly in the room condition telemonitoring application or other similar applications. The data transmission time had also been measured using LAN and Wifi media which can be seen in Figure 13.

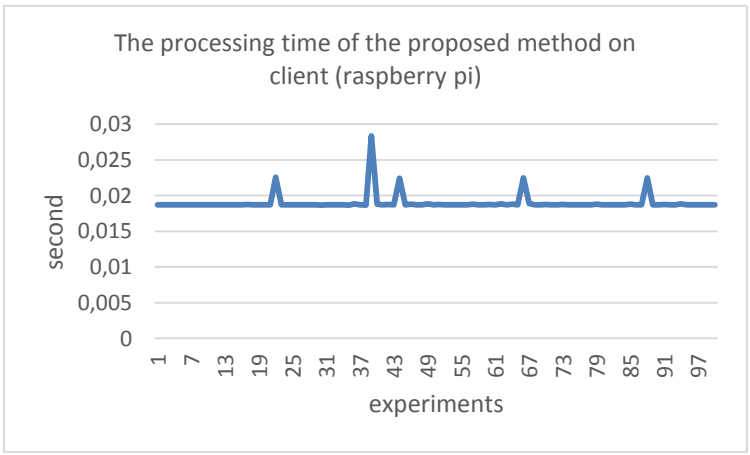


Figure 11. The processing time from integration data until the data (frame) is ready to be sent on the raspberry pi

p-ISSN 2088-1541
e-ISSN 2541-5832

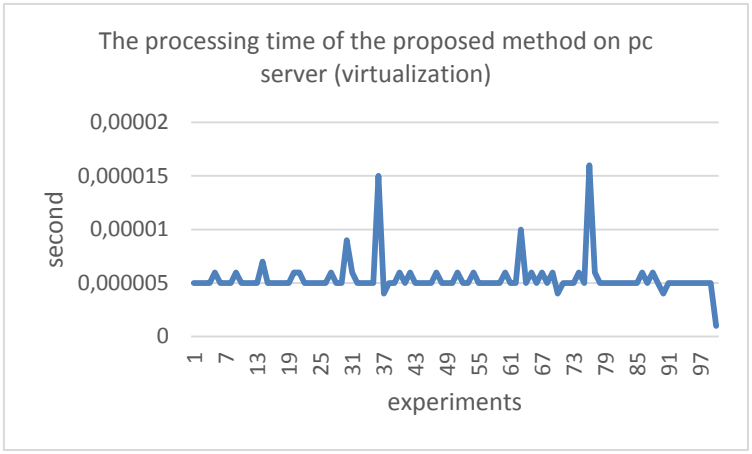
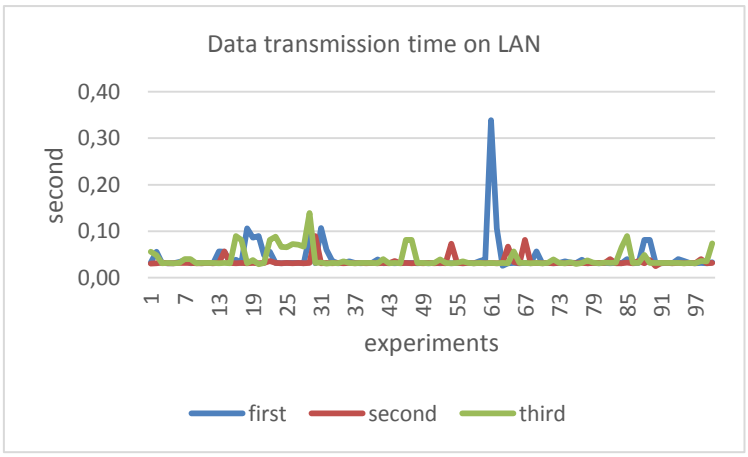
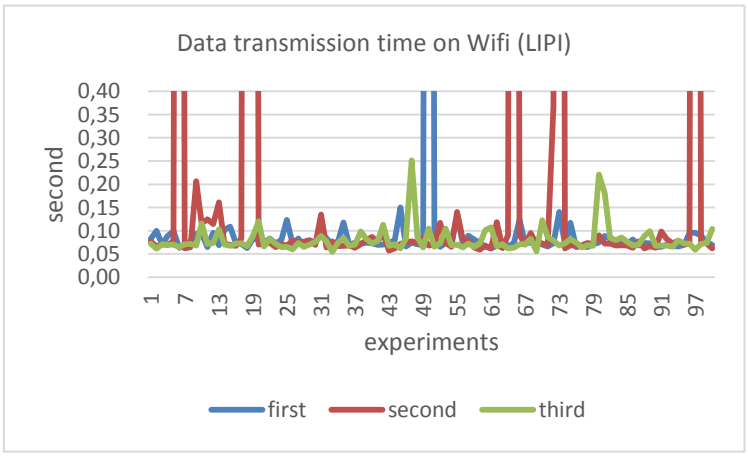


Figure 12. The processing time from the frame is received until the data is ready to be displayed on the pc server



(a)



(b)

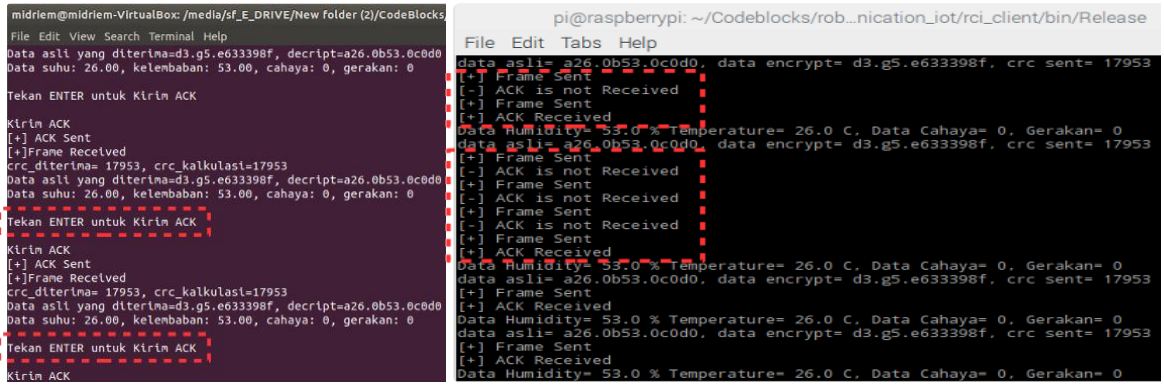
Figure 13. The speed of the data transmission time starts from the data sent from the raspberry pi to the pc server until ack is received by raspberry pi from the pc server using a communication media: (a) LAN; (b) Wifi

The number of experiments of data transmission time speed in Figure 13 is 300 times (3 x 100 times), which is measured from the data sent by the raspberry pi to the pc server until ack is received by raspberry pi from the pc server using LAN media (Figure 13.a), Wifi (Figure 13.b).

the time intervals in Figure 13.b is reduced to make it clearer. Based on the experiments, it was found that the average data transmission time speed using LAN media is 0.04 seconds, and using Wifi media is 0.08 seconds. It should be noted that the time is two times of transmission (data and ack) along with the process of checking the originality of the data on the pc server, so that it can be seen that the data transmission time is fast and can be used on this system where the data transmission interval is only every minute, so still far from the transmission time of the next data.

3.2. The interference experiments

In Figure 13.b, it can be seen that there are several times where the processing time is 5 seconds, this is because the waiting time is 5 seconds whereas the ack is not received in raspberry pi. It can happen because either the frame from raspberry pi is not received in pc server so pc server is not send the ack, or the ack from pc server is not received in raspberry pi. An example of this case can be seen in Figure 10.a and Figure 10.b, in a box with a red dashed line where in raspberry pi (Figure 10.a) it says "ack is not received" because ack is not received, whereas in the pc server (Figure 10.b) is "ack re-sent", it is mean pc server wants raspberry pi to re-sent the same frame. Another interference simulation had been done to test the reliability of this method as shown in Figure 14.

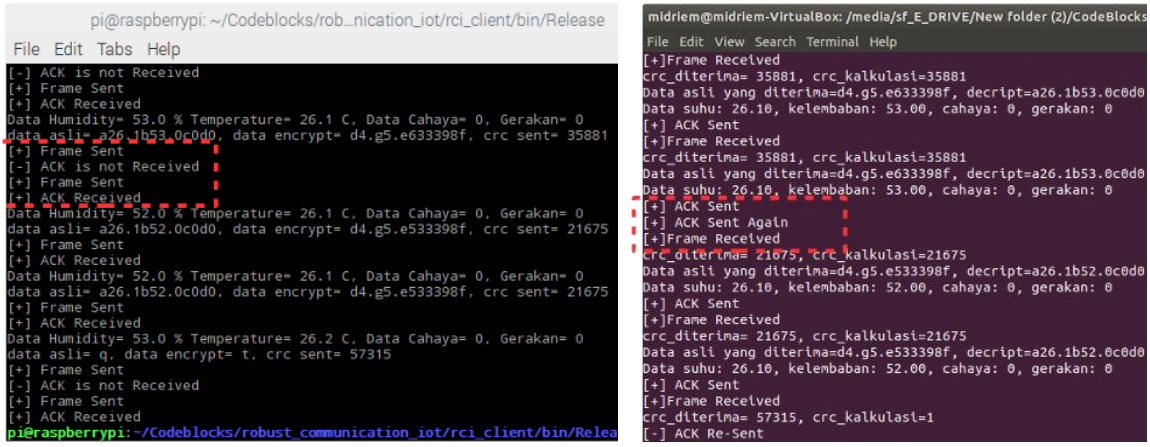


(a) (b)

Figure 14. An display example of an interference simulation by not sending an ack from the pc server until the enter key is pressed on: (a) pc server; (b) raspberry pi.

In the box with a red dashed line in Figure 14, it can be seen that the interference done by not sending ack from the pc server (Figure 14.a) until the enter button is pressed, so the raspberry pi (Figure 14.b) shows "ack is not received" which means ack is not received in raspberry pi. Longer the enter key is not pressed, more warnings “ack is not received” will show. Another interference simulation are also been done as in Figure 15.

The interference simulation in the box with a red dotted line in Figure 15 is by modifying the source code in the raspberry pi so that at certain iterations, raspberry pi will says that the ack is not received or the ack was received is incorrect (Figure 15.a), then the pc server will send ack again to raspberry pi as shown in Figure 15.b. From all the interference experiments, it can be seen that the interference handling mechanism of the proposed method can work well and the data sent by the raspberry pi can still be received by the pc server.



(a) (b)

Figure 15. An display example of an interference simulation on the raspberry pi which states that the ack is not received or the ack received is incorrect on: (a) raspberry pi; (b) pc server.

4. Conclusion

The data transmission method proposed in this paper had been successfully made and implemented in the a room condition telemonitoring application to transmit data from the raspberry pi to the pc server (virtualization) every minute where the data are temperature, humidity, light condition, and motion detection. The data transmission method proposed in this paper is using the stop and wait arq mechanism, where each frame sent is an integration of all data from three sensors using an identifier, adding encryption and decryption, and checking the originality of the data using the crc method. Based on the experiments, it can be seen that the data sent by raspberry pi can be received perfectly by pc server (virtualization), despite there are interferences from the communication media, and other interference simulations, and the processing time is fast.

Acknowledgments

The author would like to thank the Research Center for Electrical Power and Mechatronics - Indonesian Institute of Sciences (LIPI) especially the Industrial Automation Research Group which has supported this research.

References

[1] K. Kumar, D. Zindani and *J. P. Davim.*, Industry 4.0 Developments towards the Fourth Industrial Revolution, 1st ed., Singapore: Springer, 2019. pp. 59.

[2] N. Bouchemal, R. Maamri and N. Bouchemal, “Mobile Agent System Based Cloud Computing for Ubiquitous Telemonitoring Healthcare,” in *International Conference on Mobile, Secure, and Programmable Networking*, Paris, 2018, vol. 4, pp. 107-116.

[3] T. H. Y. Ling and L. J. Wong, “Elderly infrared body temperature telemonitoring system with XBee wireless protocol,” in *International Conference MSPN 2018*, Paris, 2018, vol. 22, pp. 103–120.

[4] M. Mirdanies, “Optimization of Robot Telemonitoring System Software using multi-thread method” *Journal of Informatics, Control Systems, and Computers (INKOM Journal)*, vol. 11, no. 1, p. 15–24, 2018.

[5] M. Elhoseny, G. Ramírez-González, O. M. Abu-Elnasr, S. A. Shawkat, A. N and A. Farouk, “Secure Medical Data Transmission Model for IoT-Based Healthcare Systems,” *IEEE Access*, vol. 6, pp. 20596–20608, 2018.

[6] C. P. Kumar and R. Selvakumar, “Erasure Codes for Reliable Communication in Internet of Things (IoT) Embedded with Wireless Sensors,” in Lecture Notes on Data Engineering and Communications Technologies book series (LNDECT, volume 14), Cham: Springer, 2017, pp. 115–137.

[7] H. C. Hwang and J. G. Shon, “Design and Implementation of a Reliable Message

Transmission System Based on MQTT Protocol in IoT," *Wireless Personal Communications*, vol. 91, no. 4, pp. 1765–1777, 2016.

[8] M. Mirdanies, H. M. Saputra and E. Rijanto, "Algorithm of 32-bit Data Transmission Among Microcontrollers Through an 8-bit Port," *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 6, no. 2, p. 75-82, 2015.

[9] N. Cameron, "Sensors," in *Arduino Applied*, Berkeley, CA: Apress, 2019, pp. 31–78.

[10] A. Rayes and S. Salam, "The Internet in IoT," in *Internet of Things From Hype to Reality*, Cham: Springer, 2019, pp. 37–65.

[11] I. Alsmadi, R. Burdwell, A. Aleroud, A. Wahbeh, M. Al-Qudah and A. Al-Omari, "Encryption and Information Protection/Integrity and Concealment Methods: Lesson Plans," in *Practical Information Security*, Cham: Springer, 2018, pp. 91–120.

[12] S. Jiang, "Error Control," in *Wireless Networking Principles: From Terrestrial to Underwater Acoustic*, Singapore: Springer Singapore, 2018, pp. 35–50.

[13] P. Ivaniš and D. Drajić, "Cyclic Codes," in *Information Theory and Coding - Solved Problems*, Cham: Springer, 2017, pp. 237–325.

[14] B. Group, "CRC Mathematics and Theory | Embedded Systems Experts." [Online]. Available: http://www.barrgroup.com/Embedded-Systems/How-To/CRC-Math-Theory. [Accessed: 19-Aug-2019].

[15] M. Olsson and M. Olsson, "Strings and Numbers," in *Modern C Quick Syntax Reference*, Berkeley: Apress, 2019, pp. 109–112.

[16] K. C. Wang and K. C. Wang, "Timers and Time Service," in *Systems Programming in Unix/Linux*, Cham: Springer, 2018, pp. 187–204.