JEEMECS (Journal of Electrical Engineering, Mechatronic and Computer Science
Vol. 4, No. 1, February 2021, pp. 31-36
ISSN 2614-4859

Wi-Fi-Based Internet of Things (IoT) Data Communication Performance in Dense Wireless Network Traffic Conditions

Michael Ardita [a,1,*], Mira Orisa [a,2]

[a] Department of Electrical Engineering - National Institute of Technology, Malang 65145, Indonesia
[1] michael.ardita@lecturer.itn.ac.id *; [2] mira.orisa@lecturer.itn.ac.id
[*] corresponding author

Keywords
Internet of Things (IoT)
wireless network performance
Wi-Fi

ABSTRACT

Currently, there are many Internet of Things (IoT) devices use Wi-Fi networks to connect to the Internet. As the Wi-Fi network frequency band is used by many parties, the possibility of disconnecting the Internet connection with IoT devices is still high. In this research, we explored the performance of IoT on Wi-Fi networks in high traffic conditions. System testing in this study was carried out using several IoT devices at different distances from the AP. The experimental environmental conditions were also formed so that the Wi-Fi traffic was quite high, generated by software from several laptops. The test results showed that the packet loss in very crowded traffic conditions reached 100%. Round trip time (RTT) data transmission delay of less than 10 ms during normal conditions increased to thousands of milliseconds when the Wi-Fi network conditions were very heavy.

1. Introduction

The development of information and communication technology nowadays has been able to transmit digital information wirelessly at high speed. Currently, communication via the internet has also been applied to communication between physical objects around us [1]. Communication between physical objects is often referred to as the Internet of Things (IoT) [2]. Currently, the predicted machine-to-machine (M2M) connection is more than the total population in the world [3]. The current contributor to IoT communication traffic is also because many cities are competing to become smart cities [4]. With the IoT, many activities can be controlled by simply using a mobile phone. Unfortunately, many users are not aware of the effect of Wi-Fi traffic density on the quality of the connection between control devices and IoT devices. Many users complain of sudden disconnection with IoT devices when many people are at the location of the IoT device. Based on these conditions, we aim to explore more deeply by doing experiments to get answers to this phenomenon. The results of this research are expected to be used to build better data communication protocols for IoT networks

2. The Proposed Method/Algorithm

At first, data communication was only for communication between computers (client-server), while nowadays, the Internet has connected billions of devices spread across various countries. Currently, wireless access to the Internet can use Wi-Fi or cellular telecommunications networks. For wireless Internet access at a distance of fewer than 100 meters from the AP, you can use Wi-Fi. For wireless Internet access over a distance of several km, you can use a 3G or 4G (LTE) cellular network. LTE already has high data transfer rates up to 300 Mbps downlink and 75 Mbps for uplink [5].

The results of previous studies indicated that the latency of data transmission over the Internet on wireless cellular networks was greatly affected by environmental conditions [6]. Many IoT devices use Wi-Fi to connect to the Internet. Wi-Fi works on the IEEE 802.11 standard for sharing channel access together by multiple users [7] [8]. Currently, Wi-Fi technology is more than 2

decades old. A new, more efficient, Wi-Fi standard is IEEE 802.11ax [9]. Low-speed IoT devices in general have not adopted the new standard.

The challenge faced in sending information over a wireless network is the possibility of packet loss due to the high traffic density on radio channels. Packet loss can also occur due to the phenomenon of attenuation of the carrier radio signal due to obstructions and multiple trajectories. Packet loss can also occur due to interference from hidden terminals operating on the same frequency channel [10]. Wi-Fi uses the IEEE 802.11 carrier sense multiple access/collision avoidance (CSMA/CA) mechanism to share access time on a wireless channel. Before sending data, generally, the source node will send a request to send (RTS) to the destination node. As the data from the IoT device is the Wi-Fi access point, the RTS data packet will first be sent to the access point. The access point will reply with a Clear to Send (CTS) data packets if the channel is empty. The RTS-CTS process then followed by sending the data packets.

One of the most widely used IoT devices is the ESP8266 [11]. If the traffic conditions are heavy, it is possible that the RTS from the IoT device will be hit by a data packet from another device that has a stronger signal. This condition will be very pronounced if the IoT device is in an outdoor location. Outdoor Wi-Fi devices generally have great power when used as a link for long distances. Because the transmit power of IoT devices is small, it is likely that RTS will not be heard by outdoor Wi-Fi devices that are far away. As a result, IoT devices do not get the opportunity to send data plans until the time-out limit.

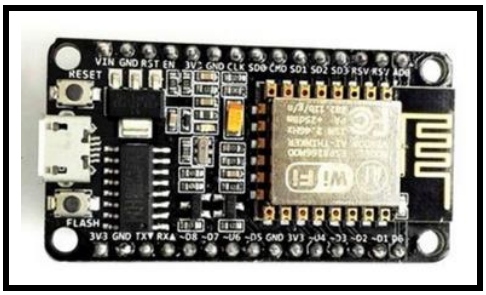

Fig. 1. NodeMCU ESP8266

In this study, the IoT module used was NodeMCU ESP8266. This module was integrated between the micro-controller and the Wi-Fi module which works at a frequency of 2.4 GHz. Details of the NodeMCU module can be seen in Figure 1. This module has a fairly large transmit power (about 25 dBm). Unfortunately, this module only has a small antenna so it cannot reach a large distance. At NodeMCU, there is also an ADC with a resolution of 12 bits so that it can be used directly for analog signal conversion from a sensor.

3. Research Methodology

Experiments in this measurement were carried out by emulating a heavy traffic Wi-Fi network. The experimental design can be seen in Figure 2. In this experiment, 4 IoT devices were used with locations that were spread out at several points with different distances from the AP. One of the IoT device nodes placed close to the AP was used as a reference for the other nodes. Each IoT device sent its data to IoT-Server #1 and IoT-Server #2. IoT-Server #1 was a server connected to the Internet and IoT-Server #2 was an IoT server located on a local computer network. Wi-Fi traffic on a large scale can be generated by software from one or two laptops. Server #3 (Dense Traffic Server) was a server that acts to send a large response data plan (1000 bytes per data packet) when triggered by data sent from a laptop (traffic generator client). The size of the trigger data from the laptop (traffic generator client) can be adjusted to the size of 100 bytes or 1000 bytes per data packet.

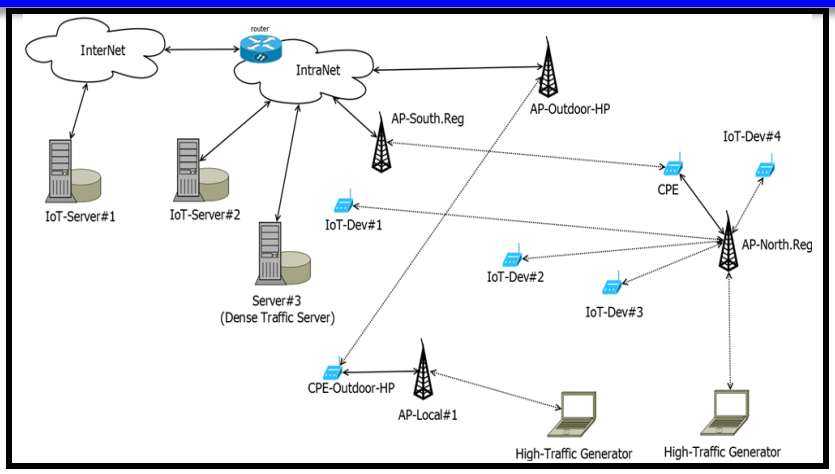

Fig. 2. Network Topology for Dense WiFi Traffic Experiment

There were 4 access points (AP) used with the working frequency set on channel 11. All of these APs were intended to be able to generate fairly dense Wi-Fi traffic on channel 11. AP.North.Reg is an AP that serves IoT and one of the laptops that were used to trigger large traffic on a local Wi-Fi network. AP.North.Reg was configured like a wireless repeater that will load the wireless traffic twice. This repeater configuration had two access points which caused the traffic to even denser. Therefore, the collision domains on the local wireless network will be quite high. On Wi-Fi (IEEE-802.11), there were request-to-send (RTS) and clear-to-send (CTS) sending mechanisms for CSMA-CA. With the custom-built repeater and traffic generator software, each Wi-Fi node (AP, Client, Node-MCU) will generate random RTS-CTS which will also generate a denser Wi-Fi traffic.

4. Results and Discussion

To get optimal results, the experiment was carried out in a span of 50 minutes. The measurement results were divided into several parts, namely:

a. Measurement of signal strength on each IoT device;
b. Measurement of traffic on the intruding side (client and server intruders);
c. Measurement of communication performance on IoT devices when network traffic conditions were heavy.

4.1. The Results Of Measuring The Wi-Fi Signal Strength On Each Iot Device

To find out the strength of the Wi-Fi signal, at each IoT node, the process of reading the receive signal strength indicator (RSSI) was also carried out. RSSI showed the strength of the Wi-Fi signal received from the AP. The value of the average RSSI for each IoT device can be seen in Table 1. Node-C had an RSSI of -52 dBm as it was the closest to the AP. Node-D and Node-E have located some distance from the AP so they got a power of around -80 dBm. Node-F was the most remote IoT device so it only got a signal of -90 dBm. By using several RSSI levels, it is expected to found which RSSI level IoT devices failed to send their data packets.

Table 1. Average RSSI (Receive Signal Strength Indicatior (RSSI) at Every IoT Node

Table Head	*RSSI*	
	dBm	***mWatt***
IoT-LoLinC	-52.2737	0.00000592420
IoT-LoLinD	-90.1551	0.00000000096
IoT-LoLinE	-81.8223	0.00000000657
IoT-LoLinF	-79.7695	0.00000001055

4.2. Measurement results on the side of the disturbing traffic source

The heavy traffic was generated at 2 separate time frames. The phase 1 package was carried out from the 12th to the 17th minute. The second phase of packets flooding was carried out in the time range of the 40th to the 50th minute. The data packet size of the laptop in stage 1 is 100 bytes

per packet. The data packet size of the laptop in stage 2 was 1000 bytes per packet. The results of the measurement of traffic volume disturbance generated by the laptop and the traffic generator server can be seen in Figure 3. The largest traffic occurred in the 47th-minute time slot, the average Tx throughput was 143,483 bytes/sec and the average Rx throughput was up to 114,650 bytes/sec. The average number of data packets sent was also 143.5 pps (Tx) and 114.7 pps (Rx). With a total data packet of around 258.1 pps, going through a repeater produced 516.2 pps. Before sending data, each Wi-Fi node will perform a CSMA-CA mechanism by sending RTS and CTS packets. Therefore, there will also be more data packets on the Wi-Fi network. From Figure 3, it can be seen that a larger trigger data packet size in stage 2 can provide a traffic flood with a larger amount than stage 1.

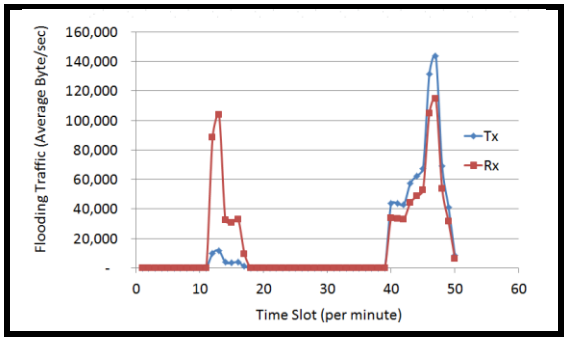

Fig. 3. Measurement Byte (per 8 bit) Rate of Traffic Flooding

Detailed data from the traffic disturbance generated by the software are shown in Table 2. Tx.B/s is the rate of data sent from the laptop to the server kerf. Rx.B/s is the rate of response data from the server to the client. Tx.Pkt/s is the rate of the trigger packet generated by the software on the laptop. Rx.Pkt/s is the rate of response data packets from the server received by the laptop. Tot.Pkt is the combination of the values of Tx.Pkt/s and Rx.Pkt/s. The first row of the table shows that in the 12th-minute time slot the client is able to send data to the server at a rate of 9.935 bytes per second. Because the size of the data packet from client to server is 100 bytes, the average packet per second is around 99.5 pps. As the response data packet size from the server was 1000 bytes per packet, the reply data packet from the server was 88.567 bytes per second or as many as 88.6 packets per second. From the traffic disturbance side, it can also be seen that the returned packet was smaller than the data packet sent by the client to the server (from 99.5 pps sent only 88.6 pps reply). Interruption traffic in the first stage was generated in the time slot of the 12th minute to the 17th minute. The next interruption traffic was generated at the 40th to the 50th minute with a larger trigger data packet size.

Table 2. Measurement Data Rate of Traffic Flooding

Time Stamp	Slot	Tx.B/s	Rx.B/s	Tx.Pkt/s	Rx.Pkt/s	Tpt.Pkt/s
20201207-1212	12	9,953	88,567	99.5	88.6	188.1
20201207-1213	13	11,707	103,667	117.1	103.7	220.7
20201207-1214	14	3,785	32,350	37.9	32.4	70.2
20201207-1215	15	3,582	30,517	35.8	30.5	66.8
20201207-1216	16	3,865	33,100	38.7	33.1	71.8
20201207-1217	17	1,117	9,600	11.2	9.6	20.8
	-	-	-	-	-	-
20201207-1240	40	43,683	33,817	43.7	33.8	77.5
20201207-1241	41	43,800	33,417	43.8	33.4	77.2
20201207-1242	42	43,050	33,050	43.1	33.1	76.1
20201207-1243	43	57,150	44,117	57.2	44.1	101.3
20201207-1244	44	62,417	48,500	62.4	48.5	110.9
20201207-1245	45	67,400	52,717	67.4	52.7	120.1
20201207-1246	46	131,650	104,900	131.7	104.9	236.6
20201207-1247	47	143,483	114,650	143.5	114.7	258.1
20201207-1248	48	69,250	53,917	69.3	53.9	123.2
20201207-1249	49	41,250	31,333	41.3	31.3	72.6
20201207-1250	50	8,583	6,433	6.433	6.4	15.0

From the Node-MCU (IoT client) side, if the Wi-Fi channel is overload, the NodeMCU will wait for a quiet channel until a time-out limit to be able to send data packets. In general, MCU nodes will send data periodically to the IoT server. In the IoT software used in this experiment, data transmission was carried out in about 5 seconds. In normal conditions, the server-side will receive around 12 data packets from each IoT node every minute.

4.3. The Results Of Measuring The Performance Of Iot Data Communication During Heavy Traffic Conditions

The main result displayed was the data packet loss rate. Another measurement result was a round trip time delay (RTT delay) in sending information between the source node and the destination node. In terms of traffic that were used as a disturbance, it can also be measured the amount of traffic and the packet loss rate.

The effect of this traffic disturbance can be seen in Figure 4. Information C, D, E, and F shows the identity of the IoT device. C represents the identity of the IoT-LoLinC node. Likewise for D, E and F, under normal conditions, the average data packet received by the server is around 13 packets per second. In the event of traffic congestion on a Wi-Fi network (12 to 17 minutes and 40 to 50 minutes), the data packet received by the server every minute decreased significantly. The worst condition occurred in the 46th to 47th minute where no data packet from the IoT device with id D, E, and F reached the server. When traffic conditions are heavy, only IoT devices with node C identities can still survive sending data packets.

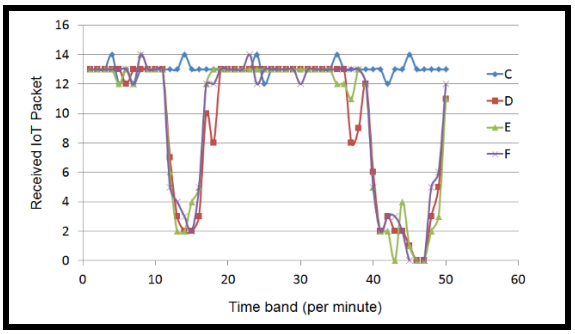

Fig. 4. Received IoT Packet per minute at Server Side

RTT delay measurement is carried out to observe the effect of traffic density on the length of time for sending data packets between the IoT nodes and the server. The RTT delay measurement results are shown in Figure 5. Under normal conditions, the RTT delay only takes less than 10 milliseconds. At peak times, the RTT delay can reach thousands of milliseconds. When viewed from the identity of the IoT device nodes, only node C has the RTT delay not affected by environmental disturbances.

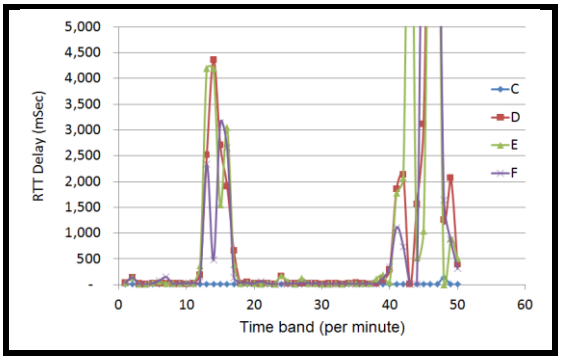

Fig. 5. Round Trip Time Delay Reply from IoT Server

In this experiment, one of the IoT servers is located in a data center in Jakarta. Judging from the long propagation time of Internet data packets on fiber optic media, within 1 millisecond (ms) of data packets can travel as far as 200 km. If it is assumed that the length of the optical cable between Malang and Jakarta is around 800 km, then the round-trip propagation time (RTT delay) is about 2 x 4ms or 8 ms. From this condition, it can be concluded that when the traffic is heavy, the waiting

time for access to the radio channel is much greater than the propagation time between the IoT nodes to the server.

5. Conclusion

From the experimental results, it was found that IoT (Node-MCU) devices failed to send IoT data packets when the Wi-Fi network traffic was high. The only MCU nodes that can survive to communicate with the AP during high traffic conditions were the only nodes that got a better Wi-Fi signal (around -60 dBm). IoT devices that only got an RSSI level of -80 dBm or less will be very easily disturbed in high Wi-Fi traffic. For the MCU nodes that were far from the AP, when the network traffic is high, they cannot send any data packets. In terms of RTT delay, when the network conditions are congested, RTT which is usually less than 10 milliseconds can become thousands of milliseconds.

References

[1] J. Jin, J. Gubbi, S. Marusic and M. Palaniswami, "An Information Framework for Creating a Smart City Through Internet of Things," IEEE Internet of Things Journal, vol. 1, no. 2, pp. 112-121, 2014.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," IEEE Communications Surveys Tutorials, vol. 17, no. 4, pp. 2347 - 2376, 2015.

[3] D. Evans, "The Internet of Things: How the Next Evolution of the Internet Is Changing Everything," https://cisco.com, 2011.

[4] M. Naphade, G. Banavar, C. Harrison, J. Paraszczak and R. Morris, "Smarter Cities and Their Innovation Challenges," IEEE Computer, vol. 44, no. 6, pp. 32-39, 2011.

[5] S. Kanchi, S. Sandilya, D. Bhosale, A. Pitkar and M. Gondhalekar, "Overview of LTE-A technology," in 2013 IEEE Global High Tech Congress on Electronics, Shenzhen, China, 2013.

[6] M. Ardita, Suwadi, A. Affandi and Endroyono, "HTTP communication latency via cellular network for Intelligent Transportation System applications," in 2016 International Conference on Information & Communication Technology and Systems (ICTS), Surabaya, Indonesia, 2016.

[7] G. Bianchi, "IEEE 802.11-saturation throughput analysis," IEEE Communications Letters, vol. 2, no. 12, pp. 318-320, 1998.

[8] I. Aad and C. Castelluccia, "Differentiation mechanisms for IEEE 802.11," in Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213), Anchorage, AK, USA, 2001.

[9] E. Khorov, A. Kiryanov, A. Lyakhov and G. Bianchi, "A Tutorial on IEEE 802.11ax High Efficiency WLAN," IEEE Communications Surveys & Tutorials, vol. 21, no. 1, pp. 197-216, 2019.

[10] D. Malone, P. Clifford and D. J. Leith, "MAC Layer Channel Quality Measurement in 802.11," IEEE Communications Letters, vol. 11, no. 2, pp. 143-145, 2007.

[11] R. K. Kodali and K. S. Mahesh, "Low cost ambient monitoring using ESP8266," in 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), Greater Noida, India, 2016.