p-ISSN 2085- 6989
e-ISSN 2654 - 4733
Elektron Jurnal Ilmiah Volume 14 Nomor 2 Desember 2022 **TERAKREDITASI RISTEKDIKTI No. SK : 28/E/KPT/2019**
DOI : https://doi.org/10.30630/eji.14.2.298

Penerapan *Hypertext Transfer Protocol Web Server* untuk *Over-The-Air Auto Update Firmware* pada Perangkat IoT

Iin Karmila Yusri[1*], Kasim[2], Andi Muhammad Akbar[3]
[1,2,3] Program Studi Teknik Komputer dan Jaringan, Politeknik Negeri Ujung Pandang
*Corresponding Author, email:iin.yusri@poliupg.ac.id

Abstract— Perangkat dalam suatu sistem IoT seringkali terletak dalam banyak tempat dan secara fisik sulit dijangkau oleh administrator sistem. Selain itu siklus hidup perangkat lunak dan *firmware* perangkat yang pendek memerlukan pembaruan secara berkala untuk meningkatkan fungsi sistem yang ditanam pada perangkat agar dapat menyesuaikan dengan kebutuhan. Hal ini menunjukkan sulit melakukan *update* secara fisik secara berulang. Sistem *auto update firmware over the air* menjadi solusi untuk masalah akses fisik dan pembaruan secara berkala yang dibangun dengan menerapkan *Hypertext Transfer Protocol Web Server*. Sistem dibuat dalam bentuk aplikasi web sebagai interface untuk mengatur pembaruan sistem dan perangkat lunak pembaruan *firmware* yang ditanam pada perangkat IoT. Hasil yang didapatkan menunjukkan sistem berhasil melakukan pembaruan *firmware* secara otomatis pada beberapa perangkat IoT yang terhubung dalam jaringan yang berbeda dalam waktu bersamaan. Sistem juga terbukti dapat melayani sekitar 29.000 *request* untuk pembaruan dalam waktu bersamaan.

Kata kunci: *HTTP web server, Over-the-air, pembaruan otomatis, perangkat IoT*

Abstrak— Devices in the IoT system are located in many places and difficult to be accesses physically by system administrator. Besides, short life cycle of software/firmware requires to be updated periodically in order to improve the embedded system function to meet the system requirements. These conditions shows the difficulty to update the devices physically. Auto update firmware over the air system became a solution for physical acsess and periodically updated problem which developed by implementing Hypertext Transfer Protocol Web Server. This research is part of study of developing IoT Management System in agricultural environment. This system is in form of web application as an interface to control and manage the update system and a software for update firmware that embedded in the IoT devices. The results shows that the system is able to update firmware of the IoT devices that connected to different network at the same time automatically. Besides, this system can serve around 29,000 request for firmware update concurrently.

Keywords: *HTTP web server, firmware, auto update, Over-the-Air, IoT devices*

I. PENDAHULUAN

Dalam sistem Internet of Things (IoT) terdapat kumpulan perangkat yang saling terhubung dengan perangkat lain melalui jaringan Internet yang berfungsi untuk merekam data pada sebuah lingkungan atau objek. Data tersebut, kemudian diteruskan ke sebuah aplikasi yang berada pada Internet lalu diolah lebih lanjut untuk menampilkan informasi yang tersimpan di balik kumpulan data tersebut [1]. Dalam implementasinya, perangkat IoT ini dapat berkomunikasi tanpa intervensi manusia.

Saat sistem IoT diimplementasikan, pada waktu tertentu perangkat IoT dalam sistem akan mengalami perubahan. Perubahan ini bisa berupa perubahan *behavior*, masalah keamanan, atau parameter terkait komunikasi dengan sistem atau perangkat lain [2]. Perubahan ini bisa diatasi dengan melakukan pembaruan (*update*) *firmware* [3]. Siklus hidup perangkat lunak pada perangkat IoT memerlukan pembaruan untuk meningkatkan fungsi sistem yang tertanam pada perangkat tersebut agar dapat terus berfungsi sesuai kebutuhan sistem [4].

Pembaruan *firmware* pada perangkat IoT sebelumnya dilakukan dengan mengambil perangkat IoT, menghubungkan ke komputer secara fisik melalui kabel, melakukan pembaruan, dan mengembalikan perangkat IoT ke tempat semula [5]. Pembaruan *firmware* dengan cara ini cukup sulit dilakukan karena kumpulan perangkat dalam suatu sistem IoT biasanya terletak dalam banyak tempat dan secara fisik sulit di jangkau oleh administrator sistem [5]. Penggunaan metode pembaruan *firmware* seperti ini, juga berpotensi mengganggu keamanan sistem [6].

Over-the-air (OTA) menjadi solusi untuk masalah pembaruan *firmware* secara fisik [1]. OTA adalah suatu mekanisme yang memanfaatkan jaringan *wireless* untuk mengirim data, melakukan pembaruah paket/ *firmware* dan perangkat lunak ke perangkat *mobile*, sehingga *user* dapat mengakses perangkat untuk mengubah aplikasi, parameter, *firmware* tanpa harus berada dilokasi perangkat berada [1]. Dengan OTA,

Received 08 November 2022; Revised 20 November 2022; Accepted 22 Desember 2022

administrator sistem IoT dapat mengubah dan/atau memperbarui aplikasi, parameter, *firmware*, dan perangkat lunak pada perangkat IoT tanpa mengakses perangkat secara fisik tapi melalui jaringan nirkabel .

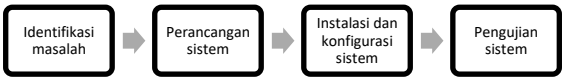
Sistem pembaruan *firmware* melalui OTA pada perangkat IoT sudah ada di pasaran, namun terbatas pada perangkat IoT dengan merek yang sama, seperti Libelium yang memanfaatkan *File Transfer Protocol* (FTP) [4]. Namun suatu sistem manajemen IoT yang menjalankan OTA *update* dapat dibangun untuk menghubungkan mikroprosessor dan perangkat lunak di perangkat IoT tanpa terikat merek perangkat yang sama [7]. OTA dapat melakukan *update firmware* melalui Arduino IDE, *web browser*, dan *Hypertext Transfer Protocol* (HTTP) *Server* [4]. Penelitian yang dilakukan oleh [4] dan [8] berhasil melakukan melakukan OTA *update* dengan Arduino IDE. Namun *update* hanya bisa dilakukan dalam jaringan yang sama. Sedangkan perangkat dalam suatu sistem IoT seringkali berada dalam jaringan yang berbeda sehingga *update* dilakukan berulang untuk setiap jaringan.

Selain itu, proses *update firmware* hanya dilakukan jika ada *request* dari perangkat [9], dan menunggu respon dari administrator IoT untuk menjalankan proses *update*.

Studi ini membangun sistem OTA *update firmware* dengan memanfaatkan HTTP webserver. Menurut [10], *update firmware* melalui HTTP *server* dapat dilakukan dengan jaringan yang berbeda. Sistem yang dibangun ini memanfaatkan HTTP Web Server untuk *update firmware* melalui OTA secara otomatis dan dapat dijalankan untuk perangkat IoT yang berada dalam jaringan yang berbeda secara bersamaan.

II. METODE

Sistem *auto update firmware* OTA menggunakan HTTP *Web Server* dibuat dalam bentuk aplikasi berbasis *website* sebagai *interface* pengguna dan sistem *auto update firmware* yang di tanam di perangkat IoT. Pembangunan sistem ini dilakukan dengan beberapa tahapan, yaitu identifikasi masalah, perancangan sistem, instalasi dan konfigurasi, dan pengujian sistem., seperti terlihat pada Gambar 1.

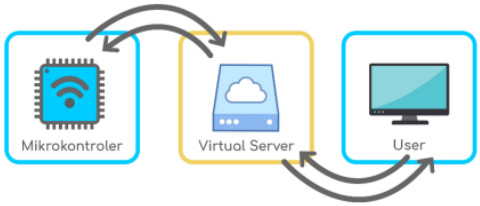


Gambar 1. Tahapan penelitian

Pada penelitian ini di identifikasi masalah pada proses *update firmware* pada perangkat IoT adalah akses fisik, adanya *firmware* edisi baru, dan perangkat IoT dalam satu sistem yang berada dalam jaringan internet yang berbeda. Maka diperlukan sistem *update firmware* perangkat IoT tanpa terkendala akses fisik dan dapat berjalan otomatis setiap ada *firmware* versi baru, dan dapat dijalankan pada perangkat IoT yang berada dalam jaringan berbeda.

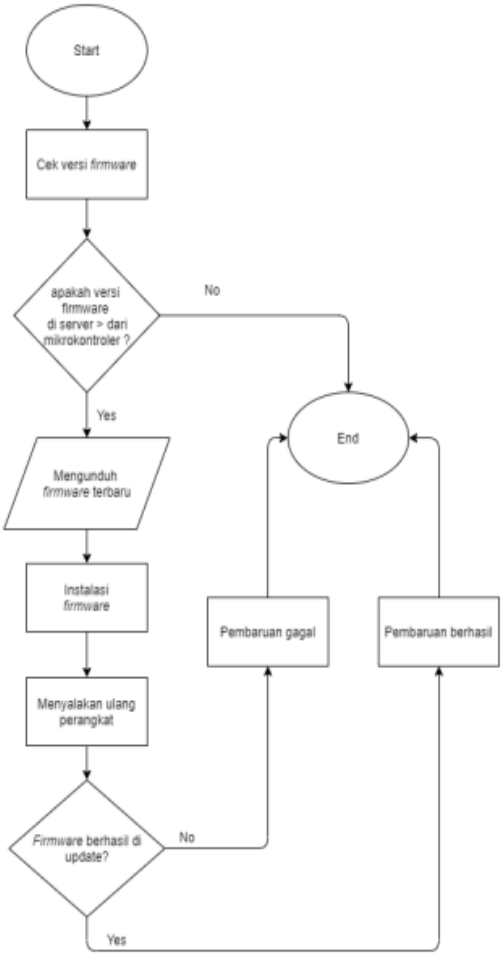
Setelah masalah teridentifikasi, dilakukan analisis kebutuhan fungsional sistem dan perangkat keras dan perangkat lunak ynng dibutuhkan untuk membangun sistem.

Sistem ini dirancang agar *user* memiliki akses untuk mengunggah *firmware* ke *Server* kemudian mikrokontroler sebagai perangkat IoT akan melakukan pengecekan secara otomatis dan mengirim versi *firmware* yang aktif ke *Server* seperti pada Gambar 2. *User* dalam sistem ini adalah administrator sistem IoT yang terdaftar pada sistem.



Gambar 2. Desain sistem

Sistem *auto update firmware* yang ditanam di perangkat IoT bekerja bedasarkan *flowchart* pada Gambar 3.



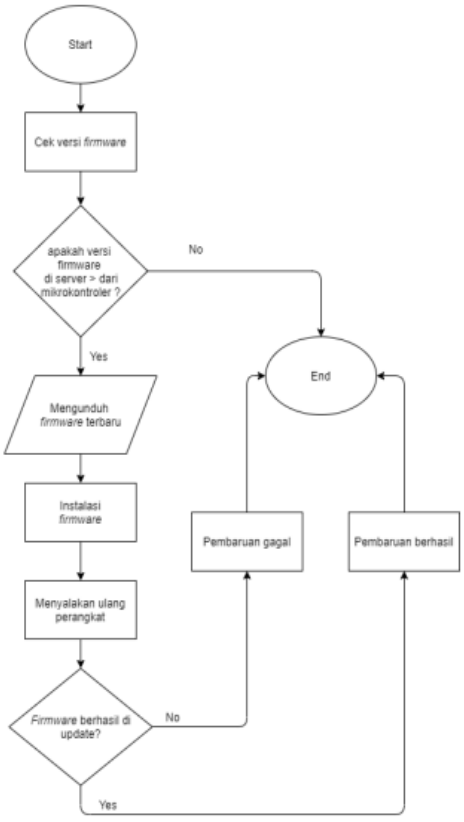
Gambar 3. *Flowchart* sistem pembaruan *firmware*

Sistem mulai bekerja dengan melakukan pengecekan versi pada *firmware* yang sedang berjalan

pada perangkat. Selanjutnya, sistem akan mengecek versi *firmware* pada *Server*. Jika versi *firmware* pada *server* lebih tinggi dari versi *firmware* yang sedang berjalan pada perangkat, maka perangkat akan mengunduh *firmware* yang ada di *Server*, menginstal *firmware* baru tersebut, dan melakukan *restart* untuk menjalankan *firmware* yang telah diinstalasi. *User* memiliki akses untuk mengunggah *firmware* ke *server* kemudian perangkat IoT akan melakukan pengecekan secara kontinyu.Untuk keamanan sistem, pada aplikasi IoT manajemen diterapkan metode *login* untuk memastikan hanya *user* yang terdaftar pada sistem yang dapat mengunggah *firmware*.

Pada tahap selanjutnya dilakukan instalasi dan konfigurasi sistem mulai dari pengadaan perangkat keras yang dibutuhkan sistem, konfigurasi, pengkodean script, *compile* dan *upload* hingga pengujian sistem sesuai dengan kebutuhan. Adapun alur dari tahap ini seperti yang digambarkan pada Gambar 4.

Pengujian sistem ini dilakukan dalam dua skenario pengujian. Skenario pengujian pertama merupakan pengujian keberhasilan pembaruan firmware OTA secara otomatis. Sedangkan pada pengujian kedua dilaksanakan dengan skenario menguji kemampuan *Server* dalam mendsitribusikan *firmware* kepada setiap *request* yang dilakukan oleh beberapa perangkat IoT dalam waktu yang sama pada jaringan yang berbeda.



Gambar 4. *Flowchart* Instalasi dan Konfigurasi

III. HASIL DAN PEMBAHASAN

Sistem *Auto-Update Firmware* melalui OTA ini terdiri atas dua buah perangkat lunak yaitu aplikasi web untuk manajemen perangkat IoT dari sisi server dan perangkat lunak sistem *update firmware over-the-air* yang di tanam pada perangkat IoT. Perangkat IoT yang digunakan adalah mikrokontroller ESP8266 dan ESP32.

Aplikasi web dibuat sebagai *interface* bagi *user* yaitu admin IoT untuk mengatur perangkat IoT seperti menambah perangkat, mengunggah *firmware*, dan menampilkan status perangkat. *Firmware* yang akan diunggah harus di *compile* kedalam bentuk *binary* (.bin) terlebih dahulu.

Pada aplikasi web ini terdapat halaman login:

a. Halaman login
b. Halaman *dashboard*
c. Halaman *firmware*
d. Halaman *user*

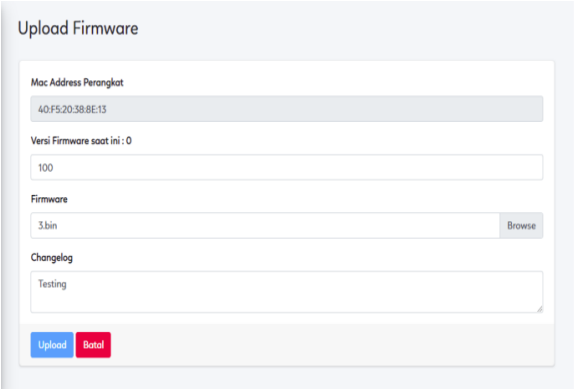
Pada seluruh perangkat IoT yang tergabung dalam satu sistem di tanam perangkat lunak sistem *update firmware over-the-air*. Proses *embeded* sistem ke perangkat IoT yang diterapkan meliputi:

1. Inisiasi library
   Inisialisasi Library merupakan proses pendeklarasikan Library yang digunakan.
2. Inisialisasi Variabel
   Inisialisasi variabel yang diterapkan pada ESP8266 sama dengan yang diterapkan pada ESP32. Inisialisasi digunakan untuk menyimpan nilai dari versi *firmware*, penentuan waktu eksekusi pengecekan *update firmware*, menyimpan kembalian dari fungsi kedalam variabel global dan dilakukan penentuan tipe data yang akan digunakan pada variabel.
3. Fungsi mengambil MAC *Address* Perangkat
   Fungsi ini untuk mengambil MAC address perangkat IoT lalu menyimpannya kedalam variabel "VarEspMAC".
4. Fungsi Mengirim versi *Firmware* ke *Server*
   Fungsi untuk mengirim versi *firmware* yang sedang berjalan/aktif pada perangkat IoT ke *database Server*.
5. Fungsi Pengecekan Alamat *Firmware*
   Setiap perangkat IoT menggunakan *firmware* yang berbeda, maka penamaan *file firmware* ditentukan berdasarkan MAC *address* perangkat masing-masing. Fungsi pengecekan alamat *firmware* merupakan fungsi yang mengatur perangkat agar mengakses dan mengunduh *firmware* berdasarkan MAC *address*-nya. Setelah alamat *firmware* diperoleh, nilainya akan disimpan kedalam variabel global untuk digunakan pada fungsi selanjutnya.
6. Fungsi *Update Over-The-Air*
   Dengan fungsi ini, perangkat akan melakukan pengecekan *firmware* sesuai dengan alamat yang sudah ditentukan (MAC *address*). Perangkat akan mencoba terhubung ke *server*, kemudian

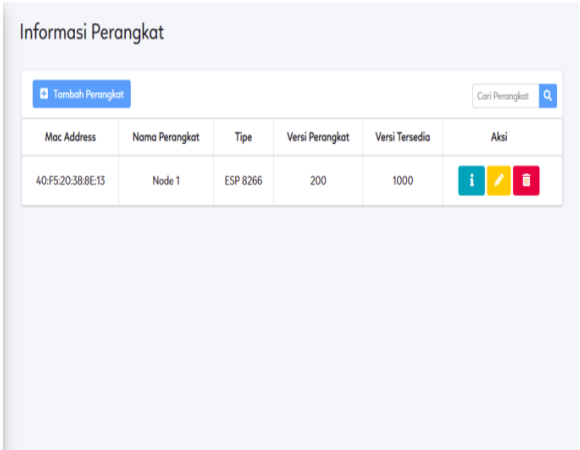
memperoleh *file* versi *firmware* yang tersedia untuk dibandingkan dengan versi *firmware* yang sedang berjalan. Apabila versi yang tersedia lebih tinggi maka akan dilakukan proses instalasi *firmware* ke dalam *eboot* perangkat dan menggantikan *boot* perangkat ketika perangkat melakukan *reboot*.

7. *Set up* Wifi dan *Timer Milis*
   Set up wifi dilakukan agar perangkat dapat terhubung ke server melalui internet. Sedangkan *set up timer milis* dilakukan untuk mengatur waktu fungsi *update firmware* OTA dieksekusi.

Pengujian skenario 1 adalah pengujian *upload* dan *update firmware* pada perangkat IoT melalui OTA. Perangkat IoT yang telah terdaftar dalam sistem Manajemen IoT dan telah ditanam perangkat lunak sistem *update firmware over-the-air*. Admin IoT dapat mengunggah *firmware* versi terbaru ke sistem manajemen IoT seperti pada Gambar 5.

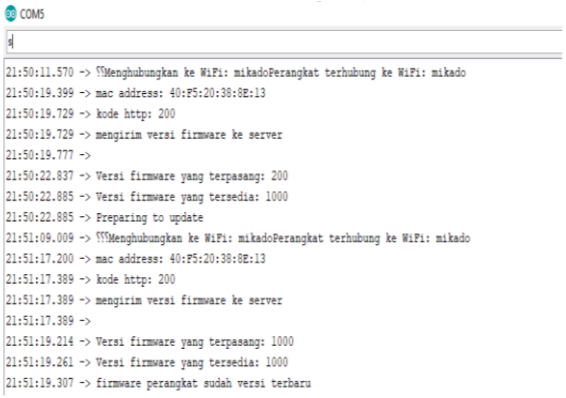


Gambar 5. Halaman *Upload Firmware*



Gambar 6. *Firmware* versi terbaru telah di unggah

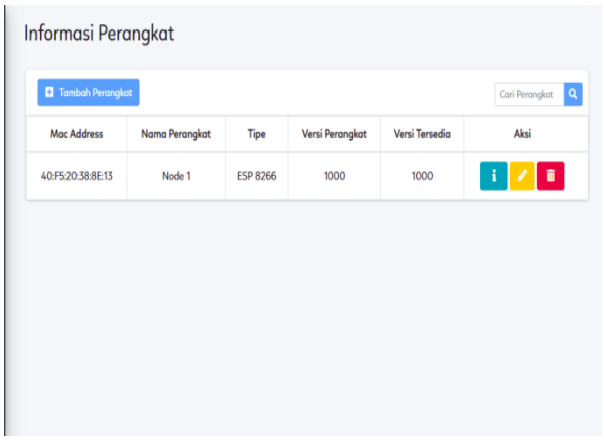
Pada Gambar 6 menunjukkan bahwa firmware telah berhasil di *upload*. Saat perangkat IoT aktif dan tehubung ke wifi yang memiliki akses ke internet maka secara otomatis perangkat IoT tersebut akan melakukan pengecekan versi *firmware* yang tersedia pada *server* kemudian membandingkannya dengan versi *firmware* miliknya. Script untuk proses ini terlihat pada Gambar 7.



Gambar 7. Script pengecekan versi dan *update firmware*

Jika versi *firmware* yang tersedia pada server lebih tinggi daripada versi *firmware* yang berjalan pada perangkat IoT terdaftar maka perangkat akan mengunduh *firmware* pada *server* dan melakukan *reboot* untuk menjalankan *firmware* versi baru yang telah dipasang. Jika versi *firmware* pada perangkat dan *server* sama maka perangkat akan mengirim pesan bahwa perangkat menggunakan *firmware* versi terbaru.

Saat *update firmware* berhasil dilakukan maka perangkat akan mengirim kembali versi *firmware*-nya ke server seperti yang ditunjukkan pada Gambar 8.



Gambar 8. Pembaruan *Firmware* berhasil

Penelitian yang dilakukan oleh [7] dan [8] melakukan *update firmware* pada perangkat yang berada dalam jaringan yang sama. Pengujian skenario 2 dilakukan pengujian *update firmware* melalui OTA pada enam buah mikrokontroller sebagai perangkat IoT yang terhubung ke internet melalui jaringan yang berbeda dalam waktu yang bersaman. Pada pengujian ini, kondisi awal *firmware* perangkat IoT hanya berisi fungsi *update*, sedangkan kondisi akhir perangkat mikrokontroler mampu mengirim versi *firmware* yang terpasang ke server.

Hasil pengujian *update firmware* pada Tabel 1 menunjukkan bahwa *server* mampu mendistribusikan *firmware* dan berhasil diinstal di enam buah perangkat

IoT dengan kondisi perangkat tersebut masing-masing terhubung ke internet melalui jaringan Wifi yang berbeda. Rata-rata waktu yang diperlukan perangkat untuk mendeteksi versi *firmware* hingga berhasil melakukan *update firmware* adalah 78,49 detik.

Tabel 1. Pengujian Pembaruan *Firmware*

MAC Address	Jenis Mikrokontroler	Waktu *Update* (d)	Status
9C:9C:1F:CB;20:94	ESP 32	80.01	Berhasil
84.CC:A8:A5:22:7D	ESP 8266	76.92	Berhasil
84.CC:A8:A5:22:BC	ESP 8266	81.07	Berhasil
40:F5:20:34:C6:84	ESP 8266	77.31	Berhasil
84:F3:EB:C8:E5:8E	ESP 8266	77.54	Berhasil
40:F5:20:38:09:54	ESP 8266	78.12	Berhasil

Selain itu, pengujian kinerja Server juga dilakukan dengan menggunakan Apache Jmeter. Pengujian ini dilakukan dengan melakukan *request* pada HTTP dengan jumlah yang besar secara bersamaan ke *server*. Pengujian ini bertujuan untuk mengetahui seberapa banyak perangkat IoT yang bisa dilayani oleh *server* dalam waktu yang bersamaan.

Tabel 2.Pengujian kinerja respon Server menggunakan Apache Jmeter

Thread (users)	*Throughput* (d)	*Max Eplased Time Result* (md)	Error %
10	6.9	630	0.00
100	64.3	599	0.00
1000	458.1	1524	0.00
10000	411.5	21051	0.38
100000	646.3	34779	70.36

Hasil pengujian kinerja respon *server* ditunjukkan pada Tabel 2. Data yang didapatkan menunjukkan bahwa server mampu mendistribusikan *firmware* ke beberapa mikrokontroler dalam waktu yang bersamaan. Pengujian *request* dilakukan dengan percobaan *server* melayani 10 hingga 100.000 *request* dari perangkat dalam waktu bersamaan. Pada data di Tabel 2 terlihat bahwa *Error* terjadi saat *server* melayani 10.000 dan 100.000 *request* dengan nilai *error* masing-masing 0.38% dan 70,36%. Hal ini berarti sistem dapat melayani hingga sekitar 29.000 *request* secara bersamaan.

IV. KESIMPULAN

Penelitian ini berhasil membangun sistem pembaruan otomatis *firmware over-the-air* untuk perangkat IoT. Sistem ini juga berhasil melakukan *update firmware* untuk beberapa perangkat IoT dalam waktu bersamaan walaupun dalam jaringan berbeda. Sistem dapat melayani hingga 29.000 *request* secara bersamaan.

Sistem ini merupakan bagian dari sistem manajemen IoT untuk lingkungan pertanian. Untuk tahap berikutnya akan di buat sistem visualisasi data untuk menampilkan kondisi semua perangkat yang terhubung dalam sistem IoT ini.

REFERENSI

[1] A. S. A. Quadri and B. O. Sidek, "An Introduction to Over-the-Air Programming in Wireless Sensor Networks," *Int J Comput Sci Netw Solut [Internet]*, vol. 2, pp. 33–49, 2014.

[2] S. Ramadhan, A. S. Budi, and M. H. H. Ichsan, "Rancang Bangun Sistem Auto-Config Sensor Baru pada Perangkat IoT secara Over-The-Air menggunakan Protokol HTTP berbasis Raspberry-Pi," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. e-ISSN*, vol. 2548, p. 964X.

[3] J. Bauwens, P. Ruckebusch, S. Giannoulis, I. Moerman, and E. De Poorter, "Over-the-air software updates in the internet of things: An overview of key principles," *IEEE Commun. Mag.*, vol. 58, no. 2, pp. 35–41, 2020.

[4] L. Hakim, W. A. Kusuma, and M. Faiqurahman, "Over The Air Update Firmware pada Perangkat IoT Dengan Protokol MQTT," *J. Sist. dan Inform.*, vol. 14, no. 2, pp. 99–105, 2020.

[5] A. K. Hananta, M. A. Murti, and N. Prihatiningrum, "Perancangan Sistem Untuk Update File Firmware Iot Menggunakan Over The Air Update," *eProceedings Eng.*, vol. 9, no. 2, 2022.

[6] C. Gao, L. Luo, Y. Zhang, B. Pearson, and X. Fu, "Microcontroller based IoT system firmware security: Case studies," in *2019 IEEE International Conference on Industrial Internet (ICII)*, 2019, pp. 200–209.

[7] I. G. N. D. Paramartha, I. N. H. Kurniawan, G. B. Subiksa, and A. S. Kartika, "Arsitektur Internet of Things (IoT) Berskala Industri dengan fitur Over The Air Update," *TIERS Inf. Technol. J.*, vol. 2, no. 2, pp. 31–36, 2021.

[8] O. B. Pratama, A. Bhawiyuga, and K. Amron, "Pengembangan Perangkat Lunak IoT Cloud Platform Berbasis Protokol Komunikasi HTTP," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. e-ISSN*, vol. 2548, no. 9, p. 964X, 2018.

[9] D. Frisch, S. Reißmann, and C. Pape, "An over the air update mechanism for ESP8266 microcontrollers," in *Proceedings of the ICSNC, the Twelfth International Conference on Systems and Networks Communications, Athens, Greece*, 2017, pp. 8–12.

[10] S. Supriyanto, W. A. Kusuma, and M. Faiqurahman, "Aplikasi Berbasis Website sebagai Interface untuk Over The Air Update Firmware pada Perangkat Io T," 2019.