KOMPRESI DAN PENGIRIMAN DATA ABSENSI SIDIK JARI DENGAN PENGEMBANGAN ALGORITME LOSSLESS RAKE

Kartika Sari
STMIK Pontianak, Jl. Merdeka No.372, Pontianak.
e-mail: *[1]krtqsr@gmail.com.

Abstrak

Dengan adanya keterbatasan penyimpanan data pada perangkat absensi sidik jari, maka diperlukan adanya integrasi antara cloud dan perangkat dengan cara melakukan pengiriman data. Namun, pada proses pengiriman data secara langsung masih memiliki kelemahan, yaitu semakin besar ukuran data yang dikirimkan, pengiriman data menjadi relatif lebih lambat, begitu pula dengan akses informasi. Sehingga dibutuhkan metode untuk memangkas penggunaan memori dan waktu pengiriman data.Penelitian ini mencoba menerapkan salah satu metode kompresi data untuk mengurangi waktu pengiriman data dan mengurangi memori dengan menggunakan pengembangan dari algoritme kompresi data lossless RAKE. Data perangkat dikonversi dengan menggunakan bilangan biner 0 dan 1. Parameter yang digunakan pada penelitian ini adalah efisiensi kompresi dan waktu pengiriman data.Berdasarkan pengujian, waktu rata-rata yang dibutuhkan untuk pengiriman data absensi sidik jari secara langsung adalah 0,837 detik, dan dengan menggunakan metode kompresi data RAKE yang telah dimodifikasi sebesar 0,636 detik dengan selisih waktu sebesar 0,2014 detik per data yang dikirim. Sedangkan untuk pengukuran kompresi data RAKE didapatkan hasil efisiensi kompresi rata-rata diatas 50% dan untuk dekompresi data 100% data berhasil kembali seperti data semula. Semakin banyak biner '0' pada bilangan biner sidik jari, maka efisiensi kompresi akan semakin besar, sehingga data yang terkompresi akan semakin banyak.

Kata kunci— Algoritme RAKE, Kompresi Data, Pengiriman Data, Cloud Computing

Abstract

The limitations of fingerprint attendance machine makes it is necessary to integrate the cloud and the device by sending data. However, the process of sending data directly still has weaknesses, the greater the size of the data sent, the slower data transmission become, as well as access to information. So, we need a method to cut memory usage and data transmission times.This research apply the development of the lossless data compression algorithm RAKE to reduce data transmission time and reduce memory. The device data is converted using binary numbers 0 and 1. The parameters used in this study are the efficiency of compression and time of sending data.Based on testing, the average time required for sending fingerprint attendance data directly is 0.837s and using the modified RAKE data compression method of 0.636s with a time difference of 0.2014s per data sent. The measurement of data compression using modified RAKE got above 50% in compression efficiency and got 100% for the data decompression it's mean all the data returns to the original data. The more binary '0' in the fingerprint binary number, the greater the number of compression efficiency, so that more compressed data will be generated.

Keywords—RAKE Algorithm, Data Compressiomn, Data Transmittion, Cloud Computing

1. PENDAHULUAN

Dengan keterbatasan yang ada pada sebuah perangkat absensi sidik jari, salah satu solusi yang dapat diberikan yaitu memindahkan proses penyimpanan dan komputasi ke sebuah sistem yang lain, contohnya *cloud computing platform* sebagai lingkungan virtual yang umumnya memiliki kapasitas penyimpanan dan komputasi yang sangat besar [1]. Untuk melakukan integrasi antara *cloud* dengan IoT, perangkat IoT diharuskan untuk melakukan pengiriman data dari perangkat IoT ke cloud, dengan tujuan untuk mengalihkan proses komputasi dan penyimpanan menuju cloud [1]. Adapun tujuan utama pengiriman data adalah untuk menurunkan lama pengiriman paket data dalam komunikasi. Oleh karena itu, dalam kondisi jaringan dan kumpulan data yang sama, algoritme yang dapat melakukan pengiriman data dan menerima paket data dengan waktu yang lebih pendek selama pengiriman data dianggap lebih baik [2].

Untuk mengurangi kapasitas dalam penyimpanan data, umumnya digunakan teknik untuk memampatkan data [3,4] serta untuk meningkatkan masa pakai perangkat bertenaga baterai [5]. Algoritme yang dapat digunakan untuk kompresi data IoT adalah algoritme kompresi lossless RAKE, hal ini dikarenakan algoritme lossless lebih dapat digunakan dalam beberapa skenario IoT [6] dan meskipun beberapa algoritme kompresi lossless lain telah tersedia, misalnya algoritme Lempel-Ziv [7], sebagian besar tidak cocok pada keterbatasan penyimpanan dan sumber daya komputasi [8]. Selain itu, algoritme RAKE ini hanya membutuhkan operasi perhitungan dasar, overhead terbatas (hanya menggunakan operasi sederhana dan konversi bit pada operasi), penyimpanan terpusat (tidak bergantung pada codewords/template sebelumnya) serta algoritme ini dapat dengan mudah diperluas ke urutan bilangan bulat [5].

Proses pengiriman secara langsung masih memiliki kelemahan, yaitu semakin banyak dan besar ukuran data yang dikirimkan, maka pengiriman data ke cloud menjadi relatif lebih lambat, sehingga akses informasi menjadi tidak cepat sampai. Solusi yang dapat dilakukan adalah memperkecil ukuran (kompresi) data tersebut [7]. Untuk itu, dibutuhkan adanya metode untuk mengurangi konsumsi waktu pada proses pengiriman data. Salah satu poin penting untuk mengurangi konsumsi waktu pada proses pengiriman data adalah dengan cara mengurangi penggunaan memori. Berdasarkan pemaparan diatas, pada penelitian ini akan dilakukan implementasi dari pengembangan algoritme kompresi data lossless RAKE, untuk mendapatkan nilai kompresi data, dekompresi data serta waktu pengiriman data. Penelitian ini bertujuan untuk mengurangi waktu pengiriman data dan mengurangi penggunaan memori.

2. METODE PENELITIAN

2.1 Pengumpulan Data

Data yang digunakan pada penelitian ini merupakan data-data yang berfungsi sebagai masukan pada sistem yang dibangun. Data tersebut kemudian akan diolah dan digunakan lebih lanjut untuk mendapatkan output yang diinginkan. Pada tahap ini, data yang dikumpulkan berupa:

a. Nilai data sidik jari yang akan dikompresi, berupa data dari perangkat absensi sidik jari.
b. Ukuran dari nilai data sensor sebelum dan setelah kompresi

2.2 Algoritme Kompresi Data Berdasarkan RAKE

Algoritme kompresi data RAKE merupakan sebuah algoritme kompresi data dengan teknik loseless yang hanya efektif pada string biner sparse. Untuk sinyal dengan variasi waktu yang kecil, kompresi dapat dilakukan dengan cara menerapkan algoritme ke perbedaan sampel yang berurutan, yaitu residu $r(i) = x(i) – x(i-1)$.

Algoritme RAKE dapat digunakan jika nilai 1 pada data yang akan dikompresi kurang dari 15% dari jumlah bit. Namun, jika angka 1 yang dimiliki lebih dari 40% dari bit yang ditetapkan, kompresi dengan RAKE tidak dapat dioperasikan. Berikut merupakan aturan pada algoritme RAKE [5]:

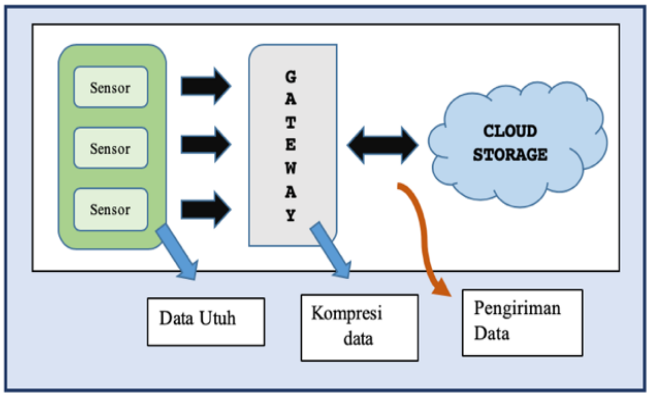
a. Codeword bernilai ‘0’ menyatakan bahwa semua bit pada RAKE bernilai ‘0’, yang artinya tidak ada set-bit (bit bernilai ‘1’) yang ditemukan pada RAKE.
b. Codeword bernilai L = 1 + ⌈log2 T⌉ - bit menyatakan bahwa terdapat setidaknya satu set-bit pada RAKE. Bit pertama dari codeword diatur menjadi bit bernilai ‘1’ untuk menyatakan bahwa ada satu set-bit telah ditemukan pada RAKE dan ⌈ log2 T ⌉ bit lainnya digunakan untuk mengkodekan posisi Pfirst-nya. Pfirst merupakan posisi ke- (n-1) dari set bit ‘1’. Posisi Pfirst dihitung mulai dari 0 sampai T-1.
c. Setelah didapatkan nilai codeword, RAKE akan bergeser sebanyak nilai (Pfirst + 1) saat set-bit ditemukan atau RAKE akan bergeser sebanyak T saat set bit tidak ditemukan.
d. Operasi tersebut diulang sampai RAKE mencapai semua bit yang akan dikompresi.
e. Urutan kompresi pada RAKE diperoleh dengan cara menggabungkan semua codeword.

Untuk dekompresi data, urutan biner asli dapat dipulihkan dari urutan terkompresi dengan cara sebagai berikut:

a. Setiap codeword ‘0’ diterjemahkan sebagai nilai ‘0’ sebanyak T.
b. Setiap kali bit 1 dibaca, maka bit ke L – 1 berikutnya dicek untuk mendapatkan nilai x. Nilai x merupakan representasi desimal dari bilangan biner yang dikodekan dalam bit ke (L – 1). Kemudian, nilai x diubah menjadi bit bernilai ‘0’ dan diikuti oleh bit bernilai ‘1’ dibelakangnya.
c. Operasi a dan b diulang hingga mencapai akhir urutan bit yang akan di-dekompresi.
d. Urutan dekompresi diperoleh dengan menggabungkan hasil pembacaan/terjemahan bit kompresi.

2.3 Sistem Keseluruhan

Fokus penelitian ini adalah pada subsistem gateway (perangkat absensi sidik jari) dan subsistem platform cloud (Digital Ocean Server). Gambar 1 merupakan diagram umum dari sistem yang akan dibangun:

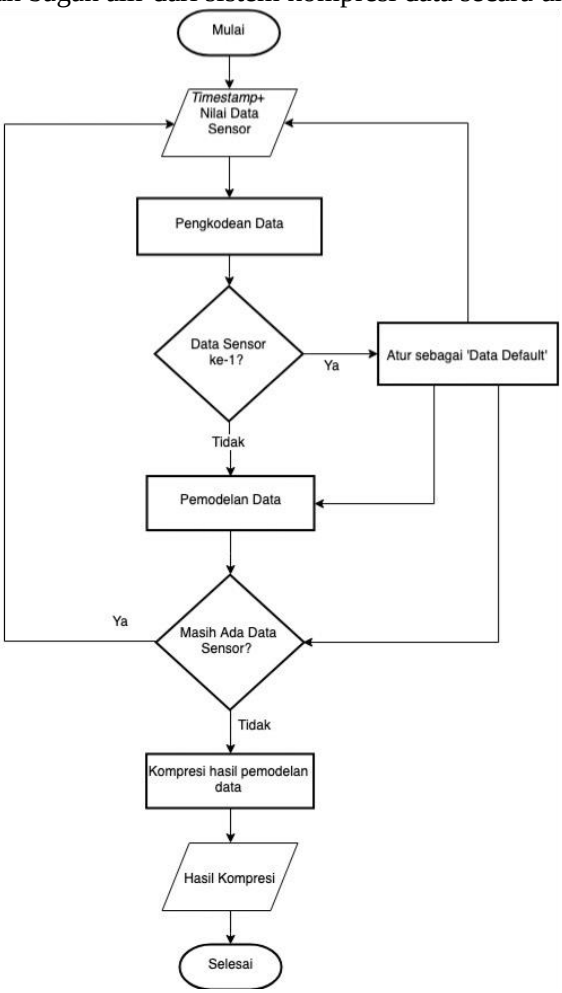


Gambar 1 Diagram Sistem secara Umum

Secara umum, terdapat beberapa node sensor pada sistem ini yang berfungsi untuk mengumpulkan data-data yang diambil dari perangkat absensi sidik jari. Sensor perangkat absensi sidik jari terhubung dengan gateway pada sisi client, memberikan nilai yang diperoleh dari semua sensor secara kondisional (terus menerus dan atau sesuai dengan waktu yang telah ditentukan). Gateway pada sistem bertugas untuk mengelola perangkat, menerima data dan kemudian melakukan kompresi data sebelum data dikirimkan ke cloud. Sedangkan *platform cloud* bertanggungjawab untuk menyimpan data sidik jari menggunakan interface layanan yang telah disediakan oleh gateway sehingga pengguna dapat menyimpan data perangkat serta melakukan dekompresi data dengan menggunakan pengembangan algoritme RAKE.

2.4 Sistem Kompresi dan Dekompresi Data secara Umum

Gambar 2 merupakan bagan alir dari sistem kompresi data secara umum [9].

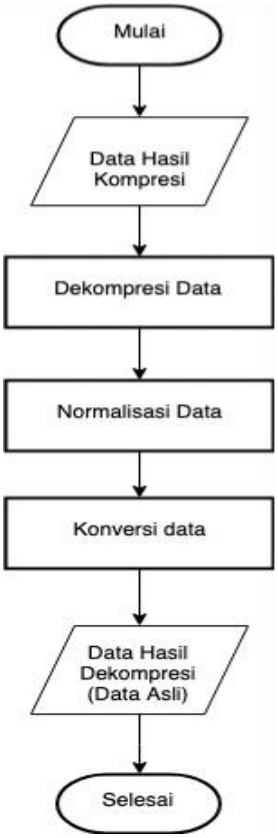


Gambar 2 Bagan Alir Sistem Kompresi Data secara Umum

Penjelasan dari bagan alir kompresi data secara umum adalah sebagai berikut [9]:

a. Langkah pertama yang dilakukan adalah pengambilan timestamp dan pengambilai nilai dari paket data sensor sesuai dengan waktu pengambilan data.
b. Kemudian dilakukan proses pengkodean data, yaitu dengan cara mengkonversi nilai data sensor ke dalam bentuk biner ‘1’ dan biner ‘0’.
c. Setelah itu dilakukan pengecekan apakah data yang diambil tersebut merupakan data pertama. Jika ya, maka data tersebut akan diatur menjadi data default dan disimpan sementara, data tersebut nantinya akan digunakan dalam proses pemodelan data untuk melakukan kompresi dan dekompresi data.
d. Jika nilai yang diambil tersebut bukan merupakan data pertama (data ke n + 1), maka akan diteruskan ke proses pemodelan data.
e. Langkah selanjutnya adalah pengecekan apakah masih ada data sensor setelahnya yang harus diambil. Jika tidak, maka akan dilanjutkan ke proses kompresi hasil dari pemodelan data. Jika masih ada data sensor, maka akan dilanjutkan ke proses pengambilan nilai timestamp dan pengambilan nilai data sensor.
f. Selanjutnya dilakukan kompresi terhadap bilangan biner hasil dari proses pemodelan tersebut dengan menggunakan algoritme RAKE.
g. Setelah nilai data sensor berhasil dikompresi, dilakukan pengiriman data hasil kompresi tersebut ke cloud. Kemudian akan dilakukan proses dekompresi data pada cloud, agar data tersebut dapat kembali utuh dan dapat digunakan/ditampilkan.

Proses dekompresi data meliputi proses normalisasi data dan proses konversi data ke bentuk data semula. Bagan alir dekompresi data secara umum dapat dilihat pada Gambar 3 [9]:



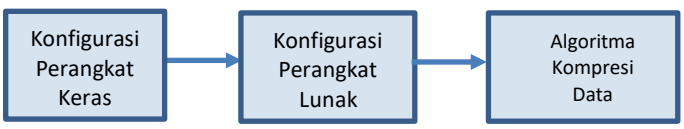
Gambar 3 Bagan Alir Dekompresi Data Secara Umum

Penjelasan bagan alir dekompresi data adalah sebagai berikut:

a. Pertama-tama dilakukan proses pengambilan data hasil kompresi pada platform *cloud.*
b. *Platform Cloud* melakukan dekompresi terhadap data hasil kompresi.
c. Setelah itu dilakukan normalisasi terhadap data hasil dekompresi data.
d. Tahap selanjutnya adalah konversi data, yaitu dengan cara mengubah nilai biner menjadi nilai data asli (data hasil dekompresi).

2.5 *Pengujian Data*

Selanjutnya dilakukan implementasi pengembangan algoritme kompresi dan dekompresi data sesuai dengan perancangan yang telah dilakukan untuk uji kinerja sistem. Tahap pengujian ini dilakukan untuk menentukan keberhasilan dari penelitian ini. Alur konfigurasi sebelum melakukan pengujian secara umum ditunjukkan pada gambar 4:

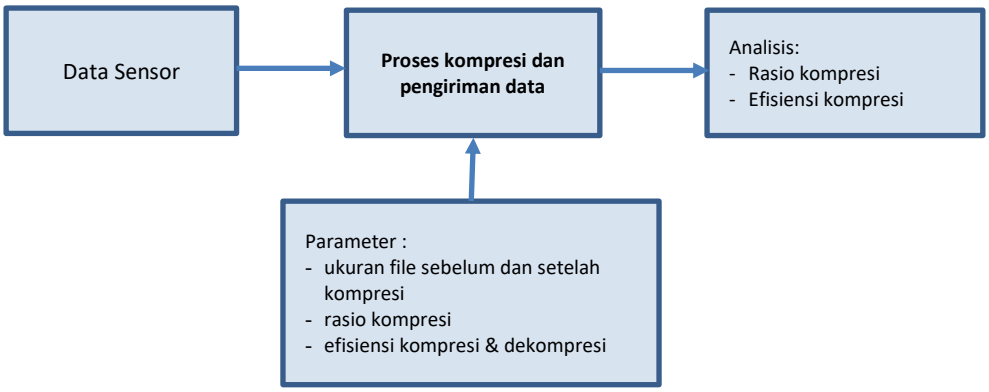


Gambar 4 Alur Perancangan Pengujian secara Umum

Gambar 4 menjelaskan alur perancangan sistem pengujian secara umum, dimulai dari konfigurasi perangkat keras yaitu Raspberry Pi sebagai client untuk melakukan kompresi data

dan melakukan pengiriman data ke cloud server. Setelah konfigurasi perangkat keras selesai dilakukan, kemudian dilakukan konfigurasi perangkat lunak yang terdiri dari sistem operasi Raspian, python digunakan sebagai bahasa pemrograman untuk Raspberry dan JSON untuk cloud. Umumnya suatu sistem pengujian harus memiliki suatu masukan, proses dan keluaran. Tahapan pengujian dimulai dengan masukan data sensor ke sistem sebagai bahan untuk menganalisis kinerja. Proses terdiri dari proses kompresi-dekompresi data dan proses pengiriman data. Output yaitu analisis data yang didapat dari hasil kompresi dan pengiriman data dari gateway ke platform.

Pada tahap ini, hasil pengujian akan dianalisis dengan menggunakan data yang didapat dari tahap evaluasi dari teknik pengkodean dan pemodelan data dan kaitannya dengan rasio kompresi, efisiensi kompresi dan efisiensi dekompresi. Analisis pada tahap ini bertujuan untuk mengetahui pengaruh dari teknik pengkodean dan pemodelan untuk data yang akan dikompresi terhadap besarnya nilai efisiensi kompresi. Dari analisis ini peneliti dapat mengetahui rata-rata uji kinerja dari masing-masing parameter yang akan diujikan. Gambar 2 merupakan diagram blok pengujian sistem.



Gambar 5 Diagram blok pengujian

Hasil pengujian akan dianalisis dengan menggunakan data yang didapat dari parameter efisiensi kompresi dan dekompresi data. Dari analisis ini peneliti dapat mengetahui uji kinerja dari masing-masing parameter.

a. Pengujian kinerja algoritme kompresi data pada file uji.

Rasio kompresi didefinisikan sebagai rasio antara jumlah bit sebelum kompresi dan sesudah kompresi [9]. Perhitungan rasio kompresi ditunjukkan pada persamaan (1).

$$Rasio\ Kompresi = \frac{Ukuran\ file\ SEBELUM\ kompresi}{Ukuran\ file\ SETELAH\ kompresi} \quad (1)$$

Sedangkan efisiensi kompresi (EK%) ditunjukkan pada persamaan (2).

$$EK\% = 100 \times (1 - \frac{1}{RK}) \quad (2)$$

Efisiensi kompresi disajikan dalam bentuk presentase yang menggambarkan suatu ukuran keberhasilan dari kompresi data.

b. Pengujian kinerja algoritme dekompresi data pada file uji.

Rasio dekompresi didefinisikan sebagai rasio antara jumlah bit sebelum dan sesudah dekompresi [9]. Perhitungan rasio dekompresi ditunjukkan pada persamaan (3).

Efisiensi dekompresi (ED%) ditunjukkan pada persamaan (4).

$$ED\% = 100 \times (1 - \frac{1}{RD}) \quad (4)$$

Efisiensi dekompresi disajikan dalam bentuk presentase yang menggambarkan suatu ukuran keberhasilan dari dekompresi data.

3. HASIL DAN PEMBAHASAN

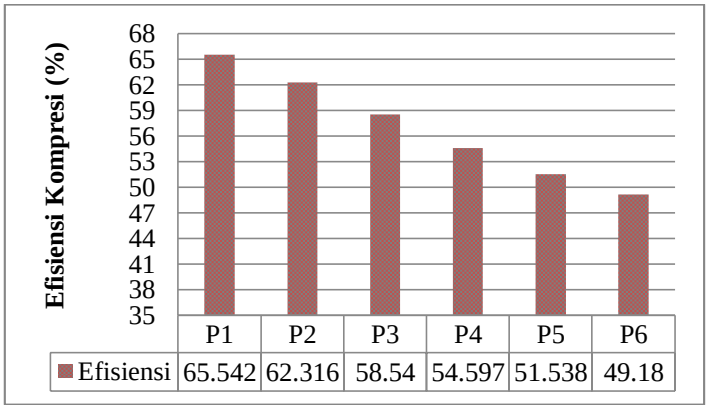
3.1 Parameter Efisiensi Kompresi

Parameter efisiensi kompresi didapatkan dari perhitungan rasio kompresi dengan mempertimbangkan variabel ukuran data sebelum kompresi dan setelah kompresi. Hasil dari pengujian kompresi data menggunakan beberapa metode yang menggunakan teknik pengkodean data dan teknik pemodelan data tersebut dilampirkan pada bab lampiran. Pada penelitian ini, dilakukan pengambilan nilai data sensor dengan menggunakan 6 periode waktu yang berbeda untuk setiap percobaan pengujian untuk mendapatkan ukuran nilai data sensor yang berbeda, sehingga dapat diketahui pengaruh besarnya ukuran data yang akan dikompresi terhadap parameter efisiensi kompresi data. Masing-masing periode dilakukan sebanyak 70 kali percobaan pengujian dengan tujuan agar hasil dari kesimpulan penelitian dapat digeneralisasikan untuk seluruh data sensor yang dikompresi. Percobaan pengujian kompresi data dilakukan dengan menggunakan pengembangan dari algoritme kompresi data RAKE untuk mengetahui perbedaan nilai parameter efisiensi kompresi untuk masing-masing metode. Tabel 1 merupakan rangkuman pengujian kompresi data.

Tabel 1 Rangkuman Pengujian Kompresi Data

PERCOBAAN KE- (Pn)							
Waktu Pengambilan data (Detik)		P1 (10)	P2 (30)	P3 (60)	P4 (120)	P5(180)	P6 (240)
Ukuran Data Sebelum Kompresi (byte)		1096	2328	4152	7800	11448	15096
Ukuran Data Setelah Kompresi (byte)	MIN	365	857	1690	3469	5398	7552
	MAX	390	899	1761	3609	5699	7790
	AVERAGE	377,657	877,286	1721,429	3541,471	5547,93	7671,757
RASIO KOMPRESI		2,903	2,654	2,412	2,203	2,064	1,968
EFISIENSI KOMPRESI (%)		65,542	62,316	58,54	54,597	51,538	49,180

Dari hasil pengujian kompresi data, didapatkan nilai ukuran data sebelum kompresi dan ukuran data setelah kompresi, nilai rasio kompresi, serta nilai presentasi kompresi. Grafik pada gambar 6 merupakan grafik efisiensi kompresi terhadap Panjang data.



Gambar 6 Grafik efisiensi Kompresi terhadap Panjang Data

3.2 *Parameter Efisiensi Dekompresi*

Pada pengujian dekompresi data, dilakukan dengan menggunakan pengembangan dari algoritme kompresi data RAKE. Pada masing-masing percobaan pengujian, dilakukan dengan menggunakan 6 periode waktu yang berbeda untuk mengetahui pengaruh panjang data yang diambil, untuk kemudian data tersebut dikompresi, dan selanjutnya di-dekompresi kembali, apakah data tersebut dapat kembali utuh seperti semula. Tabel 2 merupakan rangkuman hasil pengujian dekompresi data sensor dengan menggunakan algoritme *RAKE*.

Tabel 2 Rangkuman Pengujian Kompresi Data

Percobaan Ke-	Uk. Data Asli (byte)	Uk. Data Setelah Kompresi (byte)	Uk. Data setelah Dekompresi (byte)	Rasio Dekompresi (%)
P1	1096	377,657	1096	100
P2	2328	877,286	2328	100
P3	4152	1721,429	4152	100
P4	7800	3541,471	7800	100
P5	11448	5547,93	11448	100
P6	15096	7671,757	15096	100

Dari hasil pengujian dekompresi data, didapatkan nilai/ukuran panjang data sebelum dan setelah dekompresi tetap sama, dan tidak berubah. Kemudian pada saat dilakukan konversi data hasil kompresi, didapatkan nilai data sebelum kompresi dan nilai data setelah kompresi SAMA / UTUH untuk semua percobaan dekompresi yang telah dilakukan. Sehingga dapat disimpulkan bahwa 100% data berhasil kembali seperti semula secara utuh.

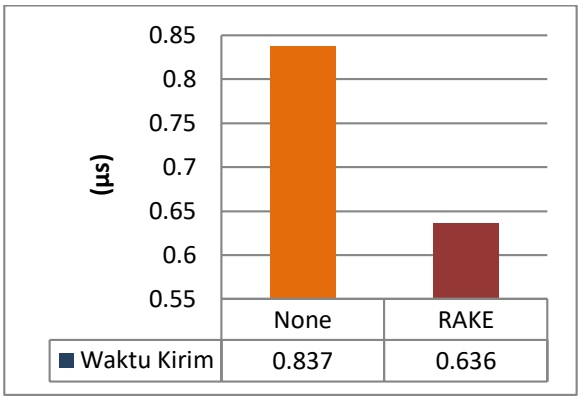
3.3 *Parameter Waktu Pengiriman Data*

Pada pengujian pengiriman data ini, dilakukan pengambilan data dengan parameter waktu pengiriman secara langsung, dan data dengan parameter waktu pengiriman menggunakan pengembangan dari algoritme kompresi data RAKE yang kemudian hasilnya akan dibandingkan. Hasil pengujian terhadap 70 data untuk 5 skenario percobaan pengujian pengiriman data. Tabel 3 merupakan tabel rangkuman pengujian pengiriman data.

Tabel 3 Hasil pengujian parameter waktu pengiriman data

	Parameter waktu pengiriman Data (detik)									
	Tanpa Kompresi/None (Detik)					**Dengan Kompresi RAKE (Detik)**				
	30	60	120	180	240	30	60	120	180	240
Min	0,281	0,328	0,400	0,992	1,642	0,199	0,237	0,374	0,688	1,458
Max	0,310	0,372	0,507	1,717	1,836	0,225	0,320	0,399	0,897	1,604
Average	0,309	0,351	0,456	1,348	1,736	0,212	0,273	0,387	0,790	1,518

Dari hasil pengujian pengiriman data, didapatkan parameter waktu pengiriman data secara langsung, dan parameter waktu pengiriman data menggunakan algoritme RAKE. Grafik pada gambar 7 merupakan grafik perbandingan waktu pengiriman data.



Gambar 7 Grafik perbandingan waktu pengiriman data

Berdasarkan grafik rangkuman perbandingan rata-rata waktu pengiriman data secara langsung dan dengan menggunakan metode kompresi data RAKE, didapatkan rata-rata waktu pengiriman data secara langsung yang waktu kirimnya menunjukkan angka 0,837 detik dan waktu pengiriman data ke *cloud* dengan menggunakan pengembangan metode kompresi data dengan menggunakan algoritme RAKE lebih cepat yaitu 0,636 detik. Sehingga dapat dikatakan bahwa pengiriman data menggunakan kompresi data RAKE lebih baik daripada pengiriman data secara langsung tanpa metode, hal ini dikarenakan data yang telah dikompresi memiliki ukuran mencapai setengah dari data asli.

4. KESIMPULAN

Hasil analisis pengujian menggunakan 5 skenario percobaaan dengan periode waktu dan ukuran data yang berbeda menunjukkan bahwa pengiriman data menggunakan metode kompresi data RAKE lebih baik daripada pengiriman data secara langsung tanpa kompresi. Berdasarkan hasil pengujian, nilai rata-rata waktu yang dibutuhkan untuk pengiriman data sensor secara langsung lebih banyak yaitu 0,837 detik, sedangkan waktu pengiriman data menggunakan metode kompresi data RAKE yaitu sebesar 0,636 detik dengan selisih rata-rata waktu pengiriman data yaitu sebesar 0,2014 detik.

Berdasarkan hasil analisis pengujian kompresi data RAKE dengan menggunakan 6 skenario percobaan, didapatkan nilai rata-rata efisiensi kompresi data RAKE diatas 50%, dan hasil dekompresi data 100%. Efisiensi kompresi akan meningkat jika biner ‘1’ pada set bitmap

kurang dari 15% dari set bitmap keseluruhan, semakin banyak biner '0' pada set bitmap, semakin banyak juga data yang dikompresi. Namun, jika biner '1' terdapat sebanyak 40% dari set bitmap, kompresi dengan menggunakan algoritme RAKE tidak dapat dilakukan.

5. SARAN

Saran yang perlu ditambahkan berdasarkan pada pengujian yang telah dilakukan adalah sebagai berikut:

1. Dapat ditambahkan uji kinerja lainnya selain dari parameter yang sudah ada, sehingga terdapat banyak parameter yang dapat dijadikan acuan pembanding untuk memaksimalkan metode kompresi data.
2. Kompresi data pada penelitian ini menggunakan data dari device yang terbatas, diharapkan pada penelitian berikutnya metode ini dapat diterapkan untuk bentuk lain dari sinyal yang terkait dengan IoT, misalnya pada data gambar dan sinyal biomedis, serta mendapatkan hasil teoritis lebih lanjut tentang algoritme dan kinerja untuk pembelajaran tentang kompresi data.

DAFTAR PUSTAKA

[1] Habibi, M.W., Bhawiyuga, A. & Basuki, A., 2018, Rancang Bangun IOT Cloud Platform Berbasis Protokol Komunikasi MQTT, Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X Vol. 2, No. 2, Februari 2018, hlm. 479-485

[2] Lin, Z. & Zhang, L., 2016, Data Synchronization Algorithm for loT Gateway and Platform, 2016 2nd IEEE International Conference on Computer and Communications (ICCC), Chengdu, 2016, pp. 114-119

[3] Salomon, D. & Motta, G., 2010, Handbook of data compression.

[4] Sayood, K., 2012, Lossless Image Compression, In, Introduction to Data Compression (Fourth Edition)

[5] Campobello, G., Segreto, A., Zanafi, S. & Serrano, S., 2017, RAKE : a Simple and Efficient Lossless Compression Algorithm for the Internet of Things, , 7, 2650–2654.

[6] Vecchio, M., Giaffreda, R. & Marcelloni, F., 2014, Adaptive lossless entropy compressors for tiny iot devices, IEEE Transactions on Wireless Communications, 13, 2, 1088–1100.

[7] Sari., K & M. Riasetiawan, The Implementation of Timestamp, Bitmap and RAKE Algorithm on Data Compression and Data Transmission from IoT to Cloud, 2018, IEEE 4th International Conference on Science and Technology (ICST), Yogyakarta, 2018, pp. 1-6. doi: 10.1109/ICSTC.2018.8528698.

[8] Pinho, A., J., 2002, An online preprocessing technique for improving the lossless compression of images with sparse histograms, IEEE Signal Processing Letters.

[9] Sari., K and M. Riasetiawan, The Development of IoT Compression Technique To The Cloud., 2019, Indonesian Journal of Computing and Cybernetics Systems (IJCCS), vol 13, no.4. doi: 10.22146/ijccs.47270.

[10] Desai, A., Nagegowda, K.S. & Ninikrishna, T., 2016, A framework for integrating IoT and SDN using proposed OF-enabled management device, Proceedings of IEEE International Conference on Circuit, Power and Computing Technologies, ICCPCT 2016, 1–4.

[11] Aazam, M., Khan, I., Alsaffar, A.A. & Huh, E.N., 2014, Cloud of Things: Integrating Internet of Things and cloud computing and the issues involved, Proceedings of 2014 11th International Bhurban Conference on Applied Sciences and Technology, IBCAST 2014, 414–419.

[12] Malhotra, N., & Chaudhary, A., 2014, Implementation of Database Synchronization Technique between Client and Server. International Journal of Engineering Science and Innovative Technology Volume 3, Issue 4, July 2014.

[13] Kumar, A., Nanjangud, C., Narendra & Umesh, B., 2016, Uploading And Replicating Internet Of Things (IoT) Data On Distributed Cloud storage", 2016 IEEE 9th International Conference On Cloud Computing, Vol. 00, No. Pp. 670-677, 2016, Doi:10.1109/CLOUD.2016.0094

[14] Sharma, M., 2010, Compression Using Huffman Coding, IJCSNS International Journal of Computer Science and Network Security.

[15] Hadiatna, F., Hindersah, H., Yolanda, D. & Triawan, M.A., 2017, Design and implementation of data logger using lossless data compression method for Internet of Things, Proceedings of the 2016 6th International Conference on System Engineering and Technology, ICSET 2016, 105–108.