Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.202296768
Vol. 9, No. 7, Desember 2022, hlm. 1655-1662 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

IMPLEMENTASI ALGORITME MORUS V2 UNTUK PENGAMANAN DATA PADA PERANGKAT *BLUETOOTH LOW ENERGY*

Diah Ratih Destyorini[1], Ari Kusyiati*[2], Reza Andria Siregar[3]

[1,2,3]Universitas Brawijaya, Malang
Email: [1]diah.ratih18@gmail.com, [2]ari.kusyanti@ub.ac.id, [3] reza.jalin@ub.ac.id
*Penulis Korespondensi

(Naskah masuk: 08 Desember 2022, diterima untuk diterbitkan: 26 Desember 2022)

Abstrak

Pengamanan data merupan bagian penting dalam penerapan jaringan berbasis *Internet of things* (IoT). Sistem IoT membutuhkan sebuah protokol komunikasi, seperti *Bluetooth Low Energy* (BLE). BLE dinilai cepat dan hemat energi dalam pengiriman data jarak dekat. BLE digunakan sebagai komunikasi *client-server* yang menerima data dari sensor. Pengiriman data tidak aman tanpa adanya proses pengaman data. Kriptografi menjadi salah satu pilihan dalam pengaman data. Pada pengembangan diperlukan pertimbangan beberapa aspek seperti, *resource* yang digunakan, serta waktu untuk proses enkripsi dan dekripsi. Pada penelitian ini algoritme MORUS V2 dipilih untuk mengamankan data dari serangan. Algoritme ini mudah diimplementasikan pada *hardware*. Kecepatan dari algoritme ini dapat mencapai 0,69 *cpb*, lebih cepat dari algoritme lain. Proses enkripsi data dari sensor dilakukan pada *server* hingga menghasilkan *ciphertext*. Kemudian akan dilakukan dekripsi pada *ciphertext* ketika diterima oleh *client*, hingga *plaintext* ditampilkan. Dari hasil pengujian algoritme MORUS V2 telah berhasil diimplementasikan melalui pengujian *test vector* dengan nilai *keystream* yang sama. Pengujian *confidentiality* telah berhasil dilakukan melalui proses enkripsi dan dekripsi. Pada pengujian serangan pasif berhasil dilakukan dengan hasil nilai *plaintext* tidak diketahui ketika data dikirim, serta tidak ada perubahan ketika data sampai pada *client*. Pengujian serangan aktif menggunakan *known-plaintext attack* (KPA) dinyatakan gagal dalam memperoleh nilai *plaintext*.

Kata kunci: Algortme MORUS V2, *Bluetooth Low Energy*, Enkripsi, *Confidentiality*

IMPLEMENTATION OF MORUS V2 ALGORITHM FOR DATA SECURITY ON BLUETOOTH LOW ENERGY DEVICES

Abstract

Data security is an essential part of implementing an Internet of things (IoT) based network. IoT systems require a communication protocol, such as Bluetooth Low Energy (BLE). BLE is considered fast and energy-efficient in sending data over short distances. BLE is used as a client-server communication that receives data from sensors. Data transmission will be insecure without a data security process. Cryptography is one of the options for securing data. The development requires consideration of several aspects, such as the resources used, as well as the time for the encryption and decryption process. In research, the MORUS V2 algorithm was chosen to secure data from attacks. This algorithm is easy to implement on hardware. The speed of this algorithm can reach 0.69 CPB, faster than other algorithms. The data encryption process from the sensor is carried out on the server to produce ciphertext. Then decryption will be carried out on the ciphertext when received by the client until the plaintext is displayed. From the test results, the MORUS V2 algorithm has been successfully implemented through test vector testing with the same keystream value. Confidentiality testing has been successfully carried out through encryption and decryption processes. The passive attack test was successfully carried out with the result that the plaintext value was unknown when the data was sent, and there was no change when the data arrived at the client. Active attack testing using a known-plaintext attack (KPA) is declared to have failed in obtaining the plaintext value.

Keywords: *MORUS V2 algorthm, Bluetooth Low Energy, Encyption , Confidentiality*

1. PENDAHULUAN

Enkripsi merupakan bagian penting dalam penerapan jaringan berbasis *Internet of things* (IoT). Pada IoT perangkat akan terhubung baik dari perangkat ke perangkat maupun perangkat ke manusia untuk berkomunikasi. Keamanan dan

privasi menjadi faktor penting dalam pengembangan IoT. Dalam pengembangannya diperlukan pertimbangan beberapa aspek seperti, *resource* yang digunakan, serta waktu untuk proses enkripsi dan dekripsi. IoT memiliki daya komputasi yang rendah sehingga dibutuhkan teknik enkripsi dan dekripsi yang ringan (Ghulam, et al., 2018). Pada sistem IoT, protokol *Bluetooth Low Energy* (BLE) menjadi salah satu pilihan. BLE dapat menangani banyak perangkat dan aplikasi nirkabel dalam jarak dekat yang tentunya cepat dan hemat energi (Keuchul, et al., 2016).

Data yang dikirim melalui protokol BLE menjadi sebuah celah bagi *attacker*. Serangan pasif seperti *eavesdropping* maupun aktif seperti *man-in-the-middle attack* (MITM) dapat terjadi dalam protokol BLE. Menggunakan algoritme enkripsi yang cocok dapat dapat meningkatkan keamanan pada sistem. Selain itu mempertimbangan berapa besar *resource* yang diperlukan untuk proses kedua proses tersebut**.** Melakukan proses enkripsi pada pada *server* serta proses dekripsi ketika data sampai pada *client* menjadi salah satu solusi yang baik. Berikut adalah penilitian pendahulu yang terkait dengan penelitian ini.

Pada penelitian "*Security of IoT Wireless Technologies*" menjelaskan perbedaan dari empat proses *pairing* yang ada pada BLE. Tiga proses pairing yaitu *Just Work, Passkey Entry* dan *Out-of-Band* digunakan pada *LE Legecy* mode. *LE Legecy* mode menujukan mudah diserang *passive eavesdropping*. Sedangkan proses *pairing* lainnya yaitu *Numeric Comparison* digunakan pada *LE Secure* tahan terhadap serangan *passive eavesdropping* maupun *MITM*. Sehingga *LE Secure* lebih baik daripada *LE Legacy*. Akan tetapi apapun metode *pairing* yang dipilih sistem akan tetap ada celah untuk diserang. Oleh karena itu dibutuhkan proses enkripsi lain (Vaadata, 2020). Proses pengamanan sistem IoT pernah dilakukan sebelumnya, dengan melalukan enkripsi pada bagian protokol *Bluetooth Low Energy*, *Wifi* dan *Xbee*. Algoritme yang digunakan adalah algoritme *Fibonacci*. Penelitian menunjukkan bahwa *Wifi* lebih unggul sedikit dari *Bluetooth Low Energy* untuk proses enkripsi, dekripsi data serta *throughput*. Sedangkan *Xbee* menunjukkan hasil keterlambatan tertinggi penundaan *transmisi* data (Amiruddin, et al., 2018).

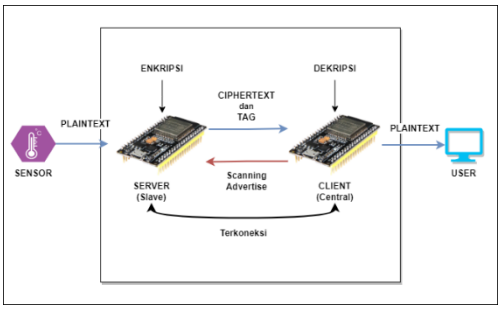
Penelitian ini akan dilakukan dengan menggunakan Algoritme MORUS V2. Algoritme ini menggunakan CAESAR *hardware* API. Algoritme MORUS V2 juga mudah diimplementasikan ke *hardware*. Kecepatan dari algoritme MORUS V2 dapat mencapai 0,69 *cpb*, lebih cepat dari AES-128-GCM (Hongjun & Tao, 2016). Pengaman pada BLE juga pernah dilakukan oleh Afif, M dengan melakukan pengamanan pengiriman data menggunakan algoritme GRAIN dengan waktu enkripsi 0,0047 detik dan dekripsi 0,0049 detik (Afif, 2021).

Sistem yang digunakan pada penelitian ini terdiri dari *client-server*. Pada bagian *server* menggunakan protokol BLE sebagai alat untuk menangani data sensor yang masuk. Sedangkan pada *client* yang juga menggunakan protokol BLE untuk menerima data sensor dari *server*. Data yang masuk ke *server* akan dilakukan proses enkripsi dengan menggunakan algoritme MORUS V2. Kemudian proses dekripsi akan dilakuakan pada *client*. Sehingga *attacker* tidak dapat menyerang baik secara aktif maupun pasif.

2. METODE PENELITIAN

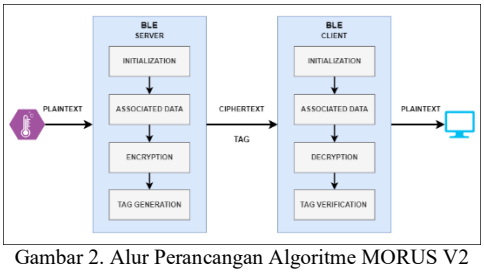
2.1. Perancangan

Pada penelitian sistem yang digunakan adalah sebuah sistem *client-server*. Sistem menggunakan perangkat sensor berbasis protokol BLE sebagai komunikasi antara node ke node. Peneliti akan mengembangkan keamanan pada sistem dengan cara mengimplementasikan algoritme kriptografi yaitu algoritme MORUS V2.



Gambar 1. Alur Kerja Sistem

Gambar 1 sistem menggunakan mikrokontroler ESP32(BLE) sebagai *client-server*. Pertama sensor akan mengirim data sensor (*plaintext*) ke *server*. *Server* akan mengirim *advertise* terlebih dahulu, dan *client* melakukan *scanning advertise* dengan alamat *service* dan *characteristic* UUID yang sesuai. Ketika terhubung maka *server* menjadi *slave* dan *client* menjadi *central*. Ketika data sensor sampai ke *server* dilakukan proses enkripsi dan pembuatan *tag*. Sebelum menerima data dari *server, client* malakukan koneksi dengan *server*. Kemudian *server* mencari *service* serta *characteristic* UUID yang cocok. Ketika terhubung *server* akan mengirim *ciphertext* dan *tag* ke *client*. Pada *client* proses dekripsi dilakukan untuk mengembalikan nilai *ciphertext* ke *plaintext*. Kemudian melakukan proses verifikasi *tag*, jika *tag valid* maka *plaintext* akan ditampilkan.



Gambar 2. Alur Perancangan Algoritme MORUS V2

Gambar 2 dijelaskan alur komunikasi mulai dari sensor sampai client. Ketika sensor mengeluarkan data berupa plaintext dan sampai pada server maka akan diubah menjadi ciphertext menggunakan algoritme MORUS V2, mulai dari initialization, associated data, encryption dan tag generation. Kemudian server akan mengirim ciphertext. Ketika client telah menerima ciphertext maka akan dilakukan proses initialization, associated data, decryption dan tag verification.

3. DASAR TEORI

3.1. *Bluetooth Low Energy*

Pada juni 2010 Bluetooth dibagi menjadi 2 jenis yaitu *Basic Rate/Enhanced Data Rate* (BR/EDR) dan *Low Energy* (LE). *Bluetooth Low Energy* (BLE) menggunakan *physical layer* yang tidak kompatibel dengan Bluetooth klasik. BLE memancarkan daya maksimum 10Mw, dan kecepatan data 1 *Mbps*. Perancangan ini berfungsi untuk digunakan pada IoT berdaya sangat rendah(Harris, et al., 2020).

3.2. ESP32

Gambar 3. Mikrokontroler ESP32

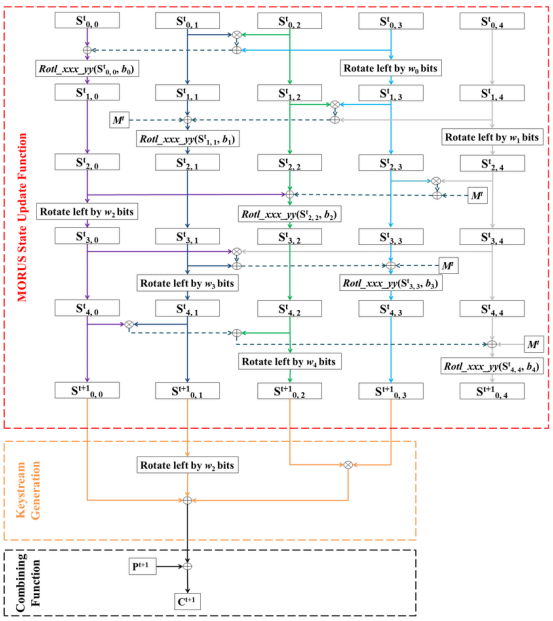
Gambar 3 merupakan mikrokontroler ESP32. ESP32 adalah sebuah mikrokontroler yang memiliki kemampuan *WIFI* dan *Bluetooth*. ESP32 yang dibuat sebagai alat pendukung perangkat *Internet of things* dengan kinerja yang baik dan harga terjangkau (Alexander, et al.,2017).

3.3. Kriptografi

Kriptografi adalah konsep penyandian untuk kerahasiaan data sehingga hanya pihak berwenang yang mengetahui makna data aslinya. Pada kriptografi *plaintext* (data asli) akan dienkripsi menjadi *ciphertext* kemudian akan di dekripsi kembali ke *plaintext* ketika akan diterima penerima. Saat ini algorithme modern banyak digunakan untuk mengamankan data. Algorithme ini dibagi menjadi 2 jenis yaitu *stream ciphers* dan *block ciphers*. *Stream ciphers* melakukan *XOR* antara *keystream* dan *plaintext* sehingga menghasilkan *ciphertext*. *Stream ciphers* banyak digunakan untuk koneksi Bluetooth, 4G seluler, koneksi TLS, dsb. Algorithme MORUS adalah salah satu dari *stream ciphers* (Qadir dan Varol, 2019).

3.4. Algoritme MORUS V2

Algorithme MORUS adalah *stream cipher* yang tergabung pada kompetisi CAESAR *Authentication Encryption*. MORUS dibagi menjadi 2 *state* dan 2 *key* yaitu 640-128, 1280-128, dan 1280-256. Nilai *cipher* diperoleh dari proses enkripsi *plaintext* "*XOR*" dengan *keystream*. MORUS V2 juga menggunakan *associated data* untuk proses integritas data. Serta pembentukan *tag* untuk *authentication* (Salam, 2021).



Gambar 4. Alur algoritme MORUS V2

A. Initialization

Pada perancangan algoritme MORUS V2 ada beberapa tahap untuk menghasilkan *ciphertext* dan *authentication tag*. Proses pertama adalah *initialization*. Gambar 2 adalah proses *state update* menggunakan masukan *IV, Key*, serta *const*. proses *state update* di lakukan sebanyak 16 kali iterasi (perulangan). Berikut merupakan persamaan proses inisialisasi:

$$For\ i = -16\ to\ -1$$
$$S^{i+1} = StateUpdate(S^i, 0) \quad (1)$$

Setelah proses *state update* selesai, dilanjutkan dengan proses *key generation* dengan cara melakukan proses perhitungan *XOR* antara S_1^0 dengan nilai K_0. Setelah melakukan

proses *key generate*, S_1 akan diperbahurui dan masukan akan digunakan untuk *state update* pada *prosessing associated data*.

B. *Processing Assocoated Data*

prosessing associated data dilakukan dengan *state update* sebanyak 1 kali iterasi. Nilai *adlen* = 256, sehingga:

$$S^{i+1} = StateUpdate(S^i, AD_i^{256}).$$
$$For\ i = 0\ to\ i, i = \frac{adlen}{256} - 1 \quad (2)$$

C. *Encryption and Decryption*

Setelah proses *associated data* selesai, proses selanjutnya adalah proses enkripsi. Berikut adalah persamaannya:

$$C_i = P_i \oplus S_0^{u+i} \oplus (S_1^{u+i} <<< 192) \oplus (S_2^{u+i} \,\&\, S_3^{u+i}) \quad (3)$$

Setelah proses enkripsi selesai dilakukan *state update* sebanyak 1 kali iterasi. Untuk proses dekripsi dilakukan ketika akan mengubah *ciphertetxt* menjadi *plaintext* ketika data diterima. Berikut adalah persamaannya:

$$P_i = C_i \oplus S_0^{u+i} \oplus (S_1^{u+i} <<< 192) \oplus (S_2^{u+i} \,\&\, S_3^{u+i}) \quad (4)$$

D. *Tag Generate and Verification*

Pembuatan *tag* dilakukan *state update* sebanyak 10 kali iterasi. Kemudian dilakukan proses pembuatan *tag* sesuai dengan persamaan berikut:

$$T' = S_0^{-u+v+10} \oplus (S_1^{-u+v+10} <<< 192) \oplus (S_2^{-u+v+10} \,\&\, S_3^{-u+v+10}) \quad (5)$$

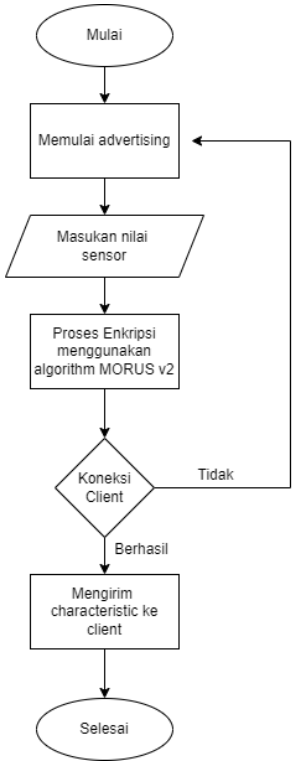
Ketika proses selesai maka akan terjadi proses verifikasi *tag*. Jika *tag* cocok maka *output* akan mengeluarkan *plaintext* serta *tag* yang baru. Jika proses verifikasi gagal maka nilai *tag* dan *plaintext* tidak akan akan ditampilkan. dari semua proses algoritme MORUS V2 didapatkan *confidentiality* serta *integrity* (Hongjun & Tao, 2016).

4. IMPLEMENTASI

Proses implementasi algoritme MORUS V2 ke sistem menggunakan *software* Arduino IDE. Pada penelitian ini menggunakan algoritme MORUS V2 1280-128-*bit key*. Pada sistem *server* dan *client* terdapat *service* dan *characteristic* UUID, void untuk *connect*, fungsi *setup* mengirim dan menerima *advertising* serta fungsi *loop* untuk mengeksekusi sistem yang telah dibuat. Kemudian dilakukan proses implemetasi algoritme. Berikut adalah implementasi sistem.

4.1. *Sensor-Server*

Pada bagian ini dilakukan implementasi pada *server* (*slave*). Pada *server* terjadi proses pembentukan *keystream*, enkripsi dan pembuatan *tag*. berikut adalah alur dari sistem dari sensor sampai ke *server*.



Gambar 1. *Flowchart* BLE *Sensor-Server*

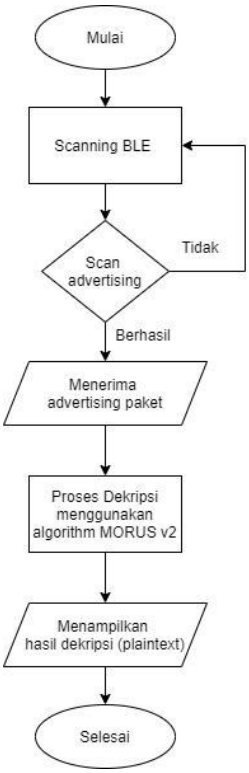
Gambar 5 adalah *flowchart* proses enkripsi yang berada pada BLE *server*. Proses pertama adalah dengan memulai *advertising* (untuk membentuk koneksi dengan *client*). Kemudian memasukan data sensor. Setelah data diterima maka akan langsung melakukan proses enkripsi menggunakan algoritme MORUS V2. Jika koneksi dengan *client* berhasil maka *server* akan mengirim *characteristic* (termasuk data *ciphertext*) ke *client*. Jika koneksi gagal maka akan kembali ke proses mengirim *advertising*.

4.2. *Client-Server*

Pada bagian ini implementasi dilakukan pada bagian *client* (*central*). Proses implementasi meliputi pembentukan *keystream*, dekripsi, pembentukan *tag* dan verifikasi *tag*. Berikut adalah alur dari sistem dari *server* sampai ke *client.*

Gambar 6 adalah *flowchart* proses dekripsi pada BLE *client*. pertama adalah membentuk koneksi dengan *server*, dengan cara *scanning advertising*. Jika berhasil terkoneksi dengan *server* maka *client* akan menerima paket (data *ciphertext*). Kemudian sistem akan langsung melakukan proses dekripsi menggunakan algoritme MORUS V2. BLE *client* akan

menampilkan hasil dekripsi (*plaintext*). Jika proses koneksi gagal, maka akan melakukan proses *scanning advertise* lagi.



Gambar 2. *Flowchart* BLE *Client-Server*

5. PENGUJIAN DAN HASIL PEMBAHASAN

5.1. Pengujian Validasi *Test Vector*

Pengujian *test vector*, dengan cara menyesuaikan nilai *keystream* sistem dengan *keystream* yang ada pada jurnal MORUS V2. Hal yang perlu dilakukan yaitu dengan menggunakan nilai *IV* dan *Key* yang sama dengan *IV* dan *Key* yang ada pada jurnal MORUS V2. Kemudian menyesuaikan nilai *keystream* yang dihasilkan sistem dengan *keystream* jurnal MORUS V2. Terdapat 3 *test vector* yang digunakan.

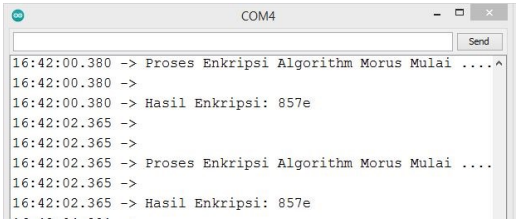
Tabel 1 menunjukkan hasil dari uji validasi antara *keystream test vector* yang ada pada jurnal MORUS V2 dengan *keystream* yang dihasilkan oleh sistem. Hasilnya menunjukkan bahwa pengujian dinyatakan *valid*. Sehingga algoritme yang telah diimplemetasi telah berjalan dengan benar.

5.2. Pengujian Kerahasian Data

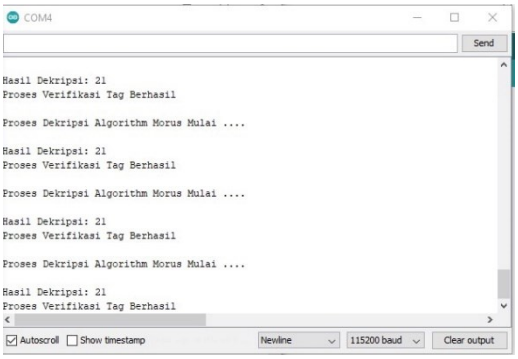
Pengujian kerahasiaan data (*confidentiality*) dilakukan untuk menguji kerahasiaan data ketika data sensor dikirim dari *server* ke *client*. keberhasilan enkripsi dan dekripsi dengan menggunakan *key* yang sama akan dipastikan dalam pengujian pada Gambar 7.

Tabel 1. Pengujian Test Vector Jurnal MORUS V2

No	*IV*	*Key*	*Test Vector*	*Keystream* Sistem
1	0x00	0x000	0x36307857	0x36307857
	0000	00000	0x40fc7d91	0x40fc7d91
	0000	00000	0xb2550c5e	0xb2550c5e
	0000	00000	0x0baf2b0a	0x0baf2b0a
	0000	00000	0x1300f193	0x1300f193
	0000	00000	0xea1def45	0xea1def45
	0000	0000	0xb96762b2	0xb96762b2
	0000		0x75886d9e	0x75886d9e
	00			
2	0x01	0x010	0x43808fe0	0x43808fe0
	0101	10101	0xa012d7eb	0xa012d7eb
	0101	01010	0xcb0dabfe	0xcb0dabfe
	0101	10101	0x86cab464	0x86cab464
	0101	01010	0x691022d1	0x691022d1
	0101	10101	0x085c5fd1	0x085c5fd1
	0101	0101	0x93c8b8f5	0x93c8b8f5
	0101		0x54b3bd2e	0x54b3bd2e
	01			
3	0x00	0x000	0x1c0a0226	0x1c0a0226
	0306	10203	0xd425cc01	0xd425cc01
	090c	04050	0x459fc020	0x459fc020
	0f12	60708	0x0f67ca63	0x0f67ca63
	1518	090a0	0x43f37b0a	0x43f37b0a
	1b1e	b0c0d	0x43cc7580	0x43cc7580
	2124	0e0f	0x772968a9	0x772968a9
	272a		0x358dbb87	0x358dbb87
	2d			



Gambar 3. Proses Enkripsi



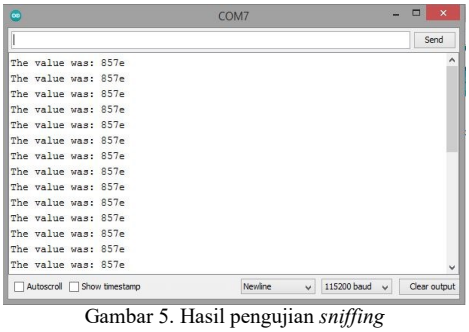
Gambar 4. Proses Dekripsi

Gambar 7 dan 8 merupakan *server* dan *client*. Pada *server* terdapat proses enkripsi dan menghasilkan *ciphertext*. Hasil *ciphertext* tersebut dikirimkan ke *client*. pada *client* proses dekripsi dilakukan setelah data diterima di *client*. kemudian dilakukan verifikasi *tag*. Ketika verifikasi *tag* berhasil makan hasil dari dekripsi ditampilkan yaitu hasil dekripsi berupa *plaintext*. Sehingga *confidentiality* sistem terpenuhi.

5.3. Pengujian Serangan

A. Serangan Pasif

Pengujian serangan pasif (*sniffing*) dilakukan dengan cara *monitoring* dan *capturing* setiap paket yang dikirim oleh protokol BLE (*server*) ke BLE (*client*) menggunakan sistem *sniffing*. Pengujian ini dilakukan untuk memastikan bahwa data yang dikirim *server* ke *client* terlindungi atau terenkripsi. Sehingga pihak yang tidak berwenang tidak dapat mengetahui pesan asli (*plaintext*) yang dikirim.

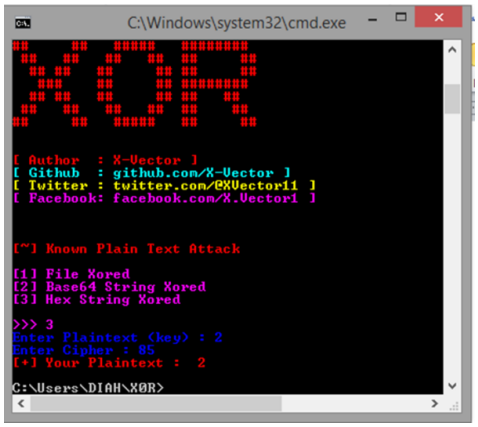


Gambar 5. Hasil pengujian *sniffing*

Gambar 9 menujukan hasil dari *sniffing* paket. Hasil data yang dikirim ke *client* adalah bilangan *hexadecimal* '857e'. Data tersebut tidak sama dengan data sensor (*plaintext*) yang asli. Sehingga proses enkripsi dipastikan *valid* dan data yang dikirim ke *client* diterima secara aman.

B. Serangan Aktif

Pengujian serangan aktif dilakukan untuk memastikan sistem aman akan serangan *attacker*. Serangan aktif yang digunakan pada penelitian ini adalah *known-plaintext attack* (KPA). KPA dilakukan untuk mencari nilai *plaintext*. Serangan ini dilakukan dengan cara memasukkan beberapa nilai dari *plaintext* dan hasil dari proses enkripsi yaitu *ciphertext*. Kemudian akan dilakukan analisa data untuk menentukan *secret key* yang digunakan untuk enkripsi dan dekripsi. *Key* tersebut akan digunakan untuk menghasilkan *plaintext* (Sakiyama, Sasaki and Li, 2016).



Gambar 6. Hasil pengujian *known-plaintext attack*

Gambar 10 adalah hasil dari serangan *kown-plaintext yaitu plaintext* "2" dari sampel *plaintext* "2" dan *ciphertext* "85". Sedangkan nilai asli dari plaintext adalah "21", sehingga dapat dibuktikan sistem aman dari *known-plaintext attack* (KPA).

6. KESIMPULAN DAN SARAN

6.1 KESIMPULAN

Berdasarkan penelitian implementasi algoritme MORUS V2 untuk pengaman data pada *Bluetooth low energy* dapat ditarik kesimpulan yaitu:

1. Implementasi algoritme MORUS V2 pada sistem (*client-server*) menghasilkan hasil *valid* pada pengujian *test vector* yang diketahui dari *keystream* yang dihasilkan sistem sama dengan *keystream* pada jurnal MORUS V2. Sehingga implementasi algoritme MORUS V2 pada sistem telah berhasil.
2. Pengaman sistem (*client-server*) menggunakan algoritme MORUS V2 telah berhasil dilakukan. Pada pengujian pengaman data dilakukan pada *server* dengan melakukan enkripsi pada data sebelum dikirim ke *client*. Setelah data diterima pada *client*, maka akan dilakukan proses dekripsi sehingga nilai asli dapat ditampilkan. Proses ini dilakukan untuk menjaga *confidentiality* pada sistem.
3. Pada pengujian serangan dapat disimpulkan bahwa serangan pasif berhasil dilakukan dengan hasil nilai asli data yang dikirim tidak diketahui ketika dilakukan *sniffing*, sehingga data yang dikirim aman tanpa ada perubahan sampai pada *client*. Sedangkan untuk serangan aktif yang menggunakan *Known-palintext* juga telah berhasil dilakukan. Pengujian dilakukan dengan memasukan sebagian nilai *plaintext* dan *cipertext,* dengan hasil nilai yang ditampilkan tidak sama dengan data yang asli, sehingga proses pengaman data menggunakan algoritme MORUS V2 telah berhasil.

6.2 SARAN

1. Pada penelitian ini dilakukan pengaman data dengan algoritme MORUS V2. Pada penelitian selanjutnya dapat dilakukan dengan menggunakan algoritme lainnya seperti ACORN atau AGEIS.
2. Pada penelitian ini hanya mengimplementasi algoritme MORUS V2 pada node ke node BLE (*client-server*). Pada penelitian selanjutnya algoritme MORUS V2 dapat diterapkan

pada sistem yang lebih *complex,* seperti sistem dengan *gateway* dan *middleware*.

DAFTAR PUSTAKA

AFIF, M. Implementasi Algoritme Enkripsi Grain-Cipher di ESP32 untuk Pengiriman Data lewat Modul Bluetooth Low Energy (BLE). Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer**,** vol. 5, no. 7, p. 2866-2870, juni 2021. ISSN 2548-964X. Tersedia pada: <https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/9366>. [Diakses 7 Februari 2022]

ALEXANDER MAIER., et al., 2017. Comparative Analysis and Practical Implementation of the ESP32 Microcontroller Module for the *Internet of things*, [e-jurnal]. Tersedia melalui: < https://www.researchgate.net/publication/320273388_Comparative_Analysis_and_Practical_Implementation_of_the_ESP32_Microcontroller_Module_for_the_Internet_of_Things > [Diakses 15 Februari 2020]

AMIRUDDIN AMIRUDDIN., et al.,2018. Secure Multi-protocol Gateway for *Internet of things*, [e-jurnal]. Tersedia melalui: https://ieeexplore.ieee.org/document/8363934 [Diakses 6 April 2020]

GHULAM MUSTAFA., et al.,2018. A Review of Data Security and Cryptographic Techniques in IoT based devices, [e-jurnal]. Tersedia melalui: < https://www.researchgate.net/publication/327325243_A_review_of_data_security_and_cryptographic_techniques_in_IoT_based_devices> [Diakses 8 Mei 2020]

HARRIS, J., SMALL, L., HOPKINS, M. AND DE LAUTOUR, N., 2020. *A Summary of Bluetooth Low Energy*, [e-jurnal] p.1. Tersedia at: <https://www.dta.mil.nz/assets/Publications/A-Summary-of-Bluetooth-Low-Energy.pdf> [Diakses 23 Mei 2021].

HONGJUN, W., TAO, H., 2016. The Authenticated Cipher MORUS V2 (V2), [e-jurnal]. Tersedia melalui: < https://competitions.cr.yp.to/*round*3/MORUSV2.pdf> [Diakses 15 Februari 2020]

IFTEKHAR SALAM., et al., 2018. Fault *Attack*s on the Authenticated Encryption Stream Cipher MORUS V2, [e-jurnal]. Tersedia melalui: < https://www.semanticscholar.org/paper/Fault-*Attack*s-on-the-Authenticated-Encryption-MORUS V2 -Salam-Simpson/bd5050f6883cd71f2e81a735c9194a577f0ea27a> [Diakses 1 Mei 2020]

KEUCHUL CHO., et al.,2016. Performance Analysis of Device Discovery of *Bluetooth Low Energy* (BLE) Networks, [e-jurnal]. Tersedia melalui: < https://www.sciencedirect.com/science/article/pii/S0140366415003886> [Diakses 6 Februari 2020]

QADIR, A. AND VAROL, N., 2019. A Review Paper on Cryptography. *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*, [e-jurnal] pp.2-3. Tersedia at: <https://ieeexplore.ieee.org/document/8757514> [Diakses 14 April 2021].

SAKIYAMA, K., SASAKI, Y. AND LI, Y., 2016. *Security Of Block Ciphers From Algorithm Design To Hardware Implementation.* Weley, pp.72-74.

VAADATA, 2020. Security of Iot Wireless Technologies, [e-jurnal]. Tersedia melalui: <https://www.vaadata.com/blog/white-paper-security-iot-wireless-technologies/> [Diakses 6 April 2020]

Halaman ini sengaja dikosongkan