Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 5, No. 3, Maret 2021, hlm. 1119-1125 http://j-ptiik.ub.ac.id

Implementasi Enkripsi *Vernam Cipher* dan Distribusi Kunci *Three-Pass Protocol* untuk Mengamankan Data *Chatting* pada ATmega328

Budiyanto[1], Rakhmadhany Primananda[2], Fariz Andri Bakhtiar[3]

Program Studi Teknik Komputer, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]budiyanto@pasuh.com, [2]rakhmadhany@ub.ac.id, [3]fariz@ub.ac.id

Abstrak

Perangkat Arduino dengan mikroprosesor ATmega328 dan modul radio NRF24 dimungkinkan sebagai alat komunikasi *chatting* secara nirkabel. Komunikasi data yang dilakukan NRF24 pada umumnya tidak terenkripsi, sehingga komunikasi *chatting* menjadi tidak aman. Ada berbagai macam algoritme enkripsi yang dapat digunakan, tetapi tidak semua algoritme enkripsi cocok diterapkan pada ATmega328 karena adanya keterbatasan. Mikroprosesor ATmega328 memiliki kemampuan komputasi yang sangat terbatas, sehingga kurang cocok jika proses enkripsi dilakukan menggunakan algoritme dengan tingkat kompleksitas komputasi yang tinggi. *Vernam Cipher* adalah algoritme yang menggunakan perhitungan komputasi sederhana dengan *XOR*. *Three-Pass Protocol* adalah protokol transmisi data sederhana yang dapat digunakan untuk mengamankan distribusi kunci enkripsi. Di dalam penelitian ini, algoritma *Vernam Cipher* dan *Three-Pass Protocol* digabungkan untuk mengamankan data *chat*. Data *chat* yang dapat diamankan dalam penelitian ini adalah teks, gambar, video dan audio. Berdasarkan hasil pengujian sebanyak 10 kali pada pengiriman data *chat*, didapat hasil uji dengan persentase 100% bahwa sistem dapat mengamankan data *chat*. Dari hasil uji didapatkan waktu rata-rata proses enkripsi 16 karakter string dengan algoritme *Vernam Cipher* pada ATmega328 yaitu 1,9 mili detik.

Kata kunci: *vernam, three, pass, protocol, chat, enkripsi*

Abstract

Arduino devices with ATmega328 microprocessors and NRF24 radio modules can be used as a wireless chat communication devices. Data communication by NRF24 is generally unencrypted, so chat communication is not secure. There are various encryption algorithms that can be used, but not all encryption algorithms are suitable for ATmega328 because of limitations. ATmega328 microprocessor has very limited computational capability, making it less suitable if the encryption process is performed using an algorithm with a high level of computational complexity. Vernam Cipher is an algorithm that uses simple calculations with XOR. Three-Pass Protocol is a simple data transmission protocol that can be used to secure the distribution of encryption keys. In this study, the Vernam Cipher algorithm and Three-Pass Protocol are combined to secure chat data. Chat data that can be secured in this study are text, images, video and audio. From the test results of 10 times on sending chat data, obtained test results with 100% percentage that the system can secure chat data. From the test results, the average time for the encryption process 16 character strings with the Vernam Cipher algorithm on the ATmega328 is 1.9 millisecond.

Keywords: *vernam, three, pass, protocol, chat, encryption*

1. PENDAHULUAN

Keamanan data memiliki peranan penting dalam menjaga privasi komunikasi *chat* antar perangkat *End Device*. *Chat* merupakan fitur yang diminati oleh para anak muda di jaringan internet. Di dalam ruang *chat*, setiap orang dapat membuat obrolan pesan teks dengan orang lain (Hassan & Mohammad, 2007). Arduino Nano merupakan pengendali mikro berukuran kecil yang telah terintegrasi dengan mikroprosesor ATmega328 (Arduino, 2018). Dengan adanya tambahan modul radio NRF24 memungkinkan Arduino untuk melakukan komunikasi data *chat* secara nirkabel. Namun, di jaringan publik sulit untuk dijaga mengenai privasi dan kerahasiaan data, karena sifatnya publik. Oleh karena itu,

perlu adanya proses untuk melindungi data *chat*.

Teknik kriptografi sangat berperan penting untuk melindungi data *chatting* melalui jaringan publik. Dengan adanya enkripsi maka data yang dikirim menjadi tidak mudah dibaca oleh semua orang (Gowda, 2016). Teknik kriptografi terbagi menjadi dua yaitu klasik dan modern. Teknik kriptografi modern sangat identik dengan proses komputasi yang kompleks sehingga butuh waktu lebih lama dalam proses enkripsi atau dekripsi. Kecepatan komputasi pada ATmega328 sangat rendah yaitu 16 MHz. Oleh karena itu, teknik kriptografi klasik lebih cocok untuk digunakan pada ATmega328. Kriptografi klasik terbagi menjadi dua jenis yaitu simetris dan asimetris. Pada sistem simetris, kunci rahasia tunggal harus dibagikan di antara *node* yang berkomunikasi. Pada sistem asimetris, setiap *node* memiliki dua kunci, salah satu kunci dirahasiakan dan kunci yang lainnya tersedia untuk umum. Keuntungan dari jenis simetris yaitu memiliki kompleksitas komputasi lebih rendah (Uchôa, et al, 2007).

Setiap kunci rahasia pada sistem kriptografi simetris harus dibagikan ke seluruh sistem yang berpartisipasi. Masalah dalam hal ini adalah kunci harus dibagikan tanpa ada proses enkripsi, sehingga kunci dapat disadap. Salah satu solusi yaitu dengan melakukan pra-distribusi kunci, sehingga tidak akan ada kunci rahasia yang perlu dikirim. Nilai pra-distribusi kunci bersifat statis, sehingga apabila kunci telah berhasil diketahui oleh pihak ketiga maka pesan selanjutnya dapat didekripsi oleh pihak ketiga menggunakan kunci yang sama. Solusi untuk masalah tersebut dapat diselesaikan dengan algoritme *One-Time Pad*.

One-Time Pad merupakan sebuah algoritme agar tidak terjadi perulangan penggunaan kunci yang sama. Dengan algoritme tersebut maka sebuah kunci pada proses enkripsi dan dekripsi hanya akan digunakan sekali dalam satu waktu. Kunci tersebut dibuat secara acak serta memiliki panjang yang sama dengan panjang *plaintext*. *Vernam Cipher* merupakan salah satu algoritme enkripsi yang menerapkan algoritme *One-Time Pad* (Nagaraj, 2012). Masalah dalam hal ini adalah kunci yang dihasilkan selalu bernilai acak sehingga proses pra-distribusi kunci sangat tidak mungkin untuk dilakukan. Oleh sebab itu, perlu adanya distribusi kunci antar *node* secara aman. *Three-Pass Protocol* adalah sebuah protokol yang dikembangkan oleh Adi Shamir pada tahun 1980. Penjelasan singkat dari protokol tersebut yaitu masing-masing sistem pengirim pesan dan sistem penerima pesan melakukan enkripsi dan dekripsi yang berbeda (Rachmawati, Aulya, & Budiman, 2018).

Telah banyak dilakukan penelitian dengan algoritme distribusi kunci *Three-Pass Protocol*. Salah satunya penelitian di tahun 2016 berjudul *Three-Pass Protocol Implementation in Caesar Cipher Classic Cryptography* menggunakan *Caesar Cipher* sebagai algoritme enkripsi utama (Oktaviana & Siahaan, 2016). Berdasarkan buku tahun 2005 berjudul *Cryptography And Network Security Principles And Practices* menjelaskan bahwa algoritme *Caesar Cipher* sangat mudah diserang dengan teknik *brute force* karena hanya memiliki 25 kemungkinan kunci saja. Kemudian penelitian yang berjudul *A Realizable Quantum Three-Pass Protocol Authentication Based on Hill-Cipher Algorithm* menggunakan algoritme enkripsi *Hill Cipher* (Abdullah, Khalaf, & Riza, 2015). Proses enkripsi *Hill Cipher* menggunakan komputasi dengan perhitungan matriks yang cukup kompleks, sehingga kurang cocok apabila diterapkan pada ATmega328.

Berdasarkan beberapa penelitian yang telah dilakukan, algoritme *Three-Pass Protocol* dapat meningkatkan keamanan dari algoritme enkripsi yang digunakan. Namun, dalam penelitian yang pernah dilakukan masih terdapat kekurangan dalam proses implementasi *Three-Pass Protocol* terutama untuk perangkat dengan kecepatan komputasi yang rendah, sehingga itulah yang mendasari penulis untuk menerapkan algoritme enkripsi *Vernam Cipher* dan algoritme distribusi kunci menggunakan *Three-Pass Protocol* pada ATmega328. Adanya penelitian ini diharapkan dapat mengamankan komunikasi *chatting* pada perangkat ATmega328.

2. TINJAUAN PUSTAKA

Buku berjudul *Cryptography And Network Security Principles And Practices* yang dirilis tahun 2005 oleh William Stallings menjelaskan bahwa algoritme *Caesar Cipher* sangat mudah diserang dengan teknik *brute force*. Cara kerja dari proses enkripsi menggunakan algoritme *Caesar Cipher* hanya dilakukan dengan menukar karakter huruf. Total huruf alfabet dari huruf a sampai z berjumlah 26 huruf. Huruf yang telah dilakukan enkripsi menggunakan *Caesar Cipher* pasti tidak sama dengan huruf yang belum dilakukan enkripsi. Jika salah satu dari 26 huruf tersebut dijadikan sebagai *ciphertext* (karakter yang sudah dilakukan proses enkripsi) maka 25 huruf lainnya merupakan *plaintext* (karakter yang belum dilakukan enkripsi). Jadi, *Caesar*

Cipher hanya memiliki total 25 kemungkinan *plaintext*.

Pada tahun 2016 juga terdapat penelitian berjudul *Three-Pass Protocol Implementation in Caesar Cipher Classic Cryptography* yang menggunakan algoritme *Three-Pass Protocol* untuk dijadikan sebagai proses untuk membuat proses enkripsi *Caesar Cipher* dapat dilakukan dengan kunci acak. Pada penelitan tersebut masih terdapat kelemahan di mana kunci hanya terdiri dari satu digit saja dan penggunaan kunci dilakukan berulang pada setiap karakter serta kunci hanya berubah ketika berganti kata. Jadi, kelemahan pada penelitian tersebut yaitu terjadi perulangan penggunaan kunci di setiap karakter pada kata.

William Stallings pada buku karangannya mengatakan bahwa algoritme enkripsi *Vernam Cipher* tidak mungkin dipecahkan. Walaupun telah dilakukan serangan *brute force*, sangat sulit untuk menemukan kunci enkripsi yang benar. Hal itu dikarenakan hasil dari proses enkripsi menggunakan *Vernam Cipher* hasilnya tidak unik. Hasil *cipertext* pada *Vernam Cipher* bisa memiliki nilai yang sama dengan *ciphertext* yang lain walaupun *plaintext* yang digunakan tidak sama. Selain itu, penggunaan tehnik *One-Time Pad* dapat membuat *Vernam Cipher* menjadi tidak mungkin untuk dipecahkan. Kesulitan dalam implementasi *One-Time Pad* di mana kunci hanya digunakan satu kali adalah proses distribusi kunci acak. Pra-distribusi kunci acak sangat tidak mungkin dilakukan.

Penelitian pada tahun 2018 berjudul *Three-Pass Protocol Scheme for Bitmap Image Security using Vernam Cipher Algorithm* digunakan kombinasi algoritme *Three-Pass Protocol* dan algoritme *Vernam Cipher* untuk digunakan pada proses penyandian file gambar. Hasil penelitian tersebut menyebutkan bahwa penelitian tersebut berhasil dilakukan dengan melakukan enkripsi di setiap piksel file gambar. Berdasarkan penelitian sebelumnya yang telah disebutkan, kombinasi *Three-Pass Protocol* dan *Vernam Cipher* dapat digunakan dengan baik.

Pada penelitian ini, algoritme *Three-Pass Protocol* dan *Vernam Cipher* digunakan untuk penyandian data *chatting*. Sistem komunikasi chatting pada penelitian ini digunakan Arduino Nano dengan ATmega328. Komunikasi *chatting* dilakukan secara nirkabel. Komunikasi nirkabel antara sistem dapat dilakukan dengan bantuan modul NRF24. Dengan tambahan modul ESP-01 membuat sistem dapat membuat layanan server situs web sehingga pengguna dan sistem chatting dapat berinteraksi melalui tampilan antarmuka pada aplikasi browser internet.

3. DASAR TEORI

3.1. Enkripsi Dan Dekripsi

Enkripsi adalah sebuah proses penyandian untuk melindungi data (Goyal & Kinger, 2013). *Plaintext* adalah data asli yang belum melalui proses enkripsi. *Key* adalah parameter yang berpengaruh saat proses enkripsi atau dekripsi. *Ciphertext* adalah data yang telah tersandi. Dekripsi adalah proses kebalikan dari enkripsi. Proses dekripsi akan membuat pesan *ciphertext* kembali menjadi *plaintext* (Gautam, et al, 2018).

3.2. *Vernam Cipher*

Vernam Cipher merupakan sebuah algoritme enkripsi dengan jenis yaitu *stream cipher* dengan simetris *key*. *Stream cipher* merupakan sebutan untuk enkripsi yang dilakukan setiap 1 karakter atau 1 bit. Simetris *key* merupakan sebutan untuk algoritme yang menggunakan 1 kunci yang sama dalam proses enkripsi atau dekripsi. Pimpinan pasukan tentara bernama Joseph Mauborgne mengusulkan penggunaan konsep *One-Time Pad* pada algoritme *Vernam Cipher*. Mauborgne mengusulkan kunci dibuat secara acak sehingga tidak ada perulangan *key* (Stallings, 2005).

$$Ci = Pi \oplus Ki \quad (1)$$

$$Pi = Ci \oplus Ki \quad (2)$$

Sumber: (Stallings, 2005)

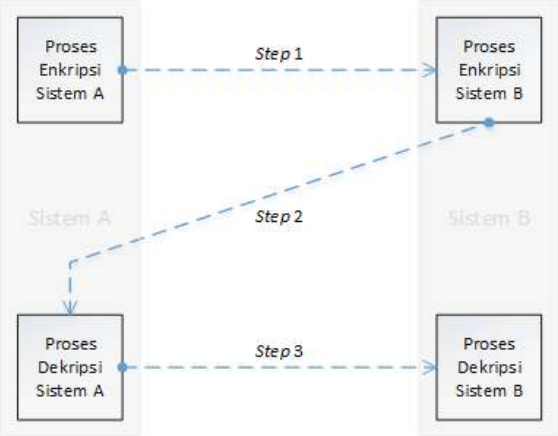
Keterangan Persamaan 1 dan Persamaan 2 :
Pi adalah angka biner dari *plaintext*
Ci adalah angka biner dari *ciphertext*
Ki adalah angka biner dari *key*
$\oplus$ adalah proses komputasi *XOR*

Persamaan 1 merupakan rumus cara kerja enkripsi pada algoritme enkripsi *Vernam Cipher*. Sedangkan Persamaan 2 merupakan rumus cara kerja dekripsi pada algoritme *Vernam Cipher*.

3.3. *Three-Pass Protocol*

Three-Pass Protocol merupakan protokol yang dikembangkan oleh Adi Shamir pada tahun 1980. Penjelasan singkat dari protokol ini yaitu masing-masing pengirim dan penerima pesan melakukan proses enkripsi dan proses dekripsi yang berbeda (Rachmawati, Aulya, & Budiman, 2018). Alur cara kerja dari algoritme *Three-Pass*

Protocol dapat dilihat pada Gambar 1.

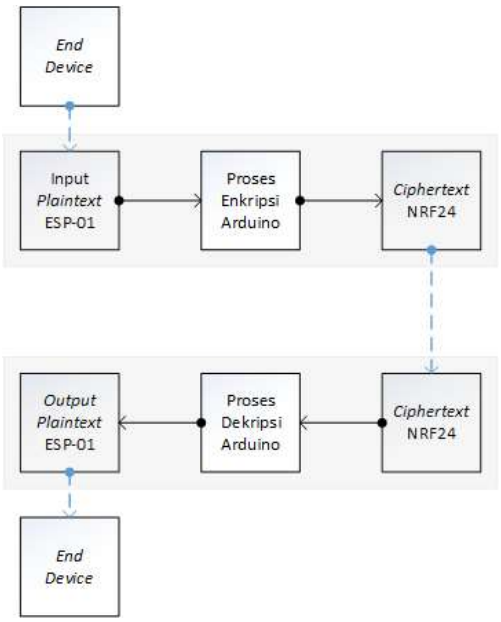


Gambar 1. Alur *Three-Pass Protocol*

4. PERANCANGAN

4.1. Gambaran Umum Sistem

Dua sistem yang dibangun berfungsi untuk melakukan *chatting* dengan cara mengirim dan menerima data secara bergantian. Data *chatting* dikirim ke ESP-01 melalui peramban situs web. Data diteruskan ke Arduino untuk dienkripsi dan diteruskan ke NRF24. NRF24 mengirim data ke sistem lain dan didekripsi oleh sistem lain hingga *plaintext* diterima oleh *End Device* yang lain. Diagram blok sistem dapat dilihat pada Gambar 2 di bawah ini.

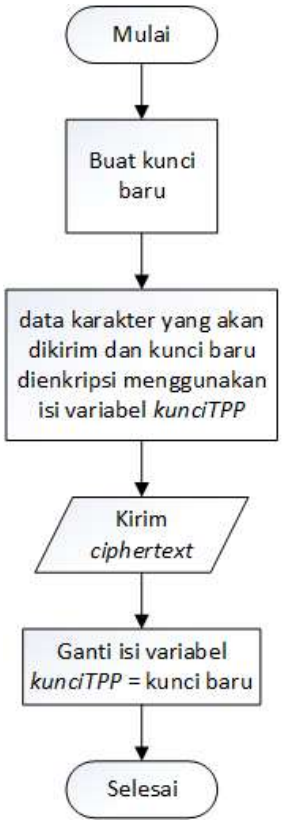


Gambar 2. Diagram blok sistem

4.2. Alur Enkripsi

Sebelum mengirim data maka sistem akan membuat kunci acak 8 karakter biner. Kunci acak tersebut akan digabung dengan 8 karakter biner dari karakter *ascii* yang akan dikirimkan, sehingga total karakter dalam setiap satu kali pengiriman yaitu 16 karakter. Proses mengirim data hanya dilakukan ketika sistem telah selesai melakukan distribusi kunci dengan *Three-Pass Protocol*. Jika sistem belum pernah melakukan distribusi kunci maka akan dilakukan proses distribusi kunci terlebih dahulu sesuai alur pada Gambar 1. Jika sistem sudah pernah melakukan proses distribusi kunci maka distribusi kunci rahasia akan dilakukan dengan cara menyisipkan kunci pada isi pesan yang dikirim sebelumnya.

Gambar 3 ditunjukkan alur distribusi kunci pada sistem pengirim yaitu mengirim data (*step* 1), menerima data (*step* 2), dan mengirim data (*step* 3). Selanjutnya alur distribusi kunci pada sistem penerima yaitu menerima data (*step* 1), mengirim data (*step* 2), dan menerima data (*step* 3). Selanjutnya, setelah kunci rahasia selesai dipakai untuk melakukan proses enkripsi maka kunci rahasia akan diganti dengan kunci acak 8 karakter sebelumnya.

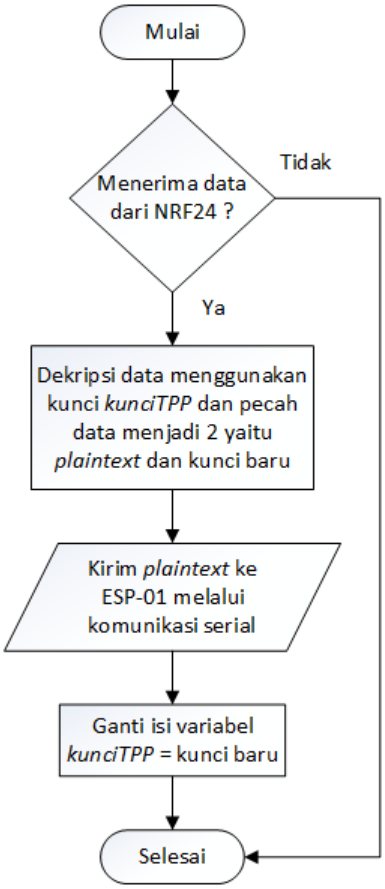


Gambar 3. Diagram alir enkripsi

4.3. Alur Dekripsi

Proses menerima data juga dilakukan jika

sistem sudah pernah melakukan distribusi kunci dengan *Three-Pass Protocol*. Setiap data dengan panjang 16 yang telah diterima akan didekripsi dan dipecah menjadi 2 bagian. Delapan karakter awal diteruskan ke modul ESP-01. Sedangkan 8 karakter selanjutnya digunakan untuk mengganti kunci rahasia saat proses dekripsi. Diagram alir dekripsi dapat dilihat pada Gambar 4.



Gambar 4. Diagram alir dekripsi

5. IMPLEMENTASI

Seluruh komponen yang ada pada kedua sistem memiliki fitur dan spesifikasi yang sama. Seluruh komponen disolder pada papan sirkuit cetak atau *Printed Circuit Board (PCB)*. Papan *PCB* tersebut dimasukkan ke dalam kotak plastik untuk meminimalisasi adanya tebaran debu atau percikan air yang mengganggu kinerja sistem.

Proses konversi antara karakter *ascii* dan biner pada saat enkripsi dan dekripsi dilakukan pada sisi *End Device* menggunakan *JavaScript*. *End Device* mengirim data ke modul ESP-01 dengan protokol komunikasi *AJAX*, sedangkan ESP-01 mengirim data ke perangkat *End Device* dengan protokol komunikasi *WebSocket*. Hasil dari implementasi perangkat keras sistem dapat dilihat pada Gambar 5.

Gambar 5. Implementasi perangkat keras

6. PENGUJIAN

6.1. *Confidentiality*

Pengujian *confidentiality* atau kerahasiaan data dilakukan untuk memastikan bahwa data *chatting* yang dikirim atau diterima oleh masing-masing sistem tidak dapat dilihat oleh sistem pihak ketiga yang tidak dikenali. Hasil pengujian *confidentiality* dapat dilihat pada Tabel 1.

Tabel 1. Hasil pengujian *confidentiality*

No	Data asli	Data *sniffing*
1	0101101100111100	1101110110111010
2	0110000111000101	0101110111111001
3	0111000011000010	1011010100000111
4	0110000110010101	1010001101010111
5	0110101111011110	1111111001001011

6.2. *Integrity*

Pengujian terhadap *integrity* atau keaslian data dilakukan untuk memastikan bahwa data *chat* yang dikirim tidak mengalami perubahan apapun. Hasil pengujian *integrity* dapat dilihat pada Tabel 2.

Tabel 2. Hasil pengujian *integrity*

No	*Hash MD5* pada file	Hasil
1	16f1074674a68ae456e4c3f5947ea60c	Valid
2	98dbd4ed9c084e56ee11a853fbb27f1c	Valid
3	052c498fe77a5c2c707ad621c25b6f33	Valid
4	d3a638b783d2dd5b01bd91f0d004d88b	Valid
5	0e2ab1f55085c58e5ef2e4ca3649c360	Valid

6.3. *Delay* Pengiriman 1 Karakter *Chat*

Pengujian *delay* pengiriman 1 karakter *chat* dilakukan untuk mengetahui hasil total waktu rata-rata yang dibutuhkan untuk melakukan 1 kali pengiriman data karakter. Hasil perhitungan waktu dimulai ketika 1 karakter telah dikonversi menjadi 8 karakter biner dan dikirim ke server. Perhitungan dihentikan ketika *client* mendapat respon dari server mengenai status pengiriman. Pada proses pengiriman dengan enkripsi, data yang dikirim oleh modul NRF24 berjumlah 16 karakter. Sedangkan proses pengiriman tanpa enkripsi, data yang dikirim oleh modul NRF24 hanya berjumlah 8 karakter. Hasil pengujian waktu total pengiriman setiap 1 karakter dapat dilihat pada Tabel 3.

Tabel 3. Total waktu pengiriman 1 karakter

No	Tanpa enkripsi	Dengan enkripsi
1	24 *ms*	23 *ms*
2	22 *ms*	23 *ms*
3	21 *ms*	27 *ms*
4	21 *ms*	23 *ms*
5	22 *ms*	22 *ms*

6.4. *Delay* Enkripsi Dan Dekripsi

Pengujian *delay* enkripsi dilakukan untuk mengetahui total waktu yang dibutuhkan untuk melakukan enkripsi pada 16 data karakter yang akan dikirim oleh modul NRF24. Total waktu yang dibutuhkan untuk proses dekripsi sama dengan proses enkripsi karena proses komputasi dan panjang data yang digunakan sama persis. Hasil pengujian waktu total enkripsi 16 karakter dapat dilihat pada Tabel 4.

Tabel 4. Total waktu enkripsi 16 karakter

No	Data	Total waktu
1	1000011010111010	1904 μs
2	0011110011111001	1908 μs
3	1100010100000111	1908 μs
4	1100001001010111	1904 μs
5	1001010101001011	1900 μs
6	1101111000001100	1896 μs
7	1101001000110000	1904 μs
8	1110001011110000	1908 μs
9	0001001001011101	1900 μs
10	0100111111011000	1900 μs

6.5. *Delay* Pengiriman NRF24

Pengujian *delay* pengiriman pada NRF24 dilakukan untuk mengetahui jumlah total waktu yang dibutuhkan oleh modul NRF24 untuk bisa mengirimkan data 16 karakter dan 8 karakter. Hasil pengujian waktu total pengiriman 16 karakter menggunakan modul NRF24 dapat dilihat pada Tabel 5. Sedangkan hasil pengujian waktu total pengiriman 8 karakter menggunakan modul NRF24 dapat dilihat pada Tabel 6.

Tabel 5. Waktu pengiriman 16 karakter pada NRF24

No	Data	Total waktu
1	1000110101101011	2624 μs
2	1101110001001011	2624 μs
3	1001110101110010	2604 μs
4	1111000111111101	2624 μs
5	0101111111001111	2628 μs

Tabel 6. Waktu pengiriman 8 karakter pada NRF24

No	Data	Total waktu
1	01011011	2408 μs
2	01100001	2404 μs
3	01110000	2436 μs
4	01100001	2404 μs
5	01101011	2400 μs

7. KESIMPULAN

Confidentiality dikatakan berhasil apabila data asli dan data *sniffing* tidak sama. *Integrity* dikatakan berhasil apabila hasil *hash* pada *file* di sisi pengirim dan sisi penerima bernilai sama. Berdasarkan beberapa hasil pengujian yang telah dilakukan dapat diambil kesimpulan bahwa sistem memiliki tingkat kerahasiaan dan keaslian data yang baik. Dengan adanya kemampuan untuk dilakukan kerahasiaan dan keaslian data maka sistem dapat membuat privasi pengguna menjadi lebih aman serta fungsional kegiatan komunikasi chatting menjadi lebih terjaga tanpa adanya kendala.

Berdasarkan 5 kali proses pengiriman data 1 karakter *chat* terenkripsi yang dilakukan oleh setiap sistem membutuhkan waktu rata-rata yaitu sekitar 23,6 *ms*. Sedangkan rata-rata pengiriman 1 karakter *chat* tanpa enkripsi dibutuhkan waktu yaitu sekitar 22 *ms*. Selain faktor komunikasi pada modul NRF24, terdapat faktor lain yang dapat mempengaruhi hasil total waktu pada saat pengiriman data seperti proses pembuatan kunci acak, komunikasi serial, dan lain sebagainya.

Berdasarkan 10 kali hasil pengujian pada proses enkripsi 16 karakter yang dilakukan oleh sistem dibutuhkan waktu rata-rata yaitu sekitar 1,9 *ms*. Berdasarkan proses pengujian pada pengiriman data menggunakan modul NRF24 dapat disimpulkan bahwa waktu rata-rata yang dibutuhkan untuk pengiriman 16 karakter yaitu sekitar 2,6 *ms* dan untuk pengiriman 8 karakter yaitu sekitar 2,4 *ms*.

Berdasarkan pembahasan kesimpulan yang telah diuraikan dapat diambil kesimpulan akhir bahwa implementasi algoritme *Vernam Cipher*

dan distribusi kunci *Three-Pass Protocol* dapat digunakan untuk mengamankan data *chatting* pada ATmega328.

DAFTAR PUSTAKA

Abdullah, A., Khalaf, R., & Riza, M. (2015). A Realizable Quantum Three-Pass Protocol Authentication Based on Hill-Cipher Algorithm. *Mathematical Problems in Engineering*.

Arduino. (2018, February 19). *Arduino Nano*. Retrieved January 2, 2019, from Arduino: https://www.arduino.cc/en/Guide/ArduinoNano

Bishop, M. (2004). *Introduction to Computer Security*. Addison-Wesley.

Gautam, et al. (2018). An Enhanced Cipher Technique Using Vigenere and Modified Caesar Cipher. *2nd International Conference on Trends in Electronics and Informatics (ICOEI)*. Kurukshetra, India. doi:10.1109/ICOEI.2018.8553910

Gowda, S. N. (2016). Innovative enhancement of the Caesar cipher algorithm for cryptography. *2nd International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Fall)*. Bareilly, India: IEEE. doi:10.1109/ICACCAF.2016.7749010

Goyal, K., & Kinger, S. (2013, July). *Modified Caesar Cipher for Better Security Enhancement, 73*.

Hassan, M., & Mohammad. (2007). Text Steganography in Chat. *IEEE*.

Nagaraj, N. (2012, April 4). One-Time Pad as a nonlinear dynamical system. *Commun Nonlinear Sci Numer Simulat, 17*.

Oktaviana, B., & Siahaan, A. P. (2016). Three-Pass Protocol Implementation in Caesar Cipher Classic Cryptography. *IOSR Journal of Computer Engineering (IOSR-JCE)*.

Rachmawati, D., Aulya, L., & Budiman, M. A. (2018). Three-pass protocol scheme for bitmap image security by using vernam cipher algorithm. *IOP Conf. Series: Materials Science and Engineering* (p. 308). IOP. doi:10.1088/1757-899X/308/1/012003

Rahim, et al. (2016, May). Design and Implementation of a Low Cost Wireless Sensor Network using Arduino and nRF24L01(+). *International Journal of Scientific Research Engineering & Technology (IJSRET)*, *5*(5), 307.

Stallings, W. (2005). *Cryptography and Network Security Principles and Practices* (4 ed.). Prentice Hall.

Uchôa, et al. (2007). A Three-Pass Protocol for Cryptography. *IEEE*.