JURNAL MEDIA INFORMASI TEKNOLOGI
Volume 2, No. 2, Oktober 2025
DOI: https://doi.org/10.69616/mit.v2i2.248

One-Time Pad Cryptography for Secure Data Transmission in IoT Smart Door Using QR Code

Muhammad Dito Asrofa*, Syamsul Bahri, Kasliono
Universitas Tanjungpura, Indonesia.
*Corresponding Author. E-mail: h1051211005@student.untan.ac.id

Article History
Received:
Sep 22nd, 2025

Revised:
Oct 9th, 2025

Accepted:
Oct 10th, 2025

Keywords
Internet of Things; Smart Door; One-Time Pad; XOR Algorithm; Data Encryption.

ABSTRACT

The increasing use of the Internet of Things (IoT) in security systems such as Smart Doors has created new challenges for data security, especially the risk of wiretapping through sniffing attacks. This research proposes applying the One-Time Pad (OTP) XOR algorithm as an Encryption method to protect QR Code-based data transmission in the Smart Door system. The implementation is carried out on three main communication paths: sending UUID from the server to the user's website, sending the results of QR Code scanning from ESP32-CAM to the server, and sending instructions from the server to the ESP32 device. The test results show that the resulting ciphertext is always different even though the plaintext is the same, with a 0% algorithm identification success rate by Cipher Identifier and a 100% resistance level to brute force XOR, based on testing using dCode.fr tools. In addition, the Encryption and Decryption processes are very fast, with an average Encryption time on the ESP32-CAM of 0.34 milliseconds and an average Decryption time on the ESP32 of 0.17 milliseconds. These results show that the OTP XOR algorithm is able to disguise data against basic cryptanalysis attacks and can be run on IoT devices that have limited resources. In the future, it is suggested to apply better key management methods such as pre-shared key (PSK), key rotation, or Key Derivation Function (KDF) to improve the security of key distribution in this symmetrical system. In addition, the security system can be improved through separating the OTP key transmission path using an approach such as Out-of-Band Key Exchange or asymmetric key wrapping with the RSA algorithm so that the key remains protected even if sniffing occurs.

PENDAHULUAN

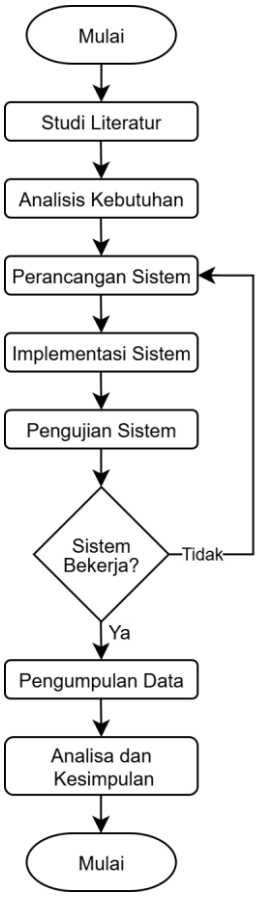
Internet of Things (IoT) memungkinkan perangkat fisik terhubung ke internet [1] untuk bertukar data [2] dan menjalankan fungsi secara otomatis [3]. Teknologi ini telah meluas ke berbagai bidang, salah satunya dalam sistem keamanan seperti *Smart Door*, yang menawarkan kenyamanan dalam mengelola akses pintu [4]. Pada sistem ini, *QR Code* umum digunakan sebagai metode autentikasi yang mengandung informasi unik dan dapat diverifikasi melalui server [5].

Meskipun menawarkan kemudahan, sistem IoT menghadapi tantangan keamanan, terutama risiko serangan *Man-in-the-Middle* (MitM) dan sniffing karena kurangnya enkripsi pada komunikasi antarperangkat [6]. Serangan ini memungkinkan pihak tidak berwenang menyadap data sensitif seperti UUID, PIN, atau perintah sistem. Untuk mengatasi masalah tersebut, diperlukan algoritma kriptografi ringan yang cocok untuk perangkat IoT seperti ESP32. Salah satu yang paling sesuai adalah algoritma *One-Time Pad* (OTP) berbasis operasi XOR, yang menggunakan kunci acak sekali pakai dengan panjang yang sama dengan pesan asli. Keunggulan algoritma ini adalah operasi XOR yang sangat ringan, membuatnya efisien untuk perangkat dengan sumber daya terbatas [7].

Meskipun demikian, implementasi kriptografi pada perangkat IoT dapat meningkatkan penggunaan CPU dan waktu proses [8]. Oleh karena itu, penelitian ini berfokus pada penerapan algoritma OTP XOR untuk mengamankan tiga jalur komunikasi utama pada sistem *Smart Door*: dari server ke aplikasi pengguna, dari ESP32-CAM ke server, dan dari server ke ESP32. Penelitian ini juga mengukur waktu proses enkripsi dan dekripsi untuk menilai keseimbangan antara keamanan dan performa sistem. Tujuan utama penelitian ini adalah membuktikan efektivitas OTP XOR dalam melindungi data dari serangan siber dan mengukur efisiensinya pada perangkat IoT.

METODE

Metodologi penelitian ini mencakup sejumlah langkah, dimulai dari studi literatur, analisis kebutuhan, perancangan sistem, implementasi, pengujian sistem, hingga analisis data dan penarikan kesimpulan [9]. Diagram metode penelitian dapat dilihat pada Gambar 1.



Gambar 1. Metode Penelitian

1. Studi Literatur
   Tahapan ini dilakukan dengan mengumpulkan dan menganalisis referensi terkait teknologi IoT, sistem *Smart Door*, serta algoritma kriptografi *One-Time Pad* (OTP) XOR. Studi literatur bertujuan untuk memahami karakteristik sistem, metode verifikasi, serta mekanisme enkripsi yang sesuai dengan keterbatasan perangkat mikrokontroler, sehingga penelitian berjalan sesuai ruang lingkup yang ditetapkan.
2. Analisis Kebutuhan
   Langkah analisis kebutuhan dilakukan untuk memahami kebutuhan yang harus dipenuhi dalam penelitian ini. Analisis ini difokuskan agar sistem *Smart Door* yang dirancang sesuai dengan tujuan penelitian dan tetap berada dalam lingkup yang telah ditentukan. Tahapan ini mencakup

identifikasi kebutuhan perangkat keras seperti ESP32-CAM, NodeMCU ESP32, dan solenoid doot lock, serta kebutuhan perangkat lunak seperti Laravel, Arduino IDE, dan Wireshark.

2.1 Kebutuhan Perangkat Keras

Berikut merupakan komponen perangkat keras yang digunakan dalam sistem *Smart Door* berbasis IoT:

a) NodeMCU ESP32
   NodeMCU ESP32 adalah mikrokontroler berbasis chip yang mendukung konektivitas Wi-Fi, dilengkapi dengan beberapa pin I/O, serta dapat diprogram menggunakan Arduino IDE. Modul ini bertindak sebagai pusat kendali dalam sistem untuk mengatur relay, *buzzer*, dan *solenoid door lock* [10].
b) ESP32-CAM
   ESP32-CAM adalah platform berbasis ESP32 dengan modul kamera dan konektivitas Wi-Fi, yang digunakan untuk pemindaian *QR Code* serta pengiriman data verifikasi ke server [11].
c) *Solenoid door lock*
   Solenoid door lock berfungsi sebagai aktuator linier untuk membuka dan menutup pintu. Bekerja pada mode *Normally Closed* (NC), solenoid ini akan aktif ketika dialiri arus 12V DC dan dikendalikan oleh relay [12].
d) Relay
   Relay merupakan saklar elektronik yang digunakan untuk mengontrol aliran daya dari mikrokontroler ke aktuator yaitu *solenoid door lock*. Modul ini bekerja berdasarkan prinsip elektromagnetik dan memungkinkan sistem dengan tegangan rendah (seperti ESP32) untuk mengendalikan perangkat bertegangan tinggi [13].
e) *Buzzer*
   Buzzer merupakan komponen output yang berfungsi sebagai indikator suara untuk memberi peringatan. Dalam sistem ini, *buzzer* diaktifkan ketika verifikasi pengguna gagal sebagai tanda akses ditolak. *Buzzer* bekerja dengan mengubah getaran elektromagnetik menjadi suara, dan biasanya digunakan dalam sistem alarm atau notifikasi berbasis mikrokontroler [14].

2.2 Kebutuhan Perangkat Lunak

Adapun kebutuhan perangkat lunak dalam membangun sistem diantara lain:

a) Laravel
   Laravel merupakan framework aplikasi web berbasis PHP dengan struktur MVC (Model-View-Controller) yang kuat. Laravel menyediakan RESTful API yang digunakan sebagai backend server untuk mengelola komunikasi antara perangkat ESP32/ESP32-CAM dengan antarmuka pengguna, termasuk dalam pengiriman data verifikasi dan kontrol akses [15].
b) Arduino IDE
   Arduino IDE adalah perangkat lunak yang digunakan untuk menulis, memprogram, dan mengunggah kode ke mikrokontroler yaitu NodeMCU ESP32 dan ESP32-CAM. IDE ini berbasis Java dan mendukung bahasa C/C++, dirancang agar mudah digunakan dalam pengembangan sistem berbasis IoT [16].
c) Ettercap
   Ettercap adalah tool open-source untuk keamanan jaringan yang digunakan dalam simulasi serangan *Man-in-the-Middle* (MITM). Tool ini memungkinkan penyadapan dan analisis lalu lintas data antarperangkat dalam jaringan local [17].
d) Wireshark
   Wireshark digunakan sebagai analyzer lalu lintas jaringan. Perangkat lunak ini memungkinkan peneliti melihat paket data secara real-time untuk memverifikasi apakah

data yang dikirim antarperangkat telah terenkripsi dengan benar selama proses transmisi [18].

2.2 Kebutuhan Perangkat Lunak

One-Time Pad (OTP) merupakan algoritma kriptografi simetris berbasis aliran (*stream cipher*) yang melakukan proses enkripsi dan dekripsi satu karakter pada satu waktu. Algoritma ini pertama kali diperkenalkan oleh Major Joseph Mauborgne pada tahun 1917 sebagai penyempurnaan dari algoritma Vernam Cipher dengan tujuan meningkatkan tingkat keamanan [19].

OTP menggunakan pad berisi deretan karakter kunci acak, yang hanya digunakan satu kali untuk mengenkripsi satu pesan. Setelah digunakan, pad harus dihancurkan agar tidak dapat dipakai kembali. Kunci yang digunakan harus sepanjang pesan dan bersifat benar-benar acak untuk menjaga keamanan. Karena itulah, OTP dianggap sebagai algoritma yang secara matematis tidak dapat dipecahkan apabila diterapkan dengan benar [20].

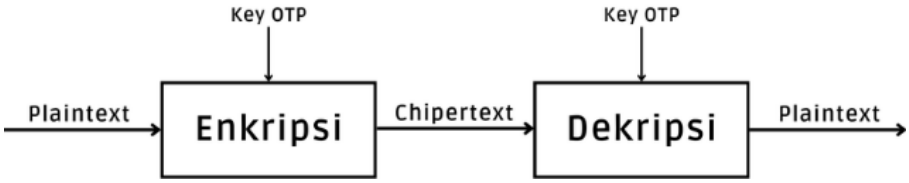
OTP bekerja dengan operasi logika XOR (Exclusive OR), di mana hasil operasi akan bernilai TRUE (1) jika dua bit berbeda, dan FALSE (0) jika sama. Panjang pesan (P) dan kunci (K) harus identik. Proses enkripsi dilihat pada Persamaan 1.

$$C = P \oplus K \tag{1}$$

Dari Persamaan 2, *ciphertext* (C) diperoleh dengan melakukan operasi XOR antara *plaintex*t (P) dan kunci (K). Untuk mendapatkan kembali pesan asli, proses dekripsi dilakukan dengan Persamaan 2.

$$P = C \oplus K \tag{2}$$

Dengan menggunakan kunci yang sama, operasi XOR antara *ciphertext* dan kunci akan menghasilkan kembali nilai *plaintex*t yang asli [21]. Keamanan OTP sangat bergantung pada keacakan kunci dan penggunaannya yang hanya satu kali. Gambaran umum algoritma OTP dapat dilihat pada Gambar 2.

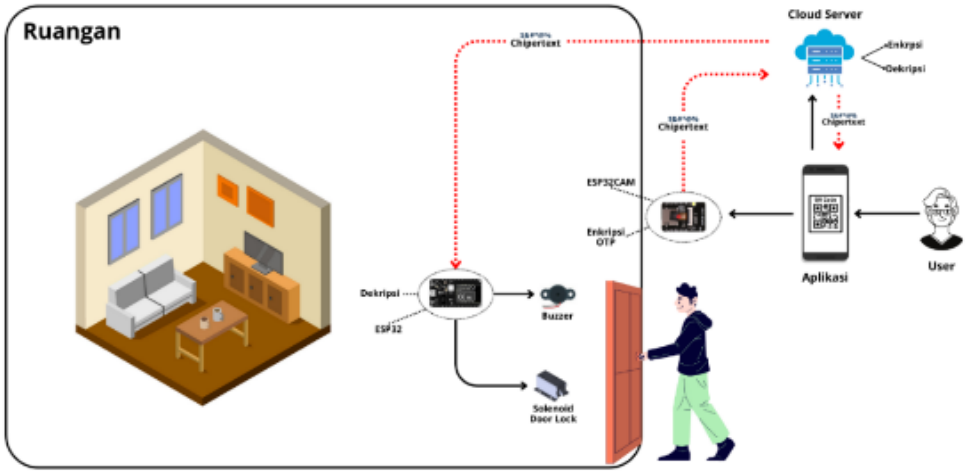


Gambar 2. Algoritma OTP secara umum

3. Perancangan Sistem

Perancangan sistem dalam penelitian ini melibatkan integrasi antara perangkat keras dan perangkat lunak dalam arsitektur sistem *Smart Door* berbasis Internet of Things (IoT). Pada sisi perangkat keras, sistem memanfaatkan ESP32-CAM untuk melakukan pemindaian *QR Code* yang berisi data untuk verifikasi pengguna. Data hasil pemindaian ini dikirimkan ke cloud server untuk dilakukan proses verifikasi. Selanjutnya, NodeMCU ESP32 berperan sebagai pengendali aktuator, yaitu modul relay, *solenoid door lock*, dan *buzzer*, yang akan diaktifkan berdasarkan hasil autentikasi yang diambil dari server.

Sementara itu, pada sisi perangkat lunak, sistem dibangun dengan memanfaatkan Arduino IDE untuk memprogram kedua mikrokontroler serta framework Laravel sebagai backend server. Komunikasi antara perangkat dan server dilakukan melalui API, dengan pengamanan menggunakan algoritma *One-Time Pad* (OTP) XOR. Enkripsi ini digunakan untuk melindungi data autentikasi seperti UUID, hasil pemindaian *QR Code*, dan instruksi sistem agar tidak mudah dibaca oleh pihak ketiga. Ilustrasi lengkap dari arsitektur sistem dapat dilihat pada Gambar 3.



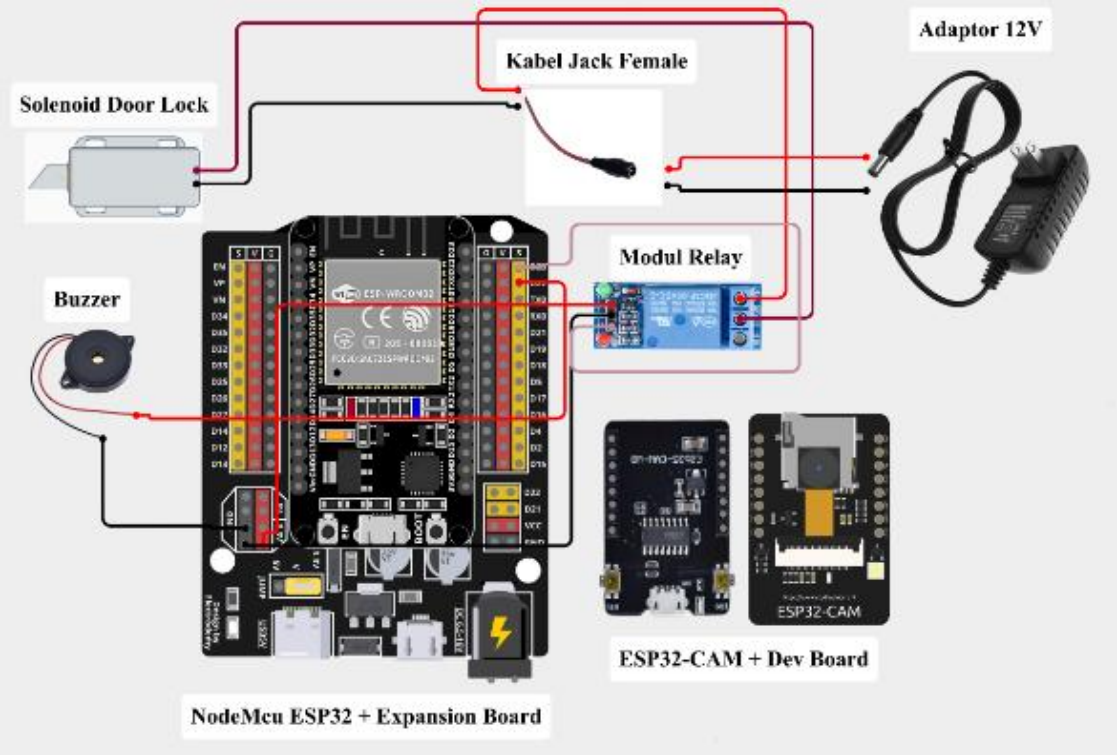
Gambar 3. Arsitektur Sistem

4. Implementasi Sistem
   Tahap implementasi dilakukan berdasarkan rancangan sistem yang telah disusun sebelumnya, dimulai dari perakitan perangkat keras hingga penanaman program perangkat lunak. ESP32-CAM digunakan sebagai perangkat utama untuk membaca *QR Code*, sedangkan NodeMCU ESP32 bertindak sebagai pengendali utama yang menangani komunikasi dengan server dan pengoperasian aktuator. Setelah seluruh perangkat keras selesai dirakit, perangkat lunak diimplementasikan dengan menanamkan program ke masing-masing mikrokontroler. Program mencakup proses pembacaan data *QR Code*, pengiriman data yang telah dienkripsi menggunakan algoritma *One-Time Pad* (OTP) XOR, serta verifikasi di sisi server.
5. Pengujian Sistem
   Setelah sistem *Smart Door* berbasis IoT selesai dibangun, dilakukan pengujian untuk memastikan bahwa sistem berfungsi sesuai dengan perancangan. Pengujian dilakukan dalam dua aspek utama, yaitu pengujian keamanan transmisi data dan pengukuran waktu proses enkripsi serta dekripsi. Pengujian keamanan dilakukan dengan mengamati lalu lintas data dalam jaringan untuk memastikan bahwa informasi sensitif seperti UUID dan instruksi sistem tidak dapat dikenali oleh pihak yang tidak berwenang. Selanjutnya dilakukan uji ketahanan *ciphertext* dengan mencoba mengenali metode enkripsi yang digunakan, serta melakukan analisis menggunakan pendekatan brute force untuk mengevaluasi daya tahan *ciphertext* terhadap upaya peretasan menggunakan tools kriptografi yang umum tersedia. Selain itu, pengujian juga mencakup pengukuran waktu proses untuk mengetahui durasi enkripsi dan dekripsi yang terjadi di setiap jalur komunikasi, guna menilai berapa lama waktu yang diperlukan algoritma OTP XOR dalam sistem *Smart Door* berbasis IoT.
6. Pengumpulan Data
   Pada tahap ini, data dikumpulkan dari hasil pengujian sistem setelah implementasi algoritma *One-Time Pad* (OTP) XOR pada sistem *Smart Door* berbasis IoT. Data yang diperoleh mencakup hasil pengamatan terhadap komunikasi data melalui simulasi sniffing, hasil uji ketahanan *ciphertext* menggunakan tools analisis kriptografi, serta hasil pengukuran waktu proses enkripsi dan dekripsi di setiap jalur komunikasi antar perangkat. Pengumpulan data ini bertujuan untuk mengevaluasi sejauh mana algoritma OTP XOR mampu menjaga kerahasiaan informasi serta menilai efisiensi proses kriptografi pada perangkat dengan sumber daya terbatas seperti ESP32 dan ESP32-CAM.
7. Analisa dan Kesimpulan
   Tahap akhir dari penelitian ini adalah melakukan analisa dan menarik kesimpulan berdasarkan hasil pengujian sistem dan data yang telah dikumpulkan. Kesimpulan digunakan untuk menjawab pertanyaan yang telah dirumuskan sebelumnya dalam perumusan masalah, khususnya terkait penerapan algoritma *One-Time Pad* (OTP) XOR dalam menjaga keamanan transmisi data pada sistem *Smart Door* berbasis IoT. Berdasarkan temuan-temuan tersebut, saran juga diberikan sebagai masukan untuk pengembangan sistem yang lebih baik di masa mendatang.

HASIL DAN PEMBAHASAN

Implementasi Perangkat Keras

Pada tahap implementasi perangkat keras, sistem *Smart Door* berbasis IoT dirancang menggunakan beberapa komponen utama, yaitu ESP32-CAM sebagai pemindai *QR Code* dan NodeMCU ESP32 sebagai pengendali utama. ESP32-CAM dipasang di dekat pintu untuk membaca *QR Code* dari pengguna, sementara NodeMCU ESP32 bertugas menerima hasil verifikasi dari server dan mengontrol aktuator yang terdiri dari relay, *solenoid door lock*, dan *buzzer*. Adapun gambar skema rancangan perangkat keras yang menggambarkan keseluruhan hubungan antar komponen dapat dilihat pada Gambar 4.



Gambar 4. Wiring Diagram Perangkat Keras

Dalam skema tersebut, NodeMCU ESP32 mengendalikan relay untuk membuka atau menutup kunci pintu, serta mengaktifkan *buzzer* sebagai notifikasi apabila verifikasi gagal. Penggunaan adaptor 12V diperlukan untuk menyuplai daya ke *solenoid door lock* yang memerlukan arus lebih besar dibandingkan komponen lainnya.

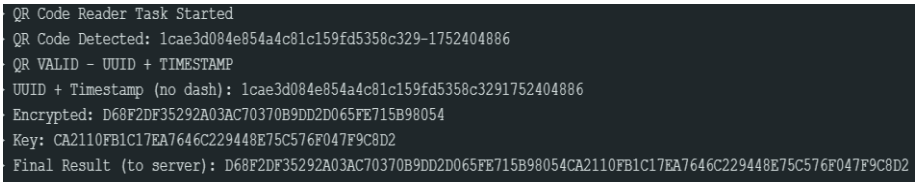
Konfigurasi hubungan pin pada NodeMCU ESP32 dengan relay dan *buzzer* dijelaskan dalam Tabel 1.

Tabel 1. Hubungan Pin Perangkat Keras

Komponen	Pin
ESP32	3v ke Relay VCC
ESP32	GND ke Relay GND
ESP32	D23 ke Relay IN
ESP32	GND ke *Buzzer*
ESP32	D22 ke *Buzzer*
Relay	COM ke Kabel merah solenoid
Relay	NO ke Kabel merah jack female
Solenoid	Kabel Hitam ke Kabel Hitam jack female

Implementasi Perangkat Lunak

Implementasi perangkat lunak dilakukan pada tiga komponen utama, yaitu ESP32-CAM, server, dan NodeMCU ESP32. Sistem ini dirancang untuk mengelola proses verifikasi *QR Code* serta mengendalikan perangkat keras seperti *solenoid door lock* dan *buzzer* berdasarkan hasil verifikasi yang telah terenkripsi menggunakan algoritma *One-Time Pad* (OTP) XOR. Pada perangkat ESP32-CAM, dilakukan pemindaian *QR Code* untuk mengekstrak UUID pengguna. Setelah berhasil dibaca, UUID dikirimkan ke server melalui jaringan Wi-Fi. Proses ini ditunjukkan pada Gambar 5.

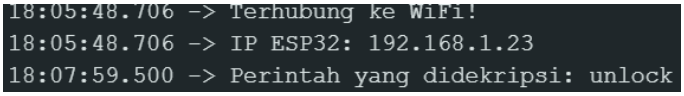


Gambar 5. Pembacaan *QR Code* pada ESP32-CAM

Setelah menerima UUID dari ESP32-CAM, server akan memverifikasi kecocokan UUID terhadap database pengguna. Jika UUID valid, maka server menyimpan perintah "unlock" dalam bentuk *ciphertext* menggunakan algoritma OTP XOR. Jika tidak valid, maka server menyimpan perintah "*buzzer*" untuk penolakan akses. Kemudian server juga akan melakukan verifikasi timestamp apabila masih dalam waktu 30 detik maka *QR Code* nya dianggap valid dan jika lebih dari 30 Detik maka *QR Code* nya dianggap sudah expired atau tidak valid. Untuk respon proses ini dapat dilihat pada Gambar 6.

Gambar 6. Respon Verifikasi dari Server

Perangkat NodeMCU ESP32 kemudian secara berkala mengambil perintah dari server dengan metode client-pull menggunakan REST API. Perintah yang diterima akan didekripsi menggunakan metode OTP XOR. Jika hasil dekripsi berupa "unlock", maka relay akan diaktifkan untuk membuka kunci pintu. Jika hasilnya "*buzzer*", maka *buzzer* akan menyala untuk memberikan notifikasi akses ditolak. Untuk hasil perintah yang diterima dapat dilihat pada. Gambar 7.



Gambar 7. Perintah yang diterima dari Server

Implementasi Algoritma OTP XOR

Implementasi algoritma *One-Time Pad* (OTP) XOR pada sistem *Smart Door* berbasis IoT bertujuan untuk mengamankan komunikasi data pada tiga jalur utama: pengiriman UUID dari server ke website pengguna, pengiriman hasil pemindaian *QR Code* dari ESP32-CAM ke server, dan pengiriman instruksi perintah dari server ke ESP32.
Contoh Proses Enkripsi dan Dekripsi

Data *Plaintex*t:
1cae3d084e854a4c81c159fd5358c3291752043661
Kunci OTP :
f7d8a9215a369cbee2a1ef83c69b379fbf44137a4b

A. Konversi Hex ke Biner 8-bit

Untuk memulai proses enkripsi, data *plaintex*t dalam bentuk hex dikonversi ke biner 8-bit untuk dilakukan operasi XOR untuk konversi hex ke biner-8bit dapat dilihat pada Tabel 2.

Tabel 2. Konversi Hex ke Biner

Index	*Plaintext* Hex	*Plaintext* Biner	*Key* Hex	*Key* Biner
0	1c	00011100	f7	11110111
1	ae	10101110	d8	11011000
...	...	...	...	...
20	61	01100001	4b	01001011

B. XOR *Byte* per *Byte*

Kemudian dilakukan operasi *byte* per *byte*. Untuk perhitungan pada *byte* ke-0 dapat dilihat pada Tabel 3.

Tabel 3. Perhitungan Pada *Byte* 0

Bit Pos	*Plaintext* Bit	*Key* Bit	XOR Result	Keterangan
7	0	1	1	0 XOR 1 = 1
6	0	1	1	0 XOR 1 = 1
...	...	...	...	...
0	0	1	1	0 XOR 1 = 1

Kemudian untuk perhitungan pada *byte* ke-1 dapat dilihat pada Tabel 4.

Tabel 4. Perhitungan Pada *Byte* 1

Bit Pos	*Plaintext* Bit	*Key* Bit	XOR Result	Keterangan
7	1	1	0	1 XOR 1 = 0
6	0	1	1	0 XOR 1 = 1
...	...	...	...	...
0	0	0	0	0 XOR 0 = 0

C. Hasil XOR Lengkap

Selanjutnya untuk hasil xor lengkap semua *byte* dapat dilihat pada Tabel 5.

Tabel 5. Hasil XOR Lengkap

Index	*Plaintext* Biner	*Key* Biner	XOR Result Biner	XOR Result Hex
0	00011100	11110111	11101011	eb
1	10101110	11011000	01110110	76
...	...	...	...	...
20	01100001	01001011	00101010	2a

Setelah seluruh *byte* selesai di-XOR-kan, diperoleh *ciphertext*: eb76942914b3d6f26360b67e95c3f4b6a816174c2a

D. Proses Dekripsi

Dekripsi dilakukan dengan operasi XOR kembali antara *ciphertext* dan *key* pada Tabel 6 berikut:

Tabel 6. *ciphertext* dan *key*

Ciphertext	*Key*
eb76942914b3d6f26360b67e95c3f4b6a816174c2a	f7d8a9215a369cbee2a1ef83c69b379fbf44137a4b

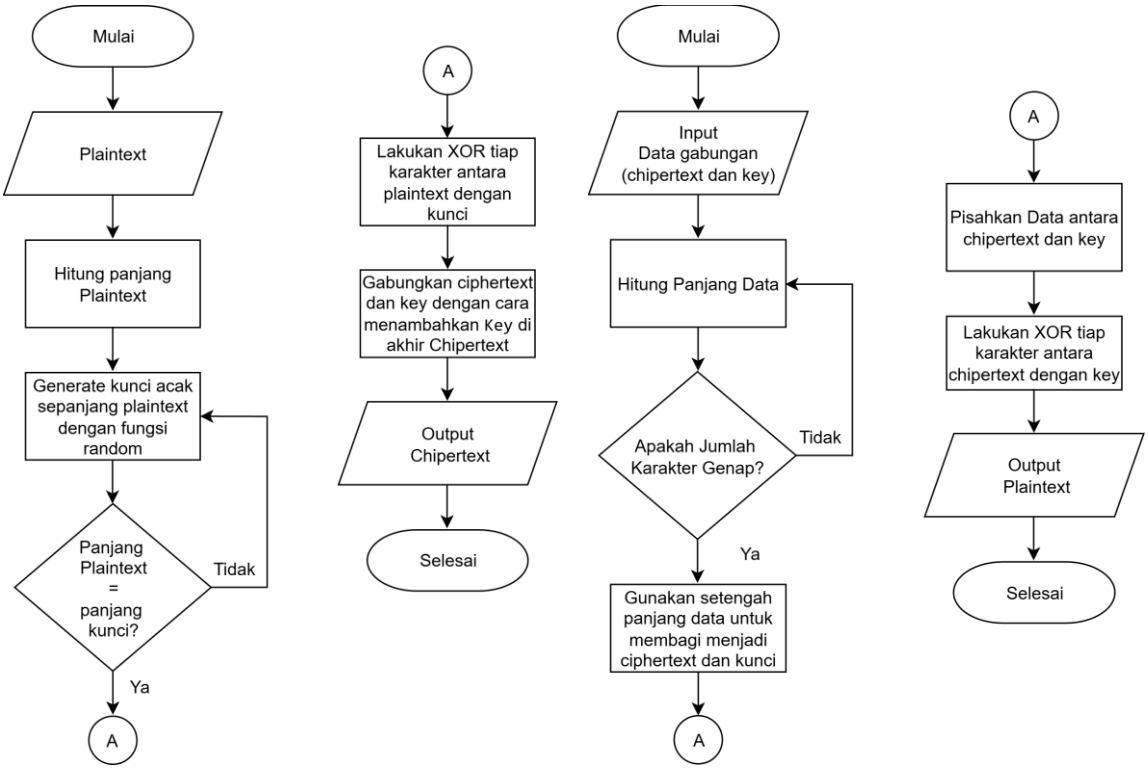
Untuk proses dekripsi dapat dilihat pada Tabel 7 berikut ini.

Tabel 6. Proses Dekripsi

Index	*Ciphertext* Hex	*Key* Hex	XOR Result Hex
0	eb	f7	1c
1	76	d8	ae
...	...	...	....
20	2a	4b	61

Hasil Dekripsi Final:
1cae3d084e854a4c81c159fd5358c3291752043661

Setiap komunikasi data dilakukan dengan operasi XOR antara *plaintex*t dengan *key* acak yang panjangnya sama. Kunci ini dihasilkan secara acak setiap kali proses enkripsi berlangsung. Diagram alir proses enkripsi OTP XOR dapat dilihat pada Gambar 8a.



(a) (b)

Gambar 8. Proses Enkripsi (a) dan Deskripsi (b)

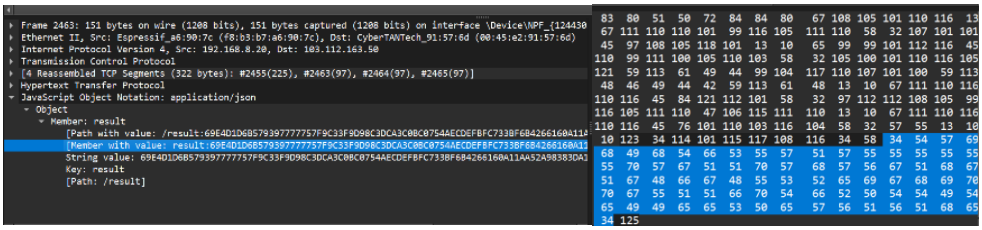
Proses dimulai dari *plaintex*t sebagai data asli. Selanjutnya dilakukan perhitungan panjang *plaintex*t. Berdasarkan panjang tersebut, sistem akan menghasilkan kunci acak menggunakan fungsi random. Setelah itu dilakukan pengecekan, apakah panjang *plaintex*t sama dengan panjang kunci. Jika sama, proses dilanjutkan dengan operasi XOR setiap karakter *plaintex*t dengan kunci. Hasil XOR kemudian digabungkan dengan kunci sehingga membentuk *ciphertext*. Terakhir, sistem menghasilkan output *ciphertext* sebagai data terenkripsi. Diagram alir proses Dekripsi OTP XOR dapat dilihat pada Gambar 8b.

Proses dekripsi dimulai dari input data gabungan berupa *ciphertext* dan *key*. Selanjutnya dilakukan perhitungan panjang data untuk menentukan titik pemisah. Berdasarkan panjang tersebut, sistem memisahkan *ciphertext* dan *key*. Setelah dipisahkan, *ciphertext* kemudian didekripsi dengan melakukan operasi XOR ulang setiap karakter *ciphertext* dengan *key*. Hasil dari proses XOR ini akan menghasilkan kembali *plaintex*t sebagai data asli.

Pengujian Algoritma OTP XOR

Pengujian ini melihat efektivitas OTP XOR dalam mengamankan transmisi data

1. Pengujian *Sniffing*
   Simulasi serangan sniffing menggunakan Wireshark menunjukkan bahwa meskipun data berhasil disadap, informasi yang ditangkap adalah *ciphertext* yang tidak dapat dibaca, seperti yang terlihat pada Gambar 9.



Gambar 9. Hasil Tangkapan Wireshark

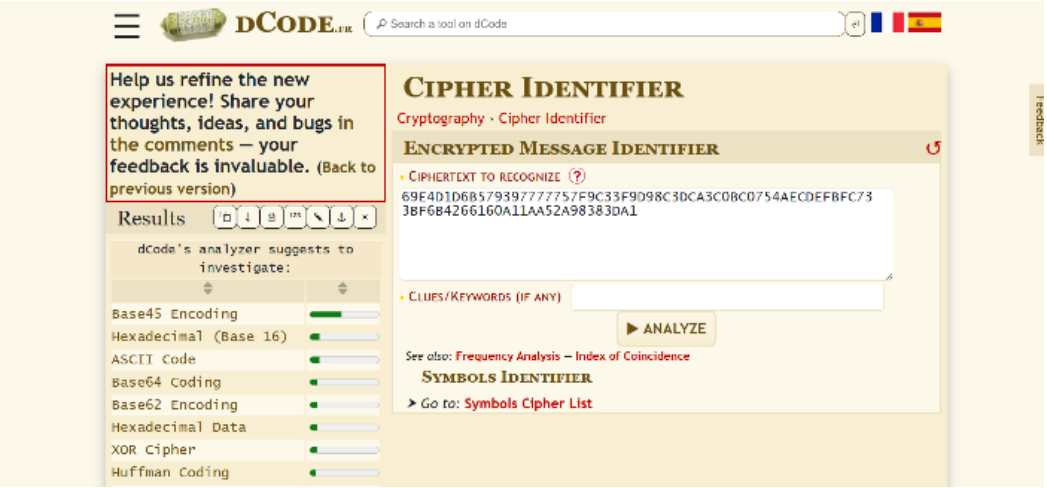
Dapat dilihat dari Gambar 9 bahwa data yang di transmisikan dalam sistem berhasil di sniffing, tetapi data yang di tangkap sudah dalam bentuk *ciphertext* sehingga penyerang tidak tahu bahwa data tersebut adalah data apa.

2. Pengujian Ketahanan *Ciphertext*

Untuk menguji ketahanan *ciphertext* terhadap teknik kriptanalisis, digunakan tools dCode.fr dengan fitur Cipher Identifier dan XOR Cipher *Decrypt*ion. Tujuan dari pengujian ini adalah untuk mengetahui apakah *ciphertext* yang ditangkap dapat dikenali atau didekripsi menggunakan metode *brute-force* public.

a) Pengujian dengan *Chiper Identifier*

Pada pengujian dengan *Cipher Identifier*, *ciphertext* dari hasil pemindaian *QR Code* akan dimasukkan pada menu chiper identifier dan kemudian di analisis. Kemudian dcode akan menampilkan beberapa metode yang memiliki kecocokan dengan *ciphertext* yang diberikan. Untuk pengujian ketahanan dengan chiper identifier dilakukan pada Gambar 10.

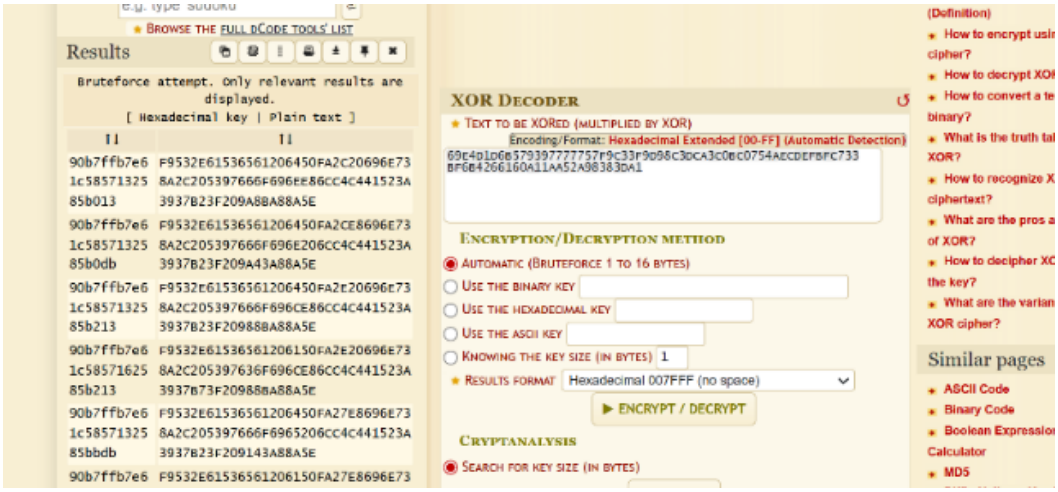


Gambar 10. Pengujian *Cipher Identifier*

Setelah dilakukan percobaan sebanyak 10 kali maka didapat hasilnya menunjukkan bahwa 100% dari *ciphertext* (10/10 data) hanya dikenali sebagai format umum seperti Hexadecimal, Base45 Encoding, atau ASCII Code, tanpa ada satu pun yang terdeteksi sebagai hasil enkripsi menggunakan algoritma OTP XOR. Hal ini mengindikasikan bahwa Cipher Identifier gagal mengenali pola enkripsi OTP karena *ciphertext* yang dihasilkan OTP benar-benar acak.

b) Pengujian dengan XOR Chiper

Selanjutnya, pengujian dilakukan menggunakan fitur XOR Cipher pada situs yang sama. *Ciphertext* hasil sniffing dari komunikasi *QR Code* akan dimasukkan ke menu xor cipher yang kemudian memilih bagian menu automatic (Bruteforce 1 to 16 *byte*s). Setelah itu tekan tombol *Encrypt* / *Decrypt* maka *ciphertext* dianalisis dengan teknik *brute-force* dengan kunci dari 1 sampai 16 *byte*s. Untuk pengujian ketahanan dengan chiper identifier dilakukan pada Gambar 11.



Gambar 11. Pengujian XOR *Chiper*

Setelah dilakukan percobaan sebanyak 10 kali seluruh *ciphertext* tidak berhasil didekripsi, yang berarti tingkat kegagalan dekripsi mencapai 100% (10/10 data). Tidak ditemukan *plaintex*t yang bermakna atau mendekati isi pesan asli, yang memperkuat bahwa implementasi OTP XOR dengan kunci acak sepanjang 16 *byte* dan hanya digunakan satu kali (one-time) memiliki tingkat ketahanan 100% terhadap metode *brute-force* XOR standar.

3. Pengujian Waktu Proses Enkripsi dan Dekripsi
Pengujian ini bertujuan untuk mengukur durasi waktu proses enkripsi dan dekripsi pada masing-masing jalur komunikasi sistem. Pengambilan data dilakukan dengan mencatat waktu menggunakan fungsi penghitung internal di masing-masing perangkat.

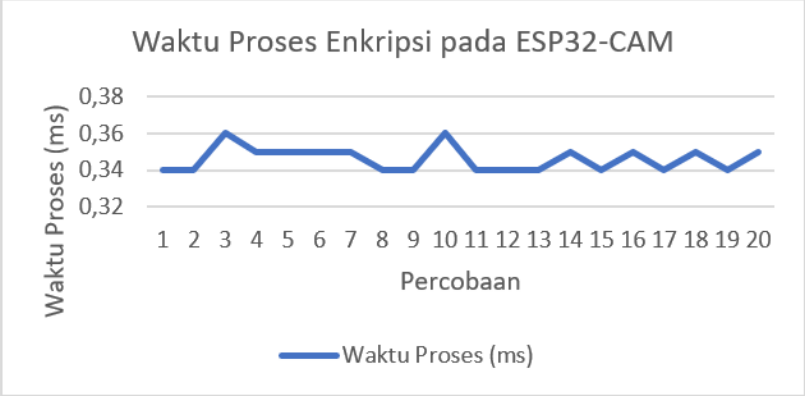
a) Server ke Website Pengguna
Pengujian dilakukan untuk mengukur waktu enkripsi oleh server dan waktu dekripsi oleh aplikasi web pengguna terhadap UUID. Waktu ini dihitung sejak UUID diproses untuk dienkripsi dan didekripsi di sisi pengguna. Untuk hasilnya dapat dilihat pada Tabel 7.

Tabel 7. Pengujian Waktu Enkripsi dan Dekripsi Data Autentikasi

No	Waktu Enkripsi (ms)	Waktu Dekripsi (ms)
1	0.01	0.01
2	0.01	0.01
3	0.01	0.01
4	0.01	0.01
5	0.01	0.01
...	...	...
20	0.01	0.01
Rata-Rata	0.011 ms	0.010 ms

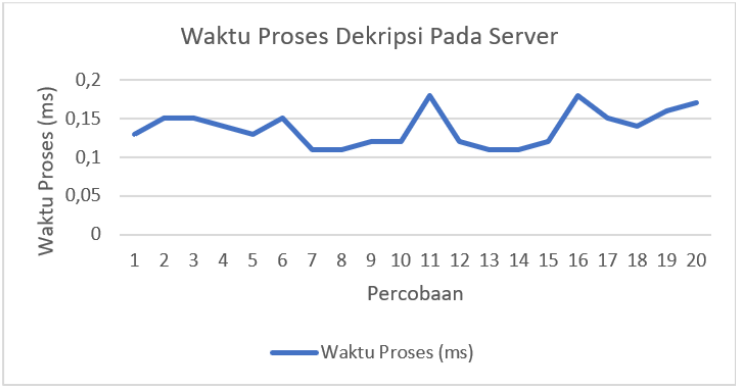
Dari Tabel 7, terlihat bahwa proses enkripsi dan dekripsi pada jalur ini cukup cepat dan stabil. Rata-rata waktu enkripsi di server adalah 0,011 ms, sedangkan waktu dekripsi di sisi pengguna adalah 0,010 ms

b) ESP32-CAM ke Server
Pengujian ini dilakukan untuk mengukur waktu enkripsi pada ESP32-CAM saat memproses hasil pembacaan *QR Code*, serta waktu dekripsi oleh server setelah data diterima. Untuk gambar grafik waktu proses enkripsi pada ESP32-CAM dapat dilihat pada Gambar 12.



Gambar 12. Grafik Waktu Enkripsi di ESP32-CAM

Gambar 12 menunjukkan grafik waktu proses enkripsi pada ESP32-CAM. Terlihat bahwa waktu proses enkripsi relatif stabil dan berkisar antara 0.34 ms hingga 0.36 ms selama 20 kali percobaan. Selanjutnya untuk gambar grafik waktu proses dekripsi pada Server dapat dilihat pada Gambar 13.

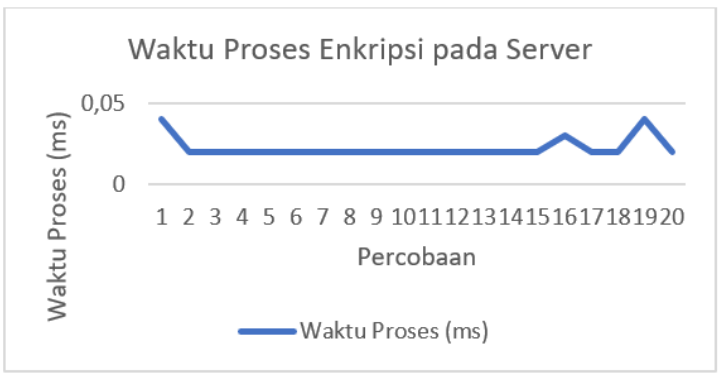


Gambar 13. Grafik Waktu Dekripsi di Server

Selanjutnya, Gambar 13 menunjukan grafik waktu proses dekripsi yang dilakukan oleh server setelah menerima data dari ESP32-CAM. Berdasarkan grafik, proses dekripsi oleh server berada pada kisaran waktu 0.11 ms hingga 0.18 ms, dengan sebagian besar percobaan menunjukkan kestabilan pada nilai 0.13 ms hingga 0.15 ms.

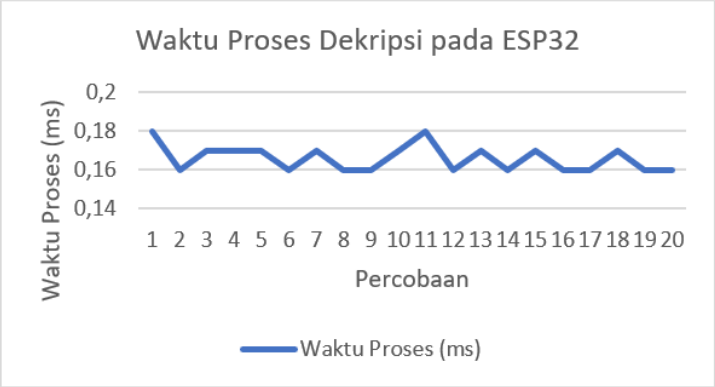
c) Server ke ESP32

Pengujian ini bertujuan untuk mengetahui seberapa cepat server dalam mengenkripsi perintah akses ("*unlock*" atau "*buzzer*"), dan seberapa cepat ESP32 melakukan dekripsi. Untuk gambar grafik waktu proses enkripsi pada Server dapat dilihat pada Gambar 14.



Gambar 14. Grafik Waktu Enkripsi di Server

Gambar 14 memperlihatkan grafik hasil pengujian waktu proses enkripsi oleh server. Dari grafik terlihat bahwa waktu proses cukup konstan, dengan sebagian besar percobaan mencatatkan durasi 0.02 ms, dan beberapa percobaan sedikit lebih tinggi hingga 0.04 ms. Rata-rata waktu enkripsi pada server berada di angka 0.0225 ms. Selanjutnya untuk gambar grafik waktu proses dekripsi pada ESP32 dapat dilihat pada Gambar 15.



Gambar 15. Grafik Waktu Dekripsi di ESP32

Selanjutnya Gambar 15 menunjukkan durasi waktu dekripsi perintah oleh ESP32. Proses dekripsi ini memiliki waktu yang sedikit lebih tinggi dibanding proses enkripsi. Nilainya berkisar antara 0.16 ms hingga 0.18 ms, dengan rata-rata 0.166 ms.

Pembahasan

Sistem *Smart Door* berbasis IoT pada penelitian ini menggunakan algoritma *One-Time Pad* (OTP) XOR untuk mengamankan tiga jalur komunikasi: pengiriman UUID dari server ke website pengguna, pengiriman hasil pemindaian *QR Code* dari ESP32-CAM ke server, serta pengiriman instruksi dari server ke ESP32. Dari sisi keamanan, pengujian sniffing menunjukkan bahwa data yang disadap selalu berupa *ciphertext* acak, meskipun *plaintex*t yang dikirim berulang. Hal ini menunjukan bahwa OTP XOR mampu menyamarkan data dengan baik melalui penggunaan kunci acak satu kali pakai. Pengujian lebih lanjut menggunakan Cipher Identifier juga memperlihatkan seluruh *ciphertext* hanya dikenali sebagai format umum seperti Hexadecimal atau ASCII, tanpa satu pun teridentifikasi sebagai enkripsi OTP. Selain itu, uji brute force XOR terhadap 20 *ciphertext* juga gagal sepenuhnya, dengan tingkat keberhasilan dekripsi 0%. Temuan ini menegaskan ketahanan OTP XOR terhadap identifikasi pola dan serangan brute force.

Dari sisi kinerja, waktu enkripsi dan dekripsi rata-rata pada seluruh jalur komunikasi berada dalam skala milidetik: 0.011 ms (server) dan 0.010 ms (web pengguna) pada jalur pertama, 0.345 ms (ESP32-CAM) dan 0.136 ms (server) pada jalur kedua, serta 0.0225 ms (server) dan 0.166 ms (ESP32) pada jalur ketiga. Hasil ini menunjukkan bahwa OTP XOR sangat ringan dijalankan pada perangkat dengan sumber daya terbatas, tetap responsif meskipun melibatkan proses enkripsi dan dekripsi pada setiap komunikasi. Secara keseluruhan, OTP XOR terbukti mampu menjaga kerahasiaan data sekaligus mempertahankan efisiensi sistem. Dengan ketahanan 100% terhadap identifikasi algoritma maupun brute force, serta durasi proses yang rendah, algoritma ini layak digunakan dalam implementasi IoT berbasis perangkat embedded seperti ESP32.

SIMPULAN

Hasil penelitian membuktikan bahwa algoritma *One-Time Pad* (OTP) XOR dapat diterapkan secara efektif pada sistem *Smart Door* berbasis IoT.Berdasarkan pengujian menggunakan *tools* dCode.fr, seluruh *ciphertext* yang dihasilkan tidak dapat diidentifikasi sebagai hasil enkripsi OTP, dan upaya dekripsi melalui serangan *brute force* XOR juga gagal sepenuhnya. Hal ini menegaskan

bahwa algoritma ini memiliki tingkat ketahanan yang sangat tinggi terhadap kriptanalisis dasar. Selain itu, dari perspektif performa, algoritma ini terbukti sangat efisien, dengan rata-rata waktu enkripsi di ESP32-CAM hanya 0.34 ms dan waktu dekripsi di ESP32 hanya 0.17 ms. Performa ini menunjukkan bahwa OTP XOR dapat diimplementasikan dengan baik pada perangkat IoT yang memiliki keterbatasan sumber daya tanpa mengorbankan kecepatan atau responsivitas sistem. Sebagai implikasi dari temuan ini, disarankan untuk mengembangkan sistem manajemen kunci yang lebih canggih, seperti *pre-shared key* (PSK), rotasi kunci, atau *Key Derivation Function* (KDF), guna meningkatkan keamanan distribusi kunci. Peningkatan lebih lanjut dapat dicapai dengan memisahkan jalur pengiriman kunci menggunakan teknik seperti *Out-of-Band Key Exchange* atau *asymmetric key wrapping* dengan algoritma RSA, memastikan kunci tetap aman bahkan jika terjadi penyadapan.

DAFTAR PUSTAKA

[1] A. M. N. Hidayat and M. Na'im Al Jum'ah, "BIMTEK FTI: Digital Village Governance," *MEKONGGA J. Pengabdi. Masy.*, vol. 1, no. 1, pp. 15–20, 2024, https://doi.org/10.69616/mekongga.v1i1.173.

[2] T. Aditya and O. A. Dhewa, "Design and Implementation of Update Script in the IoT-Based Smart Indoor Farming System Module at PT Inastek Using Over-the-Air Programming," *Media Comput. Sci.*, vol. 1, no. 2, pp. 129–138, 2024, https://doi.org/10.69616/mcs.v1i2.201.

[3] A. Selay *et al.*, "Karimah Tauhid, Volume 1 Nomor 6 (2022), e-ISSN 2963-590X," *Karimah Tauhid*, vol. 1, no. 2963–590X, pp. 861–862, 2022.

[4] M. A. Juniawan and A. H. Rismayana, "PROTOTIPE *SMART DOOR* LOCK BERBASIS INTERNET OF THINGS (STUDI KASUS LAB KOMPUTER POLITEKNIK TEDC BANDUNG)," vol. 8, no. 5, pp. 10856–10861, 2024, https://doi.org/10.36040/jati.v8i5.11148.

[5] M. Adedoyin and F. Olukoya, "Development of a Smart Lock System using *QR Code* Technology," vol. 7, no. 2, pp. 484–492, 2024, https://doi.org/10.53982/ajerd.2024.0702.46-j.

[6] H. Fereidouni, O. Fadeitcheva, and M. Zalai, "IoT and Man-in-the-Middle Attacks," 2023.

[7] J. Manullang, J. Tamba, and F. P. Sitohang, "Cryptography With *One-Time Pad* (OTP) Algorithm Xor Based," vol. 3, no. 02, pp. 54–60, 2024, https://doi.org/10.58471/ju-ti.v3i02.664.

[8] W. Adhiwibowo, A. M. Hirzan, and M. S. Suprayogi, "Peningkatan Keamanan Data End-To-End *Smart Door* Menggunakan Advanced *Encrypt*ion Standard," *J. ELTIKOM*, vol. 6, no. 2, pp. 186–194, 2022, doi: 10.31961/eltikom.v6i2.574, https://doi.org/10.31961/eltikom.v6i2.574.

[9] A. Arianto, F. D. Hudaibah, N. Nurhalifah, M. Qippiah, and S. Bantun, "Learning Innovations in Coastal Areas Through Augmented Reality and Gamification," *J. Media Inf. Teknol.*, vol. 1, no. 2, pp. 95–102, 2024, https://doi.org/10.69616/mit.v1i2.193.

[10] H. A. Wahid, J. Maulindar, and A. I. Pradana, "Rancang Bangun Sistem Penyiraman Tanaman Otomatis Aglonema Berbasis IoT Menggunakan Blynk dan NodeMCU 32," *Innov. J. Soc. Sci. Res.*, vol. 3, no. 2, pp. 6265–6276, 2023.

[11] D. Setiawan, H. Jaya, S. Nurarif, T. Syahputra, and M. Syahril, "Implementasi Esp32-Cam Dan Blynk Pada Wifi Door Lock System Menggunakanteknik Duplex," *J. Sci. Soc. Res.*, vol. 5, no. 1, p. 159, 2022, https://doi.org/10.54314/jssr.v5i1.807.

[12] Z. Avista and O. Fahlovi, "Rancang Bangun *Smart Door* Access Berbasis Fingerprint untuk Keamanan Ruang Laboratorium," *Venus J. Publ. Rumpun Ilmu Tek.*, vol. 2, no. 1, pp. 1–13, 2024, https://doi.org/10.61132/venus.v2i1.73.

[13] M. `Toby S. Pratika, I. N. Piarsa, and A. A. K. A. C. Wiranatha, "Rancang Bangun Wireless Relay dengan Monitoring Daya Listrik Berbasis Internet of Things," *JITTER J. Ilm. Teknol. dan Komput.*, vol. 2, no. 3, p. 515, 2021, https://doi.org/10.24843/JTRTI.2021.v02.i03.p10.

[14] M. Raudiah and E. Elfizon, "Perancangan Keamanan Brangkas Berbasis Arduino dan Android," *JTEIN J. Tek. Elektro Indones.*, vol. 1, no. 2, pp. 246–250, 2020, https://doi.org/10.24036/jtein.v1i2.80.

[15] A. Fauzi and H. Sembiring, "Internet-Based *Smart Door* Design of Things (IOT) with Visitor Access Controller Indoor," vol. 4, no. 1, pp. 1–8, 2024, https://doi.org/10.59934/jaiea.v4i1.566.

[16] A. Herlan, I. Fitri, and R. Nuraini, "Rancang Bangun Sistem Monitoring Data Sebaran Covid-19 Secara Real-Time menggunakan Arduino Berbasis Internet of Things (IoT)," *J. JTIK (Jurnal Teknol. Inf. dan Komunikasi)*, vol. 5, no. 2, p. 206, 2021, https://doi.org/10.35870/jtik.v5i2.212.

[17] M. E. Hashimyar, M. Aiash, A. Khoshkholghi, and G. Nalli, "Signature-Based Security Analysis and Detection of IoT Threats in Advanced Message Queuing Protocol," *Network*, vol. 5, no. 1, 2025, https://doi.org/10.3390/network5010005.

[18] D. B. KalaiSelvi and A. K, "Network Traffic Analysis Using Wireshark," *Int. J. Res. Publ. Rev.*, vol. 4, no. 12, pp. 1960–1965, 2023, https://doi.org/10.55248/gengpi.4.1223.123506.

[19] M. R. Fahlevi, D. Ridha, D. Putri, and R. Doni, "Teknik Keamanan File Teks Menggunakan Kriptografi Dengan Algoritma One Time Pad Cipher," *J. Sains Komput. Inform. (J-SAKTI*, vol. 4, no. 2, pp. 588–597, 2020.

[20] A. A. Permana, R. Taufiq, and R. Destriana, "Implementasi Aplikasi Pengamanan Pesan Gambar Menggunakan Algoritma One Time Pad," *Proceeding SENDIU 2021*, pp. 978–979, 2021.

[21] I. Meyasha, S. Widyastuti, and R. Maulana, "Penerapan Algoritma Vernam Cipher Dan Base64 Untuk Keamanan Data Pernikahan Pada Kua Kecamatan Harjamukti," vol. 18, no. 2, pp. 12–22, 2025.