Available *online* at: https://bit.ly/SISFOTEKJAR

Jurnal Sistem Informasi dan Teknologi Jaringan

(SISFOTEKJAR)

ISSN (Print) 2808-7917 | ISSN (Online) 2807-9256

Kunci Simetris pada Perangkat IoT: Kajian Literatur

Purwanto Simamora[1], Verawijaya[2], Sutrisno Arianto Pasaribu[3]

[1,2,3] Teknologi Informasi, Universitas Mahkota Tricom Unggul, Grand Jati Junction, Jl. Perintis Kemerdekaan No.3A, Medan, 20233, Indonesia
[1]purwantosimamora@gmail.com, [2]verawijaya@gmail.com, [3]sutrisno@gmail.com

ARTICLE INFORMATION

Received: August 9, 2024
Revised: August 23, 2024
Available online: September 15, 2024

KEYWORDS

Perangkat IOT, Algoritma Kunci Simetris, AES, RC6, Kriptografi

CORRESPONDENCE

Phone: +62 82164664433
E-mail: verawijaya83@gmail.com

ABSTRACT

Perangkat IoT memerlukan enkripsi yang efisien dan aman untuk melindungi data sensitif saat proses transmisi. Dari beberapa jenis kriptografi kunci simetris yang ada, AES menjadi pilihan terbaik untuk sebagian besar aplikasi IoT karena memiliki sistem keamanan yang sangat tinggi, efisien, dan mendapat dukungan luas. Selanjutnya sebagai alternatif, pilihan jatuh pada algoritma RC6 dapat digunakan pada situasi tertentu yang memerlukan performa tinggi dengan dukungan layanan terbatas. Algoritma DES dan RC4 sudah usang dan tidak cocok digunakan pada aplikasi IoT modern karena sistem keamanan yang lemah. Memilih algoritma kunci simetris yang tepat sangat penting untuk memastikan sistem keamanan, efisiensi dan keandalan dalam perangkat IoT. AES dengan berbagai kelebihannya menjadi pilihan utama dalam kriptografi kunci simetris.

1. PENDAHULUAN

Kriptografi merupakan konsep awal dalam keamanan komunikasi *internet of things* (IoT). Tujuan utama kriptografi adalah mengubah data yang dapat dibaca (plaintext) menjadi bentuk yang tidak dapat dipahami (ciphertext) menggunakan berbagai algoritma dan kunci. Proses ini memastikan pelaku kejahatan tidak dapat menguraikan pesan atau data tanpa adanya kunci dekripsi yang sesuai. Algoritma enkripsi yang umum digunakan dalam komunikasi IoT adalah AES, RC, dan DES. Untuk memastikan bahwa data yang diterima tidak rusak selama proses, kriptografi menggunakan konsep integritas pesan. Fungsi hash (MD5 atau SHA), menghasilkan has berukuran tetap yang unik untuk data asli. Jika hash data diterima sama dengan hash alis, maka proses integritas sukses [1].

Kriptografi juga memiliki metode otentikasi melalui tanda tangan digital, perangkat IoT dapat memvalidasi identitas pengirim dan melakukan verifikasi integritas data yang diterima. Hal ini mencegah pihak yang tidak bertanggung jawab untuk memanipulasi atau memasukkan informasi palsu ke dalam jaringan IoT. Memahami prinsip kriptografi sangat penting dalam memilih algoritma enkripsi yang tepat, menerapkan sistem manajemen kunci yang aman, dan memastikan komunikasi IoT secara menyeluruh [1].

Pada banyak kasus, perlindungan keamanan berbasis perangkat lunak tidak memberikan sistem keamanan yang sesuai. Peretas selalu berinovasi untuk menemukan cara terbaik untuk menyerang sistem operasi dan mencuri data penting. Teknologi enkripsi dan dekripsi data menjadi pilihan terbaik untuk menjaga keamanan perangkat IoT, data rahasia, dan informasi data pribadi sehingga ada jaminan keamanan secara efektif. Algoritma kriptografi sederhana tidak bekerja dengan baik pada perangkat IoT karena adanya keterbatasan daya, bandwidth, waktu eksekusi, dan kemampuan komputasi. Oleh karena itu, algoritma kriptografi ringan (*lightweight cryptographic algorithm*) penting untuk sumber daya terbatas, seperti perangkat yang menggunakan baterai [2]
Secara garis besar, kriptografi dapat dibedakan menjadi 2 bagian, yaitu kriptografi kunci simetris (kunci privat) dan kriptografi kunci asimetris (kunci publik). Teknik kriptografi digunakan untuk mengenkripsi atau mendekripsi informasi sesuai dengan

standar keamanan. Hal ini berkaitan dengan studi matematis tentang keamanan informasi seperti kerahasiaan, otentikasi dan integritas data dengan menggunakan kriptografi kunci simetris dan kunci tidak simetris. Kriptografi kunci simetris menggunakan kunci rahasia untuk melakukan enkripsi dan dekripsi pesan. Sebaliknya, kriptografi asimetris menggunakan kunci rahasia yang berbeda untuk enkripsi dan dekripsi dan umumnya cukup sulit untuk menebak kunci rahasia dan kunci publik tersebut [2].

Algoritma kunci simetris seperti DES, RC2, RC4, Blowfish, RC5, RC6 atau AES merupakan bentuk enkripsi tertua dan termudah dengan hanya menyediakan satu kunci yang sama untuk menyandikan (enkripsi) dan menguraikan (dekripsi) informasi. Hal ini sering menjadi alat bagi penyerang untuk menguping saluran pertukaran kunci dan menggunakannya untuk mendekripsi data. Saluran aman diperlukan antara pengirim dan penerima untuk mengubah kunci rahasia. Selanjutnya, algoritma kunci asimetris seperti DSA, ElGamal, Diffie-Hellman, RSA, ECC, PGP, TLS, S/MIME dan NTRU telah menggunakan sepasang kunci yang berbeda yaitu untuk enkripsi (kunci publik) dan dekripsi (kunci privat) untuk mengkodekan teks. Entitas apapun yang mempunyai kunci publik dapat dengan mudah menggunakannya untuk mengirim pesan dengan tetap merahasiakan kunci pribadi untuk mendekripsi pesan. Algoritma kunci asimetris memegang peranan penting dalam berbagai aplikasi keamanan, terutama dalam konteks internet dan komunikasi elektronik, karena proses enkripsi data secara aman, menggunakan tanda tangan digital dan melakukan pertukaran kunci dengan aman [3].

Penelitian ini fokus pada kriptografi kunci simetris yang diterapkan secara luas pada perangkat IoT untuk mengantisipasi segala bentuk serangan yang mungkin terjadi. Selanjutnya, penelitian ini akan mengidentifikasi algoritma kriptografi simetris yang paling sesuai untuk keterbatasan sumber daya, kecepatan enkripsi dan dekripsi data, mengukur kesulitan dalam mengimplementasikan algoritma pada perangkat IoT dan menilai sejauh mana algoritma dapat dengan cepat diintegrasikan pada perangkat atau protokol yang ada. Penelitian ini juga akan mengevaluasi kemampuan algoritma pada jaringan IoT dalam skala besar termasuk kemampuan algoritma untuk menyesuaikan diri dengan kebutuhan dan lingkungan IoT yang berbeda-beda.

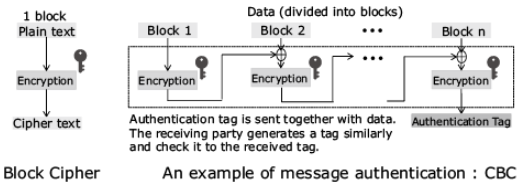
2. METODE PENELITIAN

Kriptografi adalah seni menulis rahasia untuk melindungi informasi dengan mengubahnya menjadi format yang tidak dapat dibaca dimana pesan dapat disembunyikan dari pembaca dan hanya penerima yang dituju yang dapat mengubahnya menjadi pesan asli. Tujuan utamanya adalah menjaga keamanan data dari akses yang tidak sah. Data dapat dibaca dan dipahami tanpa ada ukuran khusus yang disebut *plaintext.* Teknik untuk menyamarkan *plaintext* untuk menyembunyikan pesan disebut enkripsi. Mengenkripsi teks yang tidak dapat dibaca disebut *ciphertext.* Proses mengembalikan *ciphertext* ke dalam *plaintext* disebut dekripsi. Sistem yang menyediakan enkripsi dan dekripsi disebut *cryptosystem*. Kriptografi dikenal sebagai pelindung data sensitif dari pengguna yang tidak sah melalui enkripsi dan mengizinkan pengguna yang berwenang untuk mendekripsi data. Kriptografi terdiri dari dua jenis algoritma yaitu kunci simetris (algoritma kunci privat) dan algoritma asimetris (algoritma kunci publik) [4][6].

1. Kriptografi Kunci Simetris

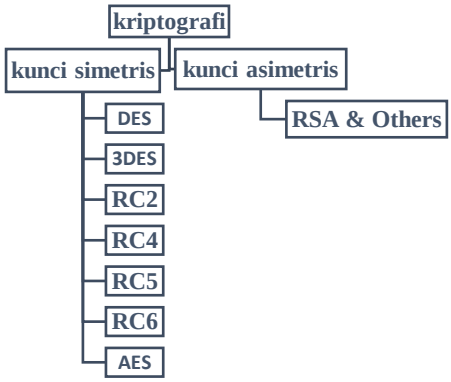
Kriptografi kunci simetris terdir dari dua fungsi utama yaitu blok atau aliran sandi (*cipher or stream cypher*) atau disebut juga kriptografi primitif dan metode untuk menerapkan fungsi utama ke paket yang disebut dengan mode operasi blok sandi untuk enkripsi atau otentikasi. Gambar 1 menunjukkan contoh mode operasi blok sandi yang digunakan untuk otentikasi (disebut CBC-MAC: *cipher block chaining message authentication*). Untuk mendapatkan kriptografi ringan (lightweight cryptography), diperlukan peningkatan efisiensi mode operasi blok sandi setara dengan kriptografi primitif [5][6][7][8].

Kriptografi simetris merupakan metode enkripsi tercepat dan paling sederhana karena kunci enkripsi dan dekripsi sama. Hal ini cocok untuk perangkat IoT yang memiliki keterbatasan sumber daya dan membutuhkan komunikasi yang cepat dan terus menerus. Akan tetapi, kripografi simetris memiliki kekurangan, diantaranya resiko kesepakatan kunci, manajemen dan distribusi kunci dan masalah otentikasi dan penyatuan [5][6][22].



Gambar 1. Contoh aplikasi kriptografi ringan

Banyak algoritma enkripsi digunakan dalam keamanan informasi. Algoritma kunci simetris terbagi dua, yaitu kunci simetris (pribadi) dan kunci asimetris (publik). Dalam enkripsi kunci simetris, digunakan kunci rahasia bersama untuk mengenkripsi dan mendekripsi data. Kunci harus didistribusikan sebelum terjadi transaksi antar entitas. Kunci sangat berperan penting karena bila kunci lemah maka penyerang dapat dengan mudah mendekripsi data. Kekuatan kunci bergantung pada ukuran dari kunci yang digunakan. Ada dua jenis algoritma kunci simetris, yaitu stream cipher dan block cipher yang masing-masing terdiri dari bit dan blok. Klasifikasi teknik enkripsi yang paling umum digunakan ditunjukkan pada gambar 1 [5][6][7][8].



Gambar 2. Klasifikasi Kriptografi

2. *DES*

Algoritma *data encryption standard* (DES) merupakan jenis algoritma kriptografi simetris yang digunakan untuk mengenkripsi data. Algoritma ini mempunyai ukuran blok yang kecil dan ukuran kunci yang pendek, sangat jarang digunakan karena tidak aman dalam hal ukuran kunci. DES memegang peranan penting dalam sejarah kriptografi dan merupakan dasar pengembangan algoritma kriptografi yang lebih kuat dan aman. Jenis serangan paling sering terjadi pada algoritma DES adalah *brute force attack* dengan mencoba semua kunci yang mungkin dan melihat kunci mana yang dapat mendekripsi pesan yang benar [4] [5][7].

Algoritma DES melibatkan beberapa langkah utama dalam proses enkripsi dan dekripsi. Berikut langkah-langkah umum untuk melakukan enkripsi menggunakan algoritma DES.

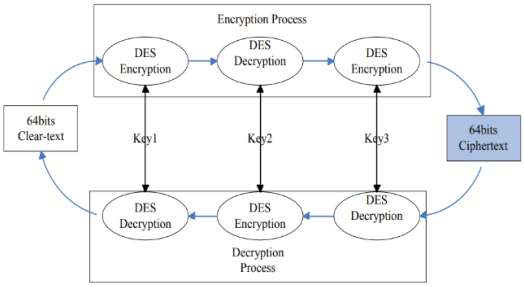
1. Bangkitkan kunci DES: Kunci DES yang digunakan akan dihasilkan dari sebuah password fungsi PBKDF2.
2. Enkripsi pesan: Gunakan kunci DES yang telah dihasilkan untuk mengenkripsi pesan.
3. Dekripsi pesan: Gunakan kembali kunci DES untuk mendekripsi pesan yang telah dienkripsi.

Berikut adalah contoh soal algoritma DES. Anda memiliki pesan plaintext yang ingin Anda enkripsi menggunakan algoritma DES. Pesan tersebut adalah "Ini adalah pesan rahasia!" dalam format ASCII dan menggunakan kata kunci PBKDF2.
Solusi untuk soal diatas diawali dengan pesan “Ini adalah pesan rahasia!” yang disebut dengan *plaintext*.

Bangkitkan kunci DES dari password dengan fungsi PBKDF2 menjadi ‘mysecretpassword’
Enkripsi pesan menjadi b'F-~\x0c\xe4\xcf\x9e\x95\xc1\x8b\t\xf3\xf7\xda&\x1d\xb60\xaf~\xc4\xe6\x0fp\xc1\x08\xcf\x12=\xa4\xff\xea\xa3/H"@8\xe7\x95'
Dekripsi pesan menjadi: Ini adalah pesan rahasia!

3. *DES*

Algoritma 3DES (*triple DES*) pertama kali diperkenalkan tahun 1998.Algoritma 3DES pada dasarnya menggunakan algoritma DES sebanyak tiga kali pada *plaintext*. Kunci 56-bit DES asli sudah cukup memberikan keamanan tetapi dengan adanya tambahan 4 kekuatan komputasi menyebabkan peningkatan serangan *brute force*. Untuk mengantisipasi hal tersebut terulang kembali, dikembangkanlah 3DES menggunakan kunci 168-bit dan beroperasi pada ukuran blok 64-bit. Meskipun algoritma ini lebih aman dari DES, 3DES dianggap sebagai blok sandi yang paling lambat karena terjadinya komputasi secara berlebihan. Tiap kumpulan data dienkripsi terlebih dahulu menggunakan DES, didekripsi dan dienkripsi lagi untuk keamanan maksimal. Panjang kunci 112 atau 168-bit, dan panjang blok mencapai 64-bit. Munculnya algoritma ini berawal dari versi pertama DES telah dirusak secara brutal dan metode kriptanalitik lainnya karena sandinya yang lemah. Algoritma 3DES ditawarkan untuk memperluas ukuran kunci DES sehingga lebih rentan terhadap serangan sejenis. Tujuan utama 3DES adalah menghindari pembuatan algoritma blok sandi baru. Diagram enkripsi dan dekripsi 3DES ditunjukkan pada gambar berikut [20][21][23].

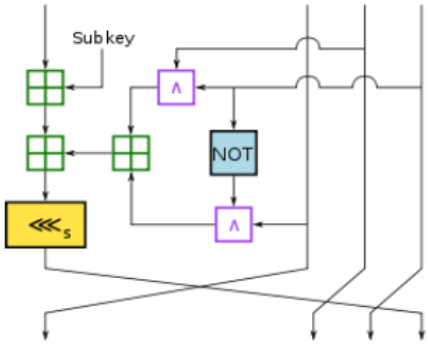


Gambar 3. Representasi proses enkripsi dan dekripsi algoritma 3DES

4. *RC2*

Algoritma RC2 (Ron’s Code/Rivest’s Cipher) merupakan salah satu algoritma yang dapat digunakan untuk melakukan enkripsi data sehingga data asli hanya dapat dibaca oleh seseorang yang memiliki kunci enkripsi tersebut. Berdasarkan panjang variabel kunci masukan, algoritma perluasan digunakan untuk mengubahnya menjadi kunci tetap 64-bit. Hal ini dilakukan dengan mengikuri rangkaian operasi yang melibatkan 6 putaran MIXING, satu putaran MASHING, 6 putaran MIXING, putaran MASHING lainnya diikuti oleh 5 putaran MASHING lainnya. Satu MIXING terdiri dari 4 transformasi MIX-UP [6][21].

Algoritma RC2 dirancang untuk RSA *data security*. Algoritma RC2 merupakan *block cipher* 64-bit dengan ukuran kunci yang bervariasi sampai dengan 128-bit. Ke-18 putarannya disusun sebagai jaringan Feistel dengan sumber daya besar, dengan 16 putaran sejenis (MIXING) dan diselingi oleh dua putaran lainnya (MASHING), Satu putaran MIXING terdiri dari empat aplikasi transformasi MIX, seperti ditunjukkan gambar 2. RC2 rentan terhadap serangan pada kunci yang saling berkaitan menggunakan 2^{34} pilihan teks [6][8][9][21].



Gambar 3. Transformasi MIX RC2, empat diantaranya terdiri dari putaran MIXING

Suatu array yang berisi empat kata 16-bit R[0], R[1], R[2] dan R[3] digunakan untuk menampung *plaintext* awal, pertengahan dan sandi akhir teks. Satu putaran MIXING didefinisikan sebagai i = 0, 1, 2, dan 3 [6][8][9].

$$R[i] = R[i] + K[j] + (R[i-1] \,\&\, R[i-2]) + (\sim R[i-1] \,\&\, R[i-3]);$$

Dimana & menunjukkan logika bitwise dan $\oplus$ menunjukkan bitwise XOR dan $\sim$ menunjukkan komplemen bitwise. Semua penambahan kata + 16-bit dilakukan modulo 216.

$$j = j + 1;$$

Dalam hal ini j adalah variabel ‘global’sehingga $K[j]$ selalu menjadi kunci pertama dalam perluasan kunci yang belum digunakan dalam operasi MIXING.

$$R[i] = R[i] \lll s[i];$$

Dimana $s[0] = 1, s[1] = 2, s[3] = 5$. Dan $R[i] = R[i] \lll s[i]$ menyatakan bahwa $R[i]$ diputar ke kiri oleh $s[i]$ bit.

Satu putaran MASHING didefinisikan sebagai berikut, untuk tiap i = 0, 1, 2, dan 3:
R[i] = R[i] + K[R[i − 1] & 63];

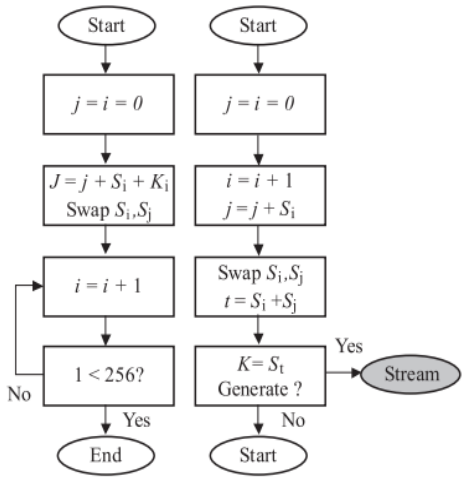
Seluruh operasi enkripsi menggunakan RC2 dapat dijelaskan sebagai berikut:
Inisialisasi kata-kata R[0], R[1], R[2], R[3] untuk mengisi blok *plaintext* 64-bit.
Perluas kunci, sehingga kata-kata K[0], ..., K[63] terdefinisi.
1. Inisialisasi j ke nol.
2. Lakukan lima putaran MIXING
3. Lakukan satu putaran MASHING
4. Lakukan enam putaran MIXING
5. Lakukan satu putaran MASHING
6. Lakukan lima putaran MIXING
7. Teks sandi adalah R[0], ..., R[3].

Meskipun tidak ada disain lemah yang terdefinisi di RC2 yang menyebabkan serangkan kriptanalis praktis, hal ini tetap dianggap sebagai sandi yang lambat.

5. *RC4*

RC4 telah digunakan dalam banyak aplikasi dan yang paling terkenal adalah *Wi-Fi encryption Standard Wireless Equivalent Privacy* (WEP) dan protokol *Transport Layer Security* (TLS) pada pembuatan konkeis HTTPS. Tetapi, untuk sebagian besar aplikasi, termasuk WEP dan TLS, RC4 tidak cukup aman untuk digunakan. Untuk lebih jelasnya, berikut cara kerja RC4 [10][21]. RC4 adalah sebuah aliran sandi yang mengenkripsi bukan hanya satu byte *plaintext* tetapi juga didisain untuk mengenkripsi satu bit atau bahkan unit yang lebih besar dari satu byte pada satu waktu. Di dalam struktur ini kuncinya dimasukkan ke generator bit acak untuk menghasilkan aliran 8-bit acak, aliran acak tidak dapat diprediksi tanpa mengetehaui kunci masukannya. Keluaran dari pembangkitan disebut aliran kunci dan menggabungkan satu byte dengan aliran *plaintext* menggunakan operasi bitwise exclusive-OR (XOR) [11][21].

RC4 sederhana dan mudah dijelaskan. Algoritma ini didasarkan pada permutasi acak. Kunci dengan panjang variabel K [ ] dari 1 sampai 256 byte (8 sampai 2048 bit) digunakan untuk menginisialisasi vektor keadaan S[ ], dengan elemen S [0] sammpai S[255]. Setiap saat S [ ] berisi permutasi semua bilangan 8-bit dari 0 sampai 255. Pada enkripsi dan dekripsi, suatu byte K dibangkitkan dari S [ ] dengan memilih satu dari 255 masukan dalam mode sistematis. Saat tiap nilai K dihasilkan, masukan dalam S [ ] kembali diperbolehkan. Gambar 3 menunjukkan diagram blok RC4 dua fase. Untuk dekripsi, nikai K XOR dengan byte sandi berikutnya, menjelaskan bahwa jika kunci berbeda untuk enkripsi dan dekripsi maka *plain text* tidak akan pernah kembali lagi, meskipun sudah menggunakan kunci yang berbeda untuk enkripsi yang dihasilkan sandi tidaklah sama [11].

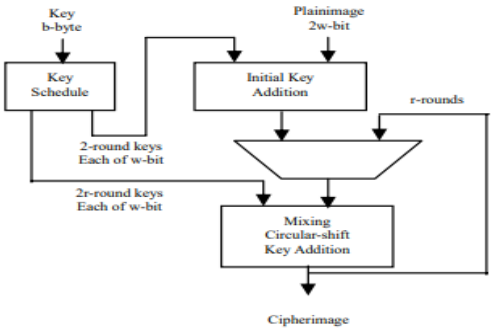


Gambar 4. Blok Diagram RC4

6. *RC5*

Rivest mengusulkan blok sandi RC5, yang yaitu sandi tranformasi nilai. Algoritma enkripsi RC5 merupakan blok simetris yang cepat diterapkan pada perangkat keras dan lunak. Fitur baru RC5 merupakan rotasi yang bergantung pada banyaknya data. RC5 memiliki ukuran kata, jumlah putaran dan panjang kunci yang bervariasi. Algoritma enkripsi dan dekripsi sangat sederhana [12].

Secara umum, RC5 aman dan kuat dari sudut pandang kriptografi dan cocok digunakan pada banyak aplikasi. Algoritma enkripsi RC5 merupakan sandi blok yang mengubah blok data 16, 32, dan 64 bit teks biasa (plaintext) menjadi sandi blok dengan panjang yang sama. RC5 menggunakan suatu kunci dengan panjang b yang dapat dipilih (0, 1, 2, ... 255) byte. Algoritma mengatur kumpulan iterasi yang disebut putaran r dalam rentang (0, 1, 2, ...255) seperti ditunjukkan pada gambar 4 [12][14].



Gambar 5. Algoritma Enkripsi

Algoritma blok sandi RC5 merupakan algoritma enkripsi simetris yang diwakili dengan notasi RC5 w/r/b, dimana w/r/b merupakan parameter yang dapat dikonfigurasi ulang. Parameternya diuraikan sebagai berikut [13] [14].
1. w adalah ukuran kata dalam bit dan nilai standarnya adalah 32 bit dari nilai yang dibolehkan yaitu 16, 32 dan 64 bit. Algoritma RC5 mengenkripsi dua blok kata yaitu *plainimange* dan *cipherimage* dengan panjang tiap bit 2w.
2. r menandakan jumlah putaran. Jumlah putaran berkisar 0 sampai 255.

3. b menandakan jumlah byte dalam kunci rahasia K. Ukuran kunci rahasia berada dalam rentang 0 – 2.400 bit.

Algoritma RC5 terdiri atas tiga komponen: algoritma perluasan kunci, algoritma enkripsi, dan algoritma dekripsi. Algoritma RC5 menggunakan tiga operasi primitif dan kebalikannya, yaitu [13]:

1. Penjumlahan/pengurangan kata modulo 2w, dimana w adalah ukuran kata.
2. Kata bit-wise eksklusif OR dinyatakan dengan $\oplus$
3. Rotasi: putar kata x ke kiri sebanyak y bit dan dilambangkan dengan $x \lll y$. Kebalikan operasinya adalah putar kata x ke kanan sebanyak y bit, dilambangkan $x \ggg y$. Catatan hanya log2 (w) bit orde rendah y yang dapat mempengaruhi putaran ini.

7. RC6

RC6 berasal dari RC5. Ada 2 fitur baru dalam RC6 yaitu penyertaan perkalian bilangan bulat dan penggunaan empat w-bit register kerja sebagai pengganti dua w-bit pada RC6. Algoritma ini sangat mirip dengan RC5 dalam strukturnya, dengan ketergantungan data rotasi, penambahan modulo 2w dan operasi XOR. Kenyataannya, RC6 dapat dipandang sebagai gabungan dua proses enkripsi secara paralel. Bedanya, RC6 menggunakan operasi perkalian tambahan yang tidak dimiliki oleh RC5 dalam rotasi tiap bit kata dan bukan pada bit yang tidak perlu. Perkalian bilangan bulat digunakan untuk menambah meningkatkan difusi per putaran sehingga putaran lebih sedikit diperlukan dan kecepatan cipher dapat ditingkatkan. RC6 juga merupakan kelompok enkripsi yang sepenuhnya berdasarkan algoritma. Versi RC6 sering disebut RC6-w/r/b dengan ukuran kata adalah w bit, enkripsi terdir dari sejumlah putaran bukan negatif r dan b menunjukkan panjang kunci enkripsi dalam byte [16][23].

Algoritma ini sangat bergantung pada kerja empat register, masing-masing berukuran 32 bit. RC6 menangani 128 bit masukan atau keluaran. Kelompok paramaternya meliputi ukuran kata dalam bit (w), jumlah putaran bukan negatif (r), dan panjang kunci enkripsi/dekripsi dalam byte (b). RC6 memiliki enam operasi primitif (+, -, <<<, >>>, *, $\oplus$). Penggunaan perkalian meningkatkan pencapaian difusi per putaran, memungkinkan keamanan yang lebih besar, dan menambah keluaran (throughput). RC6 menggunakan tabel kunci yang sudah dikembangkan, S[0, ..., t-1], terdiri dari kunci t = 2r + 4 w bit kata [15].

8. AES

Advanced Encryption Standard (AES) muncul sebagai algoritma enkripsi simetris yang disukai dari sisi keamanan dan efisiensinya yang kuat. Sebaliknya, DES semakin ditinggalkan karena rentan terhadap serangan *brute-force*. Sedangkan RC4 menghadapi masalah keamanan yang berasal dari kerentanan dalam algoritma penjadwalan utamanya. AES merupakan algoritma blok sandi simetris yang dikembangkan dari metode Rijndael dan digunakan untuk mengenkripsi dan mendekripsi informasi menggunakan kunci yang sama. AES hanya dapat mengenkripsi dan mendekripsi blok sandi 128-bit dengan 128, 192, atau 256 bit kunci [17[21].

Saat ini, AES paling banyak digunakan dan enam kali lebih cepat dari 3DES. AES paling dapat diterima sebagai standar enkripsi di dunia keamanan siber. AES digunakan pada beberapa aplikasi seperti *Signal* dan *Whatsapp*, platform komputer seperti *VeraCrypt* dan teknologi lainnya. Ukuran kunci juga cocok untuk mengamankan data rahasia pada tingkat yang memuaskan.

Meskipun umumnya menggunakan format 128-bit, AES juga menggunakan kunci bit 192- dan 256-bit untuk mengenkripsi data dalam jumlah besar. AES termasuk tahan pada berbagai serangan kecuali *brute-force*, yang mampu melakukan dekripsi pesan dengan menggunakan semua kemungkinan kombinasi enkripsi 128-bit, 192-bit atau 256-bit. Semakin tinggi ukuran kuncinya, semakin kuat enkripsinya. AES mendefinisikan empat transformasi untuk mengubah *plaintext* menjadi *ciphertext*. Langkah pertama melibatkan pengaturan data menjadi sebuah array atau matriks. Langkah kedua menggeser baris data, langkah ketiga mencampur kolom dan langkah terakhir adalah melakukan operasi XOR sederhana pada tiap kolom dengan menggunakan kunci enkripsi yang berbeda[18][21].

3. HASIL DAN PEMBAHASAN

Keamanan kunci simetris pada perangkat IoT dipengaruhi oleh banyak faktor seperti jenis algoritma yang digunakan, panjang kunci, dan implementasi sistem yang baik. Algoritma kunci simetris AES sangat populer dan dianggap aman jika menggunakan panjang kunci yang cukup (misal 128bit, 192bit atau 256 bit). Algoritma ini telah melalui pengujian yang panjang untuk digunakan dalam banyak aplikasi keamanan. Panjang kunci yang lebih panjang biasanya lebih aman karena lebih sulit untuk dipecahkan dengan serangan brute force. Dalam konteks IoT, panjang kunci minimal 128bit sudah dianggap aman untuk digunakan dalam enkripsi. Salah satu tantangan terbesar dalam menggunakan kunci simetris di IoT adalah pengelolaan dan distribusi kunci. Kunci harus didistribusikan dengan aman dari akses tidak sah. Manajemen kunci yang baik sangat menentukan keberhasilan suatu sistem keamanan. Perangkat IoT umumnya memiliki sumber daya yang terbatas dalam komputasi, memori dan energi. Oleh karena itu, algoritma kunci simetris harus digunakan dengan efisien dan sesuai dengan kemampuan perangkat. AES sering menjadi pilihan karena efisiensinya dan telah dioptimalkan dalam banyak platform.

Algoritma kunci simetris DES memiliki struktur sederhana dan menjadi standar untuk mempelajari kriptografi simetris. DES dapat digunakan pada perangkat dengan sumber daya terbatas karena cukup cepat dan efisien. Kunci simetris ini hanya memiliki 56bit dan sudah tidak aman dan tidak disarankan untuk digunakan pada komputasi modern karena kerentanannya terhadap serangan kriptanalis dan brute force. Sebagai alternatif, diperkenalkan algoritma kunci simetris 3DES dengan menggunakan 3 kunci DES. Algoritma ini menambah kompleksitas dan memperlambat proses enkripsi dan dekripsi dan tidak mendukung platform modern sehingga membuatnya kurang efisien pada aplikasi keamanan modern.

Algoritma kunci simetris AES memiliki kombinasi sistem keamanan kuat karena adanya dukungan panjang kunci 128bit, 192bit dan 256bit. Algoritma AES sangat cepat dan efisien pada

berbagai platform keamanan modern termasuk pada perangkat IoT dengan sumber daya terbatas. Saat ini AES sudah digunakan secara luas dan telah teruji pada berbagai perangkat keras. AES digunakan dalam berbagai aplikasi seperti protokol TLS untuk keamanan komunikasi web, SSL untuk enkripsi data, IPsec untuk keamanan komunikasi jaringan VPN, BitLocker di Windows, FileVaultt di *macOS*, *dm-crypt* di Linux, aplikasi 7-Zip, *WinRAR*, dan *VeraCrypt*, *Google Drive*, *Dropbox*, *iCloud*, *AWS*, Microsoft Azure, dan Google Cloud Platform dan sebagainya. AES merupakan algoritma enkripsi serbaguna, efisien, dan mendukung pada banyak platform menjadikannya pilihan utama mulai dari penggunaan sehari-hari hingga aplikasi penting seperti industri, pemerintahan dan militer.

Algoritma kunci simetris RC2 cukup efisien dan cepat tetapi kurang aman dan tidak banyak digunakan pada perangkat keras modern. Algoritma kunci simetris RC4 sangat cepat dan efisien, terutama pada perangkat dengan sumber daya terbatas. Akan tetapi algoritma ini rentan terhadap serangan dan tidak disarankan untuk digunakan. Algoritma kunci simetris RC5 dapat diatur dengan panjang kunci yang bervariasi dan parameternya dapat dikonfigurasi dan relatif cepat dan efisien, hanya saja algoritma ini lebih rumit dibandingkan dengan AES sehingga tidak populer.

Algortima RC6 dianggap sebagai algoritma enkripsi yang kuat dan efisien dengan panjang kunci variabel mulai dari 128bit hingga 256 bit, sehingga memberikan fleksibilitas dalam tingkat keamanan. Algoritma ini dirancang tahan terhadap serangan kriptanalis seperti serangan diferensial dan serangan linier. RC6 sangat cepat dalam proses enkripsi dan dekripsi terutama pada perangkat keras dengan dukungan operasi paralel. RC6 sederhana dan efisien dalam berbagai aplikasi. Akan tetapi algoritma ini tidak mendapat dukungan luas seperti AES sehingga dukungan perangkat keras dan lunak lebih terbatas. RC6 sudah menggunakan transformasi data seperti operasi XOR, penambahan modulo, pergeseran rotasi dan transformasi non-linier. Kelebihan utama dari RC6 terletak pada kecepatan dan fleksibilitas meskipun kurang mendapat dukungan secara luas seperti halnya AES.

4. KESIMPULAN

Perangkat IoT memerlukan enkripsi yang efisien dan aman untuk melindungi data sensitif saat proses transmisi. Dari beberapa jenis kriptografi kunci simetris yang ada, AES menjadi pilihan terbaik untuk sebagian besar aplikasi IoT karena memiliki sistem keamanan yang sangat tinggi, efisien, dan mendapat dukungan luas. Selanjutnya sebagai alternatif, pilihan jatuh pada algoritma RC6 dapat digunakan pada situasi tertentu yang memerlukan performa tinggi dengan dukungan layanan terbatas. Algoritma DES dan RC4 sudah usang dan tidak cocok digunakan pada aplikasi IoT modern karena sistem keamanan yang lemah. Memilih algoritma kunci simetris yang tepat sangat penting untuk memastikan sistem keamanan, efisiensi dan keandalan dalam perangkat IoT. AED dengan berbagai kelebihannya menjadi pilihan utama dalam kriptografi kunci simetris.

DAFTAR PUSTAKA

[1] Kunal, G. 2023. *Securing IoT Communications: Understanding Cryptography in the Internet of Things.* https://www.softobotics.com/blogs/securing-iot-communications-understanding-cryptography-in-the-internet-of-things/

[2] Mhaibes, H.I., Abood, M.H., Farhan, A.K. (2022). *Simple Lightweight Cryptographic Algorithm to Secure Imbedded IoT Devices*. IJIM, Vol. 16 No. 20.

[3] Al-Shabi, A.A. (2019). *A Survey on Symmetric and Asymmetric Cryptography Algorithms in Information Security*. International Journal Of Scientific and Reserach Publications, Volume 9, Issue 3, ISSN 2250-3153.

[4] Singh, P., Kumar, S. (2018). *Study & Analysis of Cryptography Algorithms: RSA, AES, DES, T-DES, Blowfish*. International Juournal of Engineering & Technology, 7 (1.5) 221-225.

[5] Panda, M. (2016). *Performance Analysis of Encryption Algorithms for Security*. International Journal of Signal Processing, Communication, Power and Embedded System (SCOPES).

[6] Gunasundari, T., Elangovan, K. (2014). *A Comparative Survey on Symmetric Key Encryption Algoritms.* IJCSMA, vol.2 Issue. 2, pg. 78-83.

[7] Elminaam, D.S.A., Kader, H.M.A., Hadhoud, M.M. (2008). *Performance Evaluation of Symmetric Encryption Algoritms* IJCSNS, vol. 8 No. 12.

[8] Elgeldawi, E., Mahrous, M. Sayed, A. (2019). *A Comparative Analysis of Symmetric Algorithms in Cloud Computing: A Survey.* International Journal of Computer Applications, volume 182-No. 48.

[9] Nema, P., Rizvi, M.A. (2015). *Critical Analysis of Various Symmetric Key Cryptographic Algorithms.* IJRITCC, Volume: 3 Issue: 6.

[10] Aumasson, J.P. (2018). *Serious Cryptography. A Practical Introduction to Modern Encryption.* No Starch Press, Inc.

[11] Riad, M.A., Shehata, A.R., Hamdy, K.E. Alsouad, H.A., Ibrahim, T.R. (2009). *Evaluation of the RC$ Algorithm as A Solution for Converged Networks.* Journal of Electrical Engineering, vol. 60, No. 3, 155-160.

[12] Belhaj, A., Zaibi, G., Kachouri, A. (2011). *Implementation of RC5 dan RC6 Block Ciphers on Digital Images.* International Journal of Information Technology, volume 3 Number 4.

[13] Faragallah, O.S. (2011). *Digital Image Encryption Based on the RC5 Block Cipher Algorithm.* Sens Imaging 12: 73-94 DOI 10.1006/s11220-011-0062-5.

[14] Chavan, S.K., Bangare, M.J. (2013). *Secure Data Storage in Cloud Service using RC5 Algorithm.* IJRTE, volume-2, issue-5.

[15] Fishawy, N.E., Zaid, O.M.A. (2007). *Quality of Encryption Measumement Quality of Encryption Measumement of Bitmap Image with RC6, MRC6, and Rijndael Block Cipher Algorithms.* International Journal of Network Security, Vol. 5, No. 3, PP. 241-251.

[16] Verma, H.K., Singh, R.K. (2012). *Performance Analysis of RC6, Twofish dan Rijndael Block Cipher Algoritms.* International Journal of Computer Applications (0975-8887), volume 42-No. 16.

[17] Hung, C., Hsu, W.T. (2018). *Power Consumption and Calculation Requirement Analysis of AES for WSN IoT.* Sensors, 18, 1675; doi: 10.3390/s18061675.

[18] Al-Mashhadani, M., Shujaa, M. (2021). *IoT Security Using AES Encryption Technology based ESP32*

Platform. The International Arab of Information Technology, Vol. 19, No. 2.

[19] Kumar J., Saxena V. (2020). *Hibridization of Cryptography for Security of Cloud*. International Journal of Future Generation Communication and Networking Vol. 13, No. 4. pp. 4007 – 4014.

[20] Radhi, S.M., Ogla, R. (2023). *In-Depth Assessment of Cryptographic Algorithms Namely DES, 3DES, AES, RSA, and Blowfish*. IJCCCE, vol. 23, No. 3. https://doi.org/10.33103/uot.ijccce.23.3.11

[21] Deshpande, K., Singh, P. (2018). *Performance Evaluation of Cryptography Ciphers on IoT Devices.* ReserachGate.https://www.researchgate.net/publication/329464607

[22] Mehmood, M.S., Shahid, M.R., Jamil, A. (2019). *A Comprehensive Literature Review of Data Encryption Techniques in Cloud Computing and IoT Environment.* IEEE, 978-1-7281-2334-9/19.

[23] Tiwari, C.S., Jha, V.K. (2023). *Enhancing the Cloud Security through RC6 and 3DES Algorithms while Achieving Low-Cost Encryption.* I.J. Wireless and Microwave Technologies, 5, 48-49. DOI: 10.5815/ijwmt.2023.05.05