IMPLEMENTASI RCCM (RIVEST CIPHER 4 BASED ON CHAOTIC MAPPING) PADA PESAN TEKS MENGGUNAKAN GUI PYTHON

RCCM (RIVEST CIPHER 4 BASED ON CHAOTIC MAPPING) IMPLEMENTATION IN TEXT MESSAGE USING PYTHON GUI

Adidtya Perdana[1], Nurul Ain Farhana[2]

[1]Program Studi Ilmu Komputer, Jurusan Matematika, Universitas Negeri Medan,
[2]Program Studi Statistika, Jurusan Matematika, Universitas Negeri Medan

e-mail: [1]**adidtya@unimed.ac.id**, [2]nurulainfarhana@unimed.ac.id

ABSTRAK

Dalam mengamankan sebuah pesan khususnya pesan digital diperlukan sebuah teknik atau algoritma dalam bidang ilmu kriptografi. Dimana kriptografi sendiri merupakan bidang keilmuan dalam informatika dan ilmu komputer yang membahas serta mempelajari tentang pengamanan sebuah pesah rahasia. Salah satu algoritma atau dalam ilmu kriptografi dinamakan dengan Cipher, yang dapat digunakan untuk mengamankan pesan rahasia yaitu RCCM. RCCM atau *Rivest Cipher 4 base on Chaotic Mapping* merupakan sebuah pengembangan dari RC4 yang menggabungkannya dengan teknik *Chaotic Map*. RC4 sendiri merupakan algoritma kriptografi yang termasuk kedalam Stream Cipher. Dalam proses pengamanan pesan pada RC4, pembangkitan kunci dilakukan menggunakan KSA (*Key Schedulling Algorithms*) dan PRGA (*Pseudo-Random Generation Algorithm*) yang diproses berdasarkan kunci yang di inputkan pengguna menghasilkan *keystream*. *Keystream* sendiri terdiri atas deretan angka acak yang masing-masingnya akan di-*exclusive or*-kan (XOR) terhadap *plaintext* yang menhasilkan *ciphertext*. Dengan menerapkan *Chaotic Map* peng-input-an kunci oleh pengguna tidak perlu dilakukan. Dan untuk implementasinya dapat menggunakan pemrograman Python dan berbasis GUI menggunakan bantuan *library* Tkinter.

Kata kunci: Kriptografi; RC4; RCCM; Stream Cipher; Pesan Text; Python; Tkinter

ABSTRACT

In securing a message, especially digital messages, a technique or algorithm is needed in the field of cryptography. Where cryptography itself is a scientific field in informatics and computer science that discusses and learns about securing a secret message. One of the algorithms or in cryptography is called a Cipher, which can be used to secure secret messages, namely RCCM. RCCM or Rivest Cipher 4 base on Chaotic Mapping is a development of RC4 which combines it with the Chaotic Map technique. RC4 itself is a cryptographic algorithm that belongs to the Stream Cipher. In the process of securing messages on RC4, key generation is carried out using KSA (Key Scheduling Algorithms) and PRGA (Pseudo-Random Generation Algorithm) which are processed based on the key entered by the user to produce a keystream. The keystream itself consists of a series of random numbers, each of which will be exclusive or (XOR) to the plaintext that produces the ciphertext. By applying the Chaotic Map, inputting the key by the user is not necessary. And for implementation, you can use Python programming and are GUI-based using the help of the Tkinter library.

Keywords*:Cryptoragphy, RC4, RCCM, Stream Cipher; Message; Python; Tkinter*

1. PENDAHULUAN

Kriptografi merupakan salah satu cabang ilmu dari bidang informatika dan ilmu komputer. Kripgografi sendiri membahas mengenai seni dan ilmu dalam mengambankan suatu pesan rahasia agar tidak dapat diketahui makna pesan tersebut oleh orang-orang yang tidak berkepentingan. Dalam pengamanan sebuah pesan khususnya pesan rahasia diperlukan suatu algoritma yang mumpuni. Salah satu algoritma yang baik dalam pengamanan pesan yaitu *Stream Cipher RC4* (*Rivest Cipher 4*). Namun dalam makalah ini akan dibahas penerapan modifikasi dari RC4 yaitu RCCM (*Rivest Cipher 4 Base On Chaotic Mapping*) (Liu et al., 2020). Algoritma RCCM sendiri merupakan peningkatan dari algoritma Stream Cipher RC4 yang menggunakan Chaotic Mapping untuk melindungi privasi data.

Didalam penelitian yang dilakukan oleh (Liu et al., 2020) membahas mengenai perlindungan privasi pada data *spasio-temporal* di IoT. Dimana salah satu topik utamanya adalah modifikasi atau peningkatan RC4 menggunakan Chaotic Mapping. Chaotic Mapping digunakan sebagai salah satu komponen utama untuk meningkatkan keamanan dan kompleksitas algoritma RC4. Chaotic Map digunakan dalam RCCM untuk menghasilkan kunci yang kuat dan otomatis yang digunakan dalam proses enkripsi dan dekripsi data.

Pada dasarnya, Chaotic Map adalah sebuah fungsi matematis yang menghasilkan urutan angka yang tampak acak namun memiliki sifat khaotik. Dalam konteks RCCM, Chaotic Map digunakan untuk menghasilkan urutan angka yang digunakan sebagai input dalam algoritma RC4. Dengan menggunakan Chaotic Map, RCCM dapat menghasilkan keystream yang memiliki sifat khaotik dan sulit diprediksi, sehingga meningkatkan keamanan dari algoritma RC4 (El Batouty et al., 2020; Liu et al., 2020; Tanyildizi & Ozkaynak, 2019).

Pada penelitian terkait seperti yang dilakukan oleh (Zhang et al., 2020) membahas mengenai sebuah Model Komunikasi Hemat Energi dan Penyimpanan Terenkripsi yang didasarkan pada RC4 untuk Rekam Medis Elektronik (EHR) guna mengatasi tantangan konsumsi daya yang meningkat dan perlindungan privasi dalam era Internet of Things (IoT) medis. Model yang diusulkan menggunakan enkripsi RC4 untuk penyimpanan terenkripsi EHR dan mencakup teknik perlindungan privasi yang digunakan dalam komputasi awan untuk IoT medis. Didalamnya terdapat modifikasi dan peningkatan algoritma yaitu penggunaan Algoritma MedSecrecy yang didasarkan pada Kompresi Huffman dan Algoritma RC4. Yang bertujuan untuk mempertahankan efisiensi enkrpsi RC4, dan meningkatkan kerahasiaan, keacakan, dan keamanan aliran kunci.

Pada penelitian selanjutnya yang dilakukan oleh Febriyani dan Arfriandi, membahas tentang bagaimana menerapkan Algoritma RC4 dalam pengamanan dokuman soal ujian. Hasil dari penelitian ini adalah sebuah website yang dapat mengamankan file dokumen berekstensi *doc menggunakan algoritma RC4 (Febriyani & Arfriandi, 2021). Penelitian yang lainnya membahas mengenai pengamanan data file word menggunakan RC4 yang menghasilkan perangkat lunak yang mampu mengamankan data word berbasis biner (Saragi et al., 2020).

Dan pada penelitian yang dilakukan penulis dalam makalah ini akan menerapkan penggunaan Chaotic Mapping pada RC4 (algoritma modifikasi RCCM) dalam mengamankan pesan teks. Dan diimplementasikan menggunakan bahasa pemrograman Python dan library Tkinter untuk GUI-nya. Dan pada makalah ini juga penulis akan memberikan implementasi awal berbasis CLI.

2. METODE PENELITIAN

Dalam penelitian ini, penulis akan memaparkan tentang perhitungan dari RC4. Kemudian perhitungan dari Chaotic Mapping, Tahapan-tahapan dalam RCCM, Proses Enkrpsi dan Dekripsinya.

RC4 (Rivest Cipher 4)

Pada Algoritma RC4 terdapat 2 komponen utama yaitu KSA (*Key Schedulling Algorithm*) dan PRGA (*Pesudo-Random Generator Algorithm*) (Madarro-Capó et al., 2021). Pada KSA membangkitkan sebuah keadaan awal dari input-an key (kunci yang diinputkan pengguna) yang diinisialisasi sebagai S dengan nilai [0, 1, 2, …, 255]. Dan dapat dilihat pada algoritma berikut ini.

Algoritma RC4 Key Schedulling Algorithm

```
for i = 0 → 255 do
    S[i] ← i
end for
j ← 0
for i = 0 → 255 do
    j←(j + S[i] + K[i mod KeyLength]) mod 256
    Swap S[i], S[j]
end for
```

Hasil dari KSA akan digunakan pada perhitungan PRGA. Tujuan dari PRGA sendiri adalah untuk menghasilkan urutan keluaran byte yang akan digunakan untuk untuk proses Enkripsi dan Dekripsi RC4.

Algoritma RC4 Pseudo-Random Generator Algorithm

```
i ← 0
j ← 0
while Generating Output do
    i ← (i + 1) mod 256
    j ← (j + S[i]) mod 256
    Swap S[i], S[j]
    Output ← S[(S[i]+S[j]) mod 256]
end while
```

Tahapan-tahapan dalam Algoritma RC4 antara lain:

1. Inputkan Pesan teks sebagai plaintext (*p*) dan kunci (*K*).
2. Kunci *K* yang diinputkan akan diproses menggunakan KSA untuk mendapatkan nilai S.
3. Setelah nilai S ditemukan, selanjutnya akan diproses menggunakan PRGA untuk mendapatkan deret atau aliran (stream) kunci byte yaitu *Keystream*.
4. Gunakan Algoritma XOR terhadap *p* dan *keystream* untuk melakukan Enkripsi maupun dekripsi.

Chaotic Mapping

Chaotic Mapping atau pemetaan kacau/acak merupakan sebuah fungsi matematis yang menghasilkan urutan angka yang terlihat acak namun memiliki sifat khaotik. Pemetaan khaotik ini didasarkan pada konsep sistem dinamis non-linear, di mana setiap nilai input akan menghasilkan output yang sangat sensitif terhadap perubahan kecil pada input tersebut. Salah satu contoh pemetaan khaotik yang umum digunakan adalah Logistic Map. Logistic Map didefinisikan oleh persamaan $x(i+1) = \mu * x(i) * (1 - x(i))$, di mana x(i) adalah nilai input pada iterasi ke-i, dan μ adalah parameter yang mengontrol perilaku khaotik dari pemetaan ini. Ketika nilai μ berada dalam rentang $3.569 \leq \mu \leq 4$, Logistic Map akan beroperasi dalam keadaan khaotik. (El Batouty et al., 2020; Liu et al., 2020; Tanyildizi & Ozkaynak, 2019).

Chaotic Mapping memiliki beberapa karakteristik penting, diantaranya:

1. Sensitif terhadap kondisi awal: Perubahan kecil pada nilai awal input dapat menghasilkan perubahan yang signifikan pada urutan angka yang dihasilkan.
2. Ketidak terdugaan: Urutan angka yang dihasilkan oleh pemetaan khaotik tampak acak dan sulit diprediksi, bahkan jika inputnya diketahui.

3. Pengacakan yang kuat: Urutan angka yang dihasilkan oleh pemetaan khaotik memiliki sifat keacakan yang kuat, sehingga sulit untuk direkonstruksi atau diprediksi oleh pihak yang tidak berwenang.

RCCM

Dalam RCCM tetap menerapkan RC4 sebagai proses utamanya namun menggunakan Chaotic Mapping dalam penetapan Kunci *K*. Sehingga pengguna tidak perlu lagi meng-inputkan-kan kunci kedalam program. Adapun tahapan-tahapan dalam RCCM antara lain:

1. Inisialisasi Parameter: Pada tahap ini, parameter-parameter yang diperlukan dalam algoritma RCCM diinisialisasi. Parameter-parameter ini termasuk nilai μ, nilai awal x_0, dan panjang keystream yang diinginkan menggunakan algoritma *Chaotic Mapping*.
2. *Key Scheduling Algorithm* (KSA): Tahap ini melibatkan algoritma KSA, yang bertujuan untuk menghasilkan kunci yang akan digunakan dalam proses enkripsi dan dekripsi. Algoritma KSA menggunakan pemetaan khaotik (misalnya, Logistic Map) untuk menghasilkan urutan angka yang digunakan sebagai kunci.
3. *Pseudo-Random Generation Algorithm* (PRGA): Setelah kunci dihasilkan, tahap selanjutnya adalah PRGA. Algoritma PRGA menggunakan kunci yang dihasilkan dari tahap sebelumnya untuk menghasilkan keystream yang akan digunakan dalam proses enkripsi dan dekripsi data. Keystream ini dihasilkan dengan menggunakan operasi XOR antara kunci dan plaintext/ciphertext.
4. Enkripsi dan Dekripsi: Setelah keystream dihasilkan, tahap terakhir adalah enkripsi dan dekripsi data. Dalam enkripsi, plaintext di-XOR dengan keystream untuk menghasilkan ciphertext. Sedangkan dalam dekripsi, ciphertext di-XOR dengan keystream yang sama untuk mendapatkan kembali plaintext asli.

Tahapan-tahapan ini dilakukan secara berulang untuk setiap blok data yang akan dienkripsi atau didekripsi. Dengan menggunakan RCCM, data dapat diamankan menggunakan keystream yang dihasilkan dari *Chaotic Mapping*, sehingga meningkatkan keamanan dan keacakan dari proses enkripsi dan dekripsi.

3. HASIL DAN PEMBAHASAN

Pada bagian ini akan dijelaskan mengenai hasil dari penerapan RCCM untuk mengamankan pesan teks menggunakan Python. Hasil ini memaparkan 2 penerapan implementasi menggunakan Python, yang pertama program Python berbasis CLI, dan kedua prgoram Python berbasis GUI menggunakan library Tkinter. Selanjutnya akan dilakukan pengujian dari hasil penerapannya dan dibahas berdasarkan beberapa kriteria.

Impelementasi Python

Pada implementasi menggunakan python berbasis CLI, terdapat beberapa beberapa fungsi yang dituliskan agar program berjalan sesuai yang diinginkan. Diantaranya:

1. Fungsi chaotic map. Pada fungsi ini bertujuan untuk menghasilkan nilai chaotic mapping yang diinginkan.

```
# Fungsi Chaotic Mapping
# ------------------------------------------------------------
def chaotic_map(pi, x):
    return pi * x * (1 - x)
```

Gambar 1 Fungsi Chaotic Mapping

2. Fungsi generate key. Fungsi ini bertujuan untuk menghasilkan kunci secara otomatis berdasarkan inputan parameter jumlah. Tujuannya untuk mendapatkan nilai key yang akan di gunakan pada perhitungan KSA.

```
# Fungsi Pembangkitkan Kunci RC4
# ------------------------------------------------------------
def generate_key(length):
    key = array.array('B',[0] * length)

    for i in range(length):
        pi = random.uniform(3.569, 4)
        x = random.uniform(0, 1)
        x = chaotic_map(pi, x)
        key[i] = int(x * 256)

    return key
```

Gambar 2 Fungsi generate key

3. Fungsi KSA. Fungsi ini bertujuan untuk memproses key untuk menghasilkan nilai S yang berguna untuk proses PRGA.

```
# Fungsi KSA
# ------------------------------------------------------------
def ksa_rccm(key):
    S = [i for i in range(256)]
    j = 0
    for i in range(256):
        try:
            j = (j + S[i] + key[i % len(key)]) % 256
            S[i], S[j] = S[j], S[i]
        except IndexError:
            break
    return S
```

Gambar 3 Fungsi KSA

4. Fungsi PRGA. Fungsi ini bertujuan untuk menghasilkan keystream dan dapat digunakan untuk proses Enkripsi Dekripsi.

```
# Fungsi PRGA
# ------------------------------------------------------------
def prga_rccm(S):
    K = [i for i in range(256)]
    i = j = 0
    for i in range(256):
        try:
            i = (i + 1) % 256
            j = (j + S[i]) % 256
            S[i], S[j] = S[j], S[i]
            K[i] = S[(S[i] + S[j]) % 256]
        except IndexError:
            break
    return K
```

Gambar 4 Fungsi PRGA

5. Fungsi enkripsi/dekripsi. Untuk proses penyandian pesan.

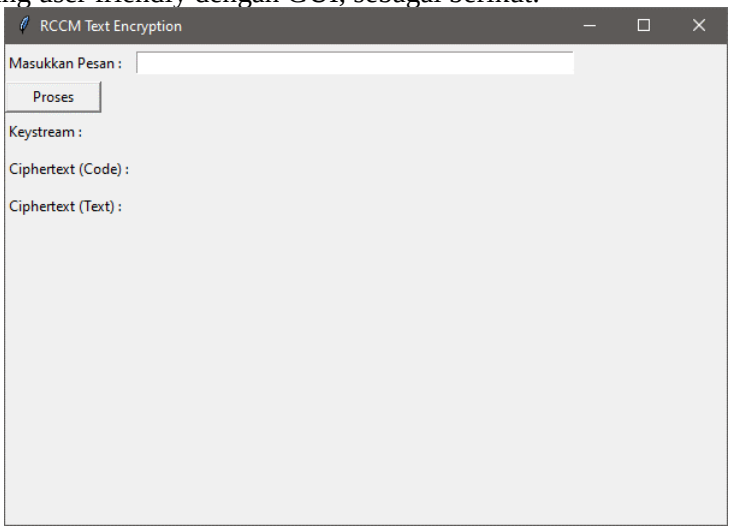
```
# Proses Enkripsi menggunakan RCCM
# ------------------------------------------------------------
def encrypt_rccm(text, key):
    ciphertext = []

    for i in range(len(text)):
        try:
            char = text[i]
            if (char.isupper()):
                ciphertext.append(ord(char) ^ key[i])
            else:
                ciphertext.append(ord(char) ^ key[i])
        except IndexError:
            break
    return ciphertext
```

Gambar 5 Fungsi Enkripsi

Implementasi Python dengan GUI

Pada implementasi Python dengan GUI menggunakan library Tkinter, fungsi-fungsi yang digunakan tetap sama dengan implementasi Python dengan CLI. Namun kelebihannya ada ditampilannya yang user friendly dengan GUI, sebagai berikut:



Gambar 6 Tampilan GUI Program

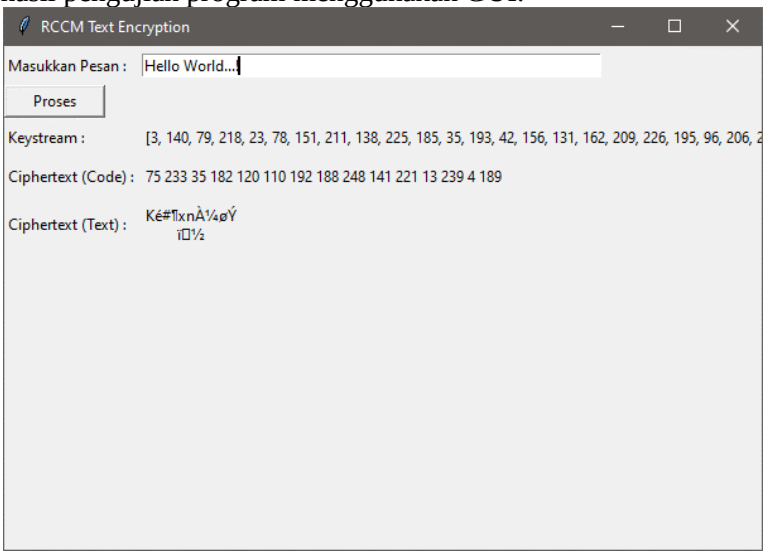
Pengujian

Dilakukan pengujian menggunakan sebuah pesan teks yaitu “Hello World…!”. Kemudian diproses menggunakan RCCM sehingga menghasilkan output seperti berikut ini untuk program CLI.

```
Pesan Asli :  Hello World...!
Keystream :  [240, 56, 184, 132, 252, 45, 48, 129, 232, 138, 73, 174, 42, 158, 191, 219, 172, 204, 164, 93, 224, 237, 213, 78, 23, 145, 204, 25,
241, 91, 222, 155, 214, 26, 25, 86, 245, 46, 46, 118, 83, 169, 23, 115, 254, 239, 143, 7, 61, 99, 234, 141, 197, 92, 163, 249, 172, 128, 136, 22
3, 220, 68, 252, 37, 33, 239, 181, 15, 113, 125, 144, 102, 121, 151, 175, 77, 210, 41, 164, 117, 192, 137, 82, 1, 36, 98, 18, 182, 125, 90, 246,
205, 43, 163, 139, 108, 142, 167, 7, 38, 51, 115, 0, 161, 124, 66, 167, 105, 116, 79, 60, 181, 40, 162, 225, 53, 127, 195, 99, 184, 182, 189, 25
, 237, 192, 201, 15, 9, 224, 93, 104, 187, 48, 193, 81, 13, 215, 45, 33, 138, 189, 166, 58, 132, 138, 2, 203, 107, 169, 91, 104, 54, 201, 141, 1
7, 30, 197, 149, 95, 219, 154, 190, 192, 219, 130, 6, 233, 182, 58, 28, 85, 212, 1, 167, 164, 21, 98, 99, 111, 121, 236, 245, 162, 224, 202, 183
, 24, 77, 173, 22, 208, 177, 165, 170, 146, 106, 70, 235, 64, 80, 6, 204, 11, 105, 213, 49, 83, 83, 125, 82, 87, 104, 87, 109, 168, 29, 252, 135
, 8, 58, 118, 197, 231, 121, 145, 79, 93, 31, 188, 71, 137, 192, 19, 159, 149, 7, 225, 199, 115, 107, 223, 80, 242, 28, 108, 252, 0, 37, 105, 24
4, 6, 40, 253, 102, 226, 81]
Ciphertext (Kode):  [184, 93, 212, 232, 147, 13, 103, 238, 154, 230, 45, 128, 4, 176, 158]
Ciphertext (Huruf):  ¸]Ôègîæ-°
Plaintext :  Hello World...!
```

Gambar 7 Hasil pengujian Program CLI

Dan berikut ini hasil pengujian program menggunakan GUI.



Gambar 8 Hasil Pengujian Program GUI

Pembahasan

Penelitian ini menggunakan Algoritma RCCM untuk pengamankan pesan rahasia. Algoritma RCCM ini merupakan pengembangan dari algoritma RC4 dengan mengkombinasikan menggunakan algoritma Chaotic Mapping. Implementasi dari RCCM pada penelitian ini menghasilkan program yang dibuat menggunakan bahasa python baik secara Command Line Interface (CLI) dan Graphical User Interface (GUI).

Dari hasil yang ditampilkan dari kedua implementasi berupa program baik yang berbasis CLI maupun GUI mampu memberikan hasil yang baik. Keduanya dapat memproses pesan teks menjadi pesan terenkripsi (ciphertext) dan dapat dikembalikan menjadi pesan asli lagi dengan cara di dekripsi.

Proses pembangkitan kunci menggunakan Chaotic Mapping, KSA, dan PRGA dapat diselesaikan dengan baik di kedua implementasi. Dan proses Enkripsi dan Dekripsi juga memberikan hasil yang baik.

4. KESIMPULAN

Dari uraian yang telah disampaikan maka dapat diambil kesimpulan sebagai beriktu:

1. Implementasi menggunakan Bahasa Python memberikan hasil yang baik dan sesuai dengan yang diharapkan dan sesuai dengan perhitungan secara manual.
2. Penerapan RCCM mampu meningkatkan keamanan dari pesan rahasia karena dengan penerapan Chaotic Mapping mampu meningkatkan keamanan dan kerahasiaan dari kunci K.
3. Dengan menerapkan RCCM, kelemahan-kelemahan yang dimiliki oleh RC4 dapat diminimalkan karena pengaruh algoritma Chaotic Mapping.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Medan sebagai lembaga yang menaungi penulis sehingga dapat menyelesaikan makalah ini dengan baik.

DAFTAR PUSTAKA

El Batouty, A. S., Farag, H. H., Mokhtar, A. A., El-Badawy, E. S. A., & Aly, M. H. (2020). Improvement of radio frequency identification security using new hybrid advanced encryption standard substitution box by chaotic maps. *Electronics (Switzerland)*, *9*(7), 1–14. https://doi.org/10.3390/electronics9071168

Febriyani, F. S., & Arfriandi, A. (2021). Implementasi Algoritma RC4 pada Sistem Pengamanan Dokumen Digital Soal Ujian. *JISKA (Jurnal Informatika Sunan Kalijaga)*, *6*(3), 171–177. https://doi.org/10.14421/jiska.2021.6.3.171-177

Liu, T., Wang, Y., Li, Y., Tong, X., Qi, L., & Jiang, N. (2020). Privacy Protection Based on Stream Cipher for Spatiotemporal Data in IoT. *IEEE Internet of Things Journal*, *7*(9), 7928–7940. https://doi.org/10.1109/JIOT.2020.2990428

Madarro-Capó, E. J., Legón-Pérez, C. M., Rojas, O., & Sosa-Gómez, G. (2021). Measuring avalanche properties on RC4 stream cipher variants. *Applied Sciences (Switzerland)*, *11*(20). https://doi.org/10.3390/app11209646

Saragi, D. R., Gultom, J. M., Tampubolon, J. A., & Gunawan, I. (2020). Pengamanan Data File Teks (Word) Menggunakan Algoritma RC4. *Jurnal Sistem Komputer Dan Informatika (JSON)*, *1*(2), 114. https://doi.org/10.30865/json.v1i2.1745

Tanyildizi, E., & Ozkaynak, F. (2019). A New Chaotic S-Box Generation Method Using

Parameter Optimization of One Dimensional Chaotic Maps. *IEEE Access*, *7*, 117829–117838. https://doi.org/10.1109/ACCESS.2019.2936447

Zhang, J., Liu, H., & Ni, L. (2020). A Secure Energy-Saving Communication and Encrypted Storage Model Based on RC4 for EHR. *IEEE Access*, *8*, 38995–39012. https://doi.org/10.1109/ACCESS.2020.2975208