Implementasi Keamanan Berbasis RSA untuk Pembuatan Kata Sandi Unik dalam Login Website

Muhammad Zaki - 13522136
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13522136@mahasiswa.itb.ac.id

***Abstract*— Sandi merupakan salah satu faktor penting dalam keamanan sistem informasi. Sandi yang lemah dapat dengan mudah dibobol oleh penyerang. Salah satu cara untuk membuat sandi yang kuat adalah dengan menggunakan algoritma RSA. Algoritma RSA adalah algoritma kriptografi asimetris yang menggunakan kunci publik dan kunci privat untuk mengenkripsi dan mendekripsi data.**

***Kata Kunci*— keamanan, kriptografi, kata sandi, RSA, login, website**

I. PENDAHULUAN

Di dunia yang sudah serba digital seperti saat ini, sebuah *website* sudah menjadi hal yang sangat lumrah. Kebergantungan manusia terhadap teknologi ini sangat signifikan, manusian tidak akan bisa terlepas dari penggunaan *website* karena ia adalah sebuah portal bagi informasi, penjualan, komunikasi, dan juga hiburan.

Manusia sangat bergantung pada *website* dalam segala hal seperti transaksi perbankan, belanja barang atau hal lainnya, sarana berkomunikasi pada teman ataupun keluarga. Akan tetapi karena kita membutuhkan banyak sekali ragam *website* kita membutuhkan ragam kata *sandi* yang kuat dan juga unik agar sulit untuk menjadi target kejahatan.

Gambar 1: Website,
diakses dari https://bakri.uma.ac.id/

Kata sandi yang lemah, perangkat lunak yang tidak di maintain secara berkala, dan juga implemntasi kode yang tidak aman dapat menciptakan kerentanan yang dapat dieksploitasi oleh penjahat cyber. Inilah mengapa keamanan situs web sangat penting.

Salah satu langkah untuk meningkatkan keamanan situs web terletak dalam ranah kriptografi, khususnya algoritma RSA (Rivest-Shamir-Adleman). Algoritma ini memiliki potensi untuk meningkatkan keamanan dalam menjelajah sebuah *website*.

Seiring dengan banyaknya algoritma kriptografi yang telah digunakan atau dikembangkan untuk menghasilkan kata sandi yang kuat, makalah ini akan membahas implementasi khusus dari algoritma RSA. Tujuan utamanya adalah untuk menciptakan kata sandi yang unik untuk setiap situs web, mengurangi risiko penggunaan kata sandi yang sama untuk beberapa situs web yang dapat meningkatkan potensi kebocoran data pengguna. Implementasi algoritma RSA untuk pembuatan sandi yang unik diharapkan dapat memberikan lapisan keamanan tambahan, melindungi data pengguna dari ancaman yang mungkin terjadi akibat penggunaan kata sandi yang repetitif.

II. LANDASAN TEORI

A. Teori Bilangan

Salah satu bidang studi pada Matematika Diskrit membahasa mengenai Teori Bilangan. Teori bilangan sendiri merupakan cabang matematika murni yang ditujukan untuk mempelajar bilangan bulat (integer) atau fungsi bernilai bilangan bulat.

Bilangan bulat (integer) adalah bilangan yang tidak mempunyai pecahan desimal, misalnya 8,21,8765, -34,0,-13451,dsb.

Berlawanan dengan bilangan bulat adalah bilangan riil yang mempunyai titik desimal, seperti 8.0, 34.25, 0.02, -0.00234.[1]

B. Sifat Pembagian pada Bilangan Bulat dan Teorema Euclidean

Adapun sifat pembagian pada bilangan bulat, misalkan a dan b bilangan bulat, $a \neq 0$. Maka a habis membagi b jika terdapat bilangan bulat c sedemikian sehingga $b = ac$. Notasi: $a \mid b$ jika $b = ac$, $c \neq Z$ dan $a \neq 0$.

Hal ini sejalan dengan teorema euclidean yang berbunyi sebagai berikut:

Misalkan m dan n bilangan bulat, $n > 0$. Jika m dibagi dengan n maka terdapat bilangan bulat unik q (quotient) dan r (remainder), sedemikian sehingga $m = nq + r$, dengan $0 \leq r < n$

C. Relatif Prima dan Aritmatika Modulo

Dua buah bilangan misal a dan b dikatakan relatif prima apabila memenuhi PBB (a, b) = 1. Sebagai contoh bilangan 20 dan 3 adalah relatif prima dikarenakan PBB (20, 3) = 1,

sedangkan bilangan 20 dan 5 tidak relatif prima dikarenakan PBB (20,5) = 5.

Aritmatika modulo adalah salah satu metode aritmatika yang menyelesaiakan permasalahan mengenai bilangan bulat. Ide dari aritmatik modulo itu sendiri adalah melihat hasil sisa pembagian dari dua buah bilangan. Sebagai contoh misalkan a dan m bilangan bulat dengan m > 0, operasi a mod m, memberi sisa a dibagi oleh m.

Sebagai contoh jika kita mempunyai dua buah bilangan 23 dan 3, jika kita menggunakan operasi modulo akan menjadi, 23 mod 5 yang akan menghasilkan 3. Operasi modulo a mod m = r dapat dijabarkan sebagai a = mq + r, dengan $0 \leq r < m$

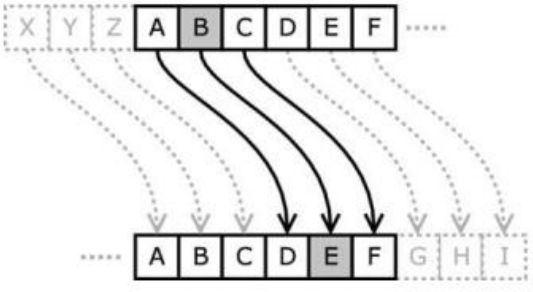
D. Kriptografi

Kata kriptografi atau *cryptography* diketahui berasal dari bahasa Yunani, kripto dan graphia. Dimana kripto memiliki arti menyembunyikan, sementara graphia berarti tulisa. Sehingga bisa dijabarkan kriptografi merupakan ilmu yang mempelajar teknik-teknik matematika yang berkaitan dengan aspek keamanan informasi.

Kriptografi menurut catatan sejarah telah eksis sejak masa kejayaan Yunani atau kurang lebih sekitar 400 tahun sebelum Masehi. Berdasarkan aspek historis kriptografi, baik kriptografi klasik maupun modern keduanya memiliki kesamaan prinsip yang besar yakni adalah keamanan.

Saat ini, perkembangan kriptografi telah mencakup berbagai paradigma, termasuk kriptografi klasik yang melibatkan metode-metode kuno seperti substitusi dan transposisi, serta kriptografi modern yang memanfaatkan algoritma-algoritma kompleks seperti RSA, AES, dan elliptic curve cryptography (ECC). Kriptografi modern menawarkan tingkat keamanan yang tinggi dengan memanfaatkan perhitungan matematis yang sulit dipecahkan.

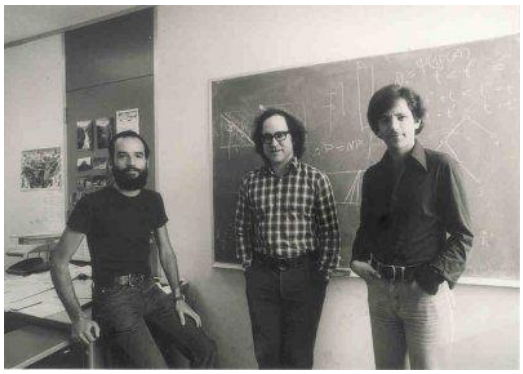
Salah satu algoritma kriptografi yang terkenal adalah Caesar Cipher. Konsep dasar dari Caesar Cipher adalah dengan melakukan pergeseran huruf sebanyak N posisi ke kanan atau ke kiri dalam alfabet. Misalnya, dengan pergeseran tiga posisi ke kanan (atau ke kiri), huruf A akan menjadi D (atau huruf Z jika pergeseran dilakukan ke kiri). Proses ini dilakukan secara seragam pada setiap huruf dalam teks pesan.



Gambar 2: Ilustrasi Caesar Cipher ROT 3
Sumber: [3]

E. Algoritma RSA

Algoritma Rivest Shamir Adleman (RSA) adalah algoritma untuk enkripsi kunci public (*public-key encryption)*. Algoritma RSA dijabarkan pada tahun 1977 oleh Ron Rivest, Adi Shamir dan Len Adleman dari *Massachusetts Institute of Technology* (MIT). Huruf RSA itu sendiri juga berasal dari inisial nama mereka (Rivest—Shamir— Adleman).



Gambar 3: Rivest—Shamir— Adleman
Diakses dari https://cryptologicfoundation.org/

Algoritma kriptografi didesain sesuai fungsinya sehingga menghasilkan kunci yang digunakan untuk enkripsi berbeda dari kunci yuang digunakan untuk dekripsi pesan. Algoritma RSA disebut menggunakan kunci publik karena kunic enkripsi yang dibuat boleh diketahui semua orang, dan juga bisa melakukan enkripsi pesan tersebut.

Keamanan algoritma RSA didasarkan pada sulitnya memfaktorkan bilangan besar menjadi faktor – faktor primanya. Adapun langkah – langkah pembuatan kunci antara lain:

1. Pilih dua bilangan prima, p dan q (rahasia)
2. Hitung n = pq. Besaran n tidak perlu dirahasiakan
3. Hitung m = (p-1) (q-1) (rahasia)
4. Pilih sebuah bilangan bulat untuk kunci publik, e yang relatif prima terhadap m, yaitu PBB (e, m) = 1
5. Hitung kunci dekripsi, d, melalui kekongruenan $ed \equiv 1 \pmod{m}$
   [3]

Algoritma kriptografi, khususnya algoritma RSA, dirancang dengan cermat untuk memenuhi kebutuhan keamanan informasi dalam proses enkripsi dan dekripsi. Dalam algoritma RSA, kunci yang digunakan untuk enkripsi dan dekripsi berbeda, dikenal sebagai kunci publik dan kunci privat. Kunci publik digunakan untuk enkripsi pesan dan bisa diketahui oleh siapa pun, sementara kunci privat hanya diketahui oleh penerima pesan dan digunakan untuk melakukan dekripsi.

Dengan langkah-langkah ini, algoritma RSA mampu menghasilkan pasangan kunci yang diperlukan untuk melakukan proses enkripsi dan dekripsi yang aman. Penggunaan bilangan prima, operasi matematika, dan kriteria keprimaan bilangan e sebagai bagian dari pembuatan kunci-kunci ini memainkan peran penting dalam memastikan keamanan dan kehandalan algoritma RSA dalam menjaga kerahasiaan pesan-pesan yang dikirimkan.

F. Kata Sandi

Kata sandi atau password memiliki peran krusial dalam pengamanan informasi pribadi maupun sensitif dalam berbagai platform digital. Sebagai kunci akses, kata sandi bertindak

sebagai mekanisme pertahanan pertama yang melindungi data dari akses yang tidak sah. Konsep keamanan kata sandi didasarkan pada prinsip bahwa kata sandi yang kuat dan unik dapat mengurangi risiko akses ilegal atau peretasan terhadap akun atau informasi yang dijalankan oleh pengguna.

Kekuatan sebuah kata sandi sangat bergantung pada kompleksitasnya, yang mencakup panjang karakter, penggunaan karakter yang beragam (huruf besar dan kecil, angka, simbol), serta ketidakmudahan dalam ditebak atau ditebus. Penggunaan kata sandi yang unik untuk setiap akun atau platform menjadi faktor penting dalam mencegah penyebaran risiko apabila satu kata sandi tertentu terkena kompromi. Secara umum, kebijakan yang disarankan adalah menghindari penggunaan kata sandi yang sama di beberapa platform yang berbeda.

G. Website

Website atau situs web merupakan sekumpulan halaman yang berisi berbagai informasi baik dalam bentuk teks, audio, gambar ataupun hal lainnya. Website memiliki berbagai fungsi, mulai dari menyediakan informasi, mempromosikan produk atau layanan, berbagi konten, hingga menjadi sarana interaksi sosial. Dalam era digital, website menjadi sarana yang penting untuk berkomunikasi dan berinteraksi di lingkungan online. Dengan penggunaan yang tepat, website dapat menjadi alat yang efektif untuk mencapai berbagai tujuan.

III. PEMBAHASAN

A. Perancangan kode

Pada makalah ini penulis menggunakan bahasa pemograman python untuk merancang enkripsi RSA. Perancangan enkripsi ini dilakukan dengan menggunakan library tambahan yaitu pycrpytodome yang mempunyai fungsi RSA didalamnya. Berikut adalah implementasi dari kode algoritma enkripsi ini:

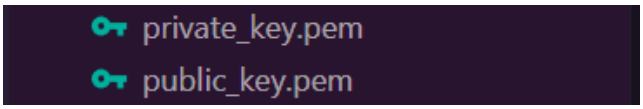
```
from Crypto.PublicKey import RSA

def generate_rsa_key():
    key = RSA.generate(1024)
    return key

def save_keys(public_key_filename, private_key_filename, rsa_key):
    # Menyimpan kunci publik
    with open(public_key_filename, 'wb') as public_file:
        public_file.write(rsa_key.publickey().export_key('PEM'))
    # Menyimpan kunci privat
    with open(private_key_filename, 'wb') as private_file:
        private_file.write(rsa_key.export_key('PEM'))

if __name__ == "__main__":
    try:
        rsa_key = generate_rsa_key()
        save_keys('public_key.pem', 'private_key.pem', rsa_key)
        print("Kunci publik dan privat RSA telah berhasil dibuat dan disimpan.")
    except Exception as e:
        print(f"Terjadi kesalahan: {e}")
```

Kode diatas untuk membuat public key dan juga private key yang akan disimpan pada file private_key.pem dan juga public_key.pem. RSA yang di generate merupakan RSA dengan modulus 1024, Pembuatan pasangan kunci publik dan privat dalam algoritma RSA dengan modulus 1024 merupakan langkah krusial dalam kriptografi asimetris. Private key dihasilkan dengan meng-generate bilangan prima secara acak yang besar, yang kemudian digunakan sebagai komponen utama dalam kunci privat. Proses ini melibatkan langkah matematis khusus yang memastikan keamanan kunci tersebut. Format PEM, di mana informasi kunci privat disimpan dalam file private_key.pem dan kunci publik disimpan dalam file public_key.pem.



Gambar 4: File private dan public key
Sumber: dokumentasi pribadi

Gambar 5: Contoh hasil private key
Sumber: dokumentasi pribadi

```
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
import random
import string

def add_padding(message):
    symbols = string.punctuation
    padding = ''.join(random.choices(symbols, k=5))
    # Menggabungkan pesan dengan padding
    padded_message = padding + message
    return padded_message

def load_public_key(public_key_filename):
    # Membaca kunci publik dari file
    with open(public_key_filename, 'rb') as public_file:
        public_key = RSA.import_key(public_file.read())
    return public_key
```

```
def load_private_key(private_key_filename):
    # Membaca kunci privat dari file
    with open(private_key_filename, 'rb') as private_file:
        private_key = RSA.import_key(private_file.read())
    return private_key


def decrypt_pass_with_rsa(encrypted_message, private_key):
    cipher_rsa = PKCS1_OAEP.new(private_key)
    decrypted_message =
cipher_rsa.decrypt(bytes.fromhex(encrypted_message))
    unpadded_message = decrypted_message
    return unpadded_message.decode()

def encrypt_pass_with_rsa(message, public_key):
    cipher_rsa = PKCS1_OAEP.new(public_key)
    encrypted_message = cipher_rsa.encrypt(message.encode())
    return encrypted_message.hex()


if __name__ == "__main__":
    try:
        public_key = load_public_key('public_key.pem')
        private_key = load_private_key('private_key.pem')

        message = input("Masukkan situs: ")

        # Melakukan enkripsi pesan dengan kunci publik RSA
yang telah disimpan
        encrypted_pass = encrypt_pass_with_rsa(message,
public_key)
        final_pass = add_padding(encrypted_pass)[:12]
        print(f"Password Anda: {final_pass}")

        # Melakukan dekripsi pesan yang telah dienkripsi
        decrypted_pass = decrypt_pass_with_rsa(encrypted_pass,
private_key)
        print(f"Password yang Didekripsi: {decrypted_pass}")

    except Exception as e:
        print(f"Terjadi kesalahan: {e}")
```

Pada kode diatas saya memotong enkripsinya hanya menjadi 7 karakter awal saja dan ditambahkan beberapa simbol diawal agar password semakin susah untuk ditebak. Pemotongan enkripsi itu dikarenakan jika tidak dipotong password akan sangat panjang dan biasanya website tidak support untuk menyimpan sebuah password yang sangat panjang. Jadi penulis memutuskan untuk hanya mengambil 7 karakter awal saja ditambahkan dengan 5 simbol diawalnya agar kata sandi semakin sulit untuk ditebak.

B. Setup perangkat lunak

Gambar 6: Python
Diakses dari: medium.com

Penulis disini memanfaatkan library python yaitu pycryptodome untuk melakukan enkripsi RSA. Anda bisa menambahkan library ini ke perangkat anda dengan catatan sudah memiliki python sebelumnya, dengan menjalankan command ini pada terminal:

```
pip install pycryptodome
```

C. Pengujian Enkripsi Data

```
Masukkan situs: google.com
Password Anda: +[+_+760b16d
```

Gambar 7: Hasil pengujian pertama
Sumber: dokumentasi pribadi

Dalam proses pengujian algoritma RSA pada gambar 8 yang disajikan dalam dokumentasi pribadi, telah terbukti bahwa enkripsi dan dekripsi berhasil dilakukan terhadap situs web google.com. Langkah-langkah tersebut melibatkan pengambilan 7 enkripsi awal dari RSA dan menambahkan 5 simbol acak di depannya untuk membentuk sebuah sandi atau password yang kemudian dianalisis dengan private key yang sesuai.

Untuk menguji apakah algoritma atau kode RSA yang kita buat valid kita coba untuk melihat apakah kode RSA kita bisa di decrypt atau di pecahkan dengan private key kita.

```
Hasil Enkripsi RSA: 760b16d18e941cc7fcf64e52e5132852b4ad95c6a9eb296c841
2d2d300aa164dcce2bd1f759255b77a0617dab637b1ace5d109dd5e7b15a8ae477ef84a
a4ee671bb18a7513707a5c29a5df6b862ef5ea007bf43f48368c79989c108052f59d735
72f92530f6665f39f5496ce34d98328b86d552bfeef879cf451529ab05b7f51
Password yang Didekripsi: google.com
```

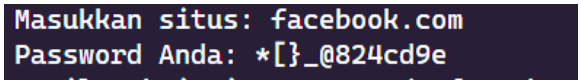
Gambar 8: Hasil pengujian algoritma RSA
Sumber: dokumentasi pribadi

Dari hasil yang tergambar, validitas algoritma atau kode RSA yang dikembangkan telah teruji, memungkinkan penggunaan private key untuk melakukan dekripsi terhadap enkripsi yang telah dihasilkan sebelumnya. Proses ini memberikan keyakinan bahwa algoritma yang telah dibuat dapat memperoleh kembali pesan atau informasi asli dari hasil enkripsi, seperti yang terdemonstrasikan dengan pengujian terhadap situs google.com.

Dengan demikian, penggunaan teknik RSA dalam enkripsi dan dekripsi telah terverifikasi dan berhasil pada implementasi pengujian yang dilakukan terhadap situs web yang disebutkan.

Selanjutnya untuk membuktikan password yang digenerate

itu unik kita akan mencoba lagi tetapi menggunakan input situs yang berbeda.



Gambar 9: Hasil pengujian pada situs facebook.com
Sumber: dokumentasi pribadi

Melalui percobaan yang dilakukan, terlihat bahwa dengan menggunakan input yang berbeda, yaitu situs yang berbeda pula, hasil enkripsi yang dihasilkan juga berbeda atau unik.

Hal ini memvalidasi bahwa proses pembuatan password yang terdiri dari 7 enkripsi awal RSA ditambah dengan 5 simbol acak di depannya, menghasilkan kunci atau sandi yang unik untuk setiap input yang berbeda. Dengan kata lain, implementasi algoritma RSA yang telah dikembangkan mampu menciptakan kunci enkripsi yang berbeda-beda untuk setiap konten atau situs yang dienkripsi.

Dengan adanya bukti ini, keunikan dari sandi yang dihasilkan pun terbukti, menegaskan bahwa algoritma atau kode RSA yang dibuat mampu memproduksi kunci yang spesifik dan berbeda untuk setiap input yang digunakan dalam proses enkripsi.

D. Analisis Hasil

Pada Makalah ini penulis menggunakan bahasa pemograman python yang digunakan juga library pycryptodome untuk melakukan enkripsi maupun dekripsi algoritma RSA. Perancangan perangkat lunak untuk proses enkripsi-dekripsi dilakukan dengan Visual Studio Code

Keunggulan utama dari penggunaan algoritma RSA dalam proses enkripsi adalah tingkat keamanannya yang tinggi. Algoritma ini terkenal karena kekuatan enkripsinya yang handal, yang tergantung pada kesulitan faktorisasi bilangan besar. Hal ini membuatnya sangat cocok untuk aplikasi yang membutuhkan tingkat keamanan yang tinggi dalam pertukaran data, seperti dalam perlindungan informasi rahasia, data transaksi keuangan, atau komunikasi pribadi.

Dengan memilih bahasa pemrograman Python dan memanfaatkan library pycryptodome, penulis berhasil mengimplementasikan algoritma RSA secara efisien dan efektif. Penggunaan Python sebagai bahasa pemrograman memudahkan dalam penulisan kode yang bersih dan mudah dipahami, sementara pycryptodome menyediakan fungsi-fungsi kriptografi yang kuat untuk menjalankan proses enkripsi dan dekripsi.

Perangkat lunak yang dirancang dapat diandalkan untuk mengamankan pesan atau data sensitif melalui enkripsi RSA, serta memungkinkan dekripsi yang tepat dengan menggunakan kunci pribadi yang sesuai. Dengan demikian, penggunaan algoritma RSA dalam konteks ini tidak hanya menawarkan tingkat keamanan yang tinggi, tetapi juga memberikan kemudahan dalam implementasi melalui penggunaan bahasa pemrograman Python dan library pycryptodome.

Menurut Federal Trade Commision atau FTC, password yang kuat adalah password yang setidaknya memiliki 12 karakter, yang bisa dibuat dengan frasa sandi dari kata-kata acak.[5]

$$E = log_2(R^L)$$

Gambar 10: Rumus entropy
Sumber: https://specopssoft.com

Dari rumus pada gambar 10 kita bisa menghitung kekuatan password yang kita buat, dimana R adalah jumlah karakter yang mungkin dan L adalah panjang password.

R dalam password kita adalah 90, dimana kita dapatkan dari jumlah set huruf besar + jumlah set huruf kecil + angka + simbol. L nya adalah 12. Jika kita hitung maka kita akan mendapat E sekitar 77 bits. Dari hasil tersebut kita mendapat kira kira terdapat 2^{77} kemungkinan password. Dengan kombinasi sebanyak itu maka akan sulit dan lama untuk dipecahkan.

E. Kekurangan

Kekurangan yang teridentifikasi pada implementasi dalam makalah ini adalah ketidakpastian dalam penghasilan hasil enkripsi yang sama untuk input yang identik, bahkan jika menggunakan pasangan kunci (public key atau private key) yang sama. Hal ini menunjukkan bahwa algoritma yang diusulkan tidak dapat diandalkan sebagai sebuah solusi untuk manajemen kata sandi atau password manager yang handal.

Keamanan sebuah password manager sangat bergantung pada sifat unik dari setiap hasil enkripsi yang dihasilkan dari input yang sama, dengan kunci yang sama pula. Namun, pada kasus di mana implementasi algoritma RSA ini menghasilkan keluaran yang berbeda untuk input yang serupa, terdapat kelemahan yang signifikan. Sebagai contoh, jika sebuah password manager menggunakan algoritma ini, sulit untuk secara konsisten memperoleh hasil enkripsi yang sama untuk kata sandi yang sama.

Ketidakpastian ini mengurangi kehandalan sistem manajemen kata sandi, karena pengguna biasanya mengandalkan pada kesamaan hasil enkripsi untuk mengakses kata sandi yang sama di waktu yang berbeda. Dengan hasil enkripsi yang bervariasi untuk input yang sama, keandalan dan konsistensi dari sistem manajemen kata sandi terganggu.

V. KESIMPULAN

Di dunia yang sudah serba digital ini keamanan sebuah informasi tidak dapat diabaikan dan menjadi hal yang sangat krusial. Penerapan algoritma RSA ini dapat memainkan peran yang signifikan dalam menjaga keamanan informasi. Algoritma RSA sebagai metode kriptografi asimetris telah terbukti menjadi salah satu pilihan yang populer dalam mengamankan data sensitif.

Dalam konteks penggunaan algoritma RSA untuk menghasilkan sandi atau password, penting untuk memastikan keunikan dari setiap sandi yang dihasilkan. Hal ini diperlukan agar setiap password yang digunakan untuk akses ke berbagai situs web memiliki karakteristik yang unik. Dengan demikian, meskipun pengguna menggunakan email yang sama untuk beberapa situs, sandi yang dihasilkan dari algoritma RSA akan bervariasi, meningkatkan keamanan akun di berbagai platform online.

Dengan demikian, penerapan algoritma RSA dapat menjadi solusi yang kuat untuk menghasilkan sandi yang unik dan meningkatkan tingkat keamanan di era digital ini. Evaluasi terus menerus diperlukan untuk menghadapi tantangan yang muncul, guna memastikan bahwa algoritma ini dapat diandalkan sebagai bagian integral dari strategi keamanan informasi yang efektif.

VII. UCAPAN TERIMAKASIH

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa yang telah memberikan karunia, sehingga penulis dapat menyelesaikan makalah yang berjudul "Implementasi Keamanan Berbasis RSA untuk Pembuatan Kata Sandi Unik dalam Login Website" yang selesai tepat pada waktunya. Tak lupa juga penulis mengucapkan terimakasih kepada kepada Bapak Dr. Ir. Rinaldi Munir, MT. dan juga Monterico Adrian, S.T., M.T. sebagai dosen pengampu mata kuliah IF2120 Matematika Diskrit atas bimbingan dan pengajaran yang telah dilakukan dikelas Matematika Diskrit ini. Terakhir, penulis mengucapkan terimakasih kepada orang tua, keluarga, dan seluruh pihak yang membantu penulis dalam menyelesaikan makalah ini.

REFERENSI

[1] R. Munir, "Aljabar Boolean (Bag. 1)." Accessed: Dec. 08, 2023. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf (accessed Dec. 08, 2023).

[2] A. PUTRA, "IMPLEMENTASI PEMROSESAN PARALEL PADA ALGORITMA RIVEST SHAMIR ALDEMAN (RSA) MENGGUNAKAN CLUSTER BEOWULF," *repository.uin-suska.ac.id*, 2017. http://repository.uin-suska.ac.id/id/eprint/17492 (accessed Dec. 08, 2023).

[3] R. Munir, Accessed: Dec. 9, 2023. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/16-Teori-Bilangan-Bagian3-2023.pdf(accessed Dec. 08, 2023).

[4] "Welcome to PyCryptodome's documentation — PyCryptodome 3.160b1 documentation,"www.pycryptodome.org. https://www.pycryptodome.org/(accessed Dec. 08, 2023)

[5] "The State of Password Security 2023 Report | Bitwarden Resources," Bitwarden. https://bitwarden.com/resources/the-state-of-password-security/ (accessed Dec. 11, 2023).

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 10 Desember 2023

Muhammad Zaki
13522136