Aplikasi Teori Bilangan dalam Kriptografi untuk Pengamanan Data dan Keamanan Informasi

Anella Utari Gunadi 13523078[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13523078@std.stei.itb.ac.id, anellautari@gmail.com

Abstrak—Keamanan data merupakan salah satu aspek yang sangat penting di era digital untuk melindungi informasi dari ancaman seperti pencurian data. Kriptografi menjadi solusi utama untuk mengamankan data dengan mengubah informasi menjadi bentuk yang tidak dapat dipahami tanpa kunci tertentu. Salah satu pendekatan dalam sistem kriptografi adalah penerapan teori bilangan, yang digunakan dalam algoritma seperti RSA dan Diffie-Hellman. Makalah ini bertujuan untuk mengkaji peran teori bilangan dalam menciptakan sistem keamanan data yang efektif. Hasil kajian menunjukkan bahwa teori bilangan memiliki peran penting dalam menciptakan sistem keamanan yang andal. Algoritma RSA unggul dalam proses enkripsi dan dekripsi, sementara Diffie-Hellman memungkinkan pertukaran kunci rahasia secara aman. Kombinasi kedua algoritma ini dapat memberikan perlindungan menyeluruh terhadap data, sehingga meminimalkan risiko pencurian informasi. Makalah ini diharapkan dapat memberikan wawasan baru dan menjadi dasar untuk pengembangan sistem keamanan data yang lebih baik kedepannya.

Kata kunci—Keamanan data, teori bilangan, kriptografi, RSA, Diffie-Hellman.

Abstract—Data security is a very important aspect in the digital era to protect information from threats such as data theft. Cryptography is becoming the main solution for securing data by converting information into a form that cannot be understood without a certain key. One approach to cryptographic systems is the application of number theory, which is used in algorithms such as RSA and Diffie-Hellman. This paper aims to examine the role of number theory in creating an effective data security system. The study results show that number theory has an important role in creating a reliable security system. The RSA algorithm excels in the encryption and decryption process, while Diffie-Hellman allows secure exchange of secret keys. The combination of these two algorithms can provide comprehensive protection for data, thereby minimizing the risk of information theft. It is hoped that this paper can provide new insights and become the basis for developing better data security systems in the future.

Keywords—Data security, number theory, cryptography, RSA, Diffie-Hellman.

I. PENDAHULUAN

Keamanan data menjadi kebutuhan yang sangat penting di era digital ini. Kejahatan seperti pencurian data semakin marak terjadi dan dapat mengancam kerahasiaan dan integritas informasi. Oleh karena itu, pengamanan data dan keamanan informasi harus ditingkatkan untuk mengurangi risiko dan dampak dari berbagai kejahatan yang dapat merugikan banyak orang.

Salah satu cara untuk melindungi dan mengamankan data adalah dengan menggunakan kriptografi. Kriptografi adalah ilmu yang mengubah pesan menjadi bentuk yang tidak dapat dibaca atau dipahami, sehingga informasi tetap aman. Dalam penerapannya, terdapat berbagai pendekatan untuk membuat dan membangun sistem kriptografi, salah satunya adalah dengan menerapkan teori bilangan.

Teori bilangan berperan penting dalam menciptakan sistem keamanan data. Dalam teori ini data penting dapat diubah menjadi bentuk yang sulit dipahami, sehingga dapat mengurangi risiko pencurian. Algoritma seperti RSA dan Diffie-Hellman adalah contoh aplikasi teori bilangan dalam kriptografi modern.

Namun, penerapan teori bilangan dalam kriptografi memiliki berbagai tantangan. Tantangan tersebut mencakup efisiensi algoritma dan pengelolaan data besar. Oleh karena itu, diperlukan analisis dan eksperimen secara mendalam untuk mengatasi tantangan tersebut sekaligus mengevaluasi efektivitas kriptografi berbasis teori bilangan dalam melindungi data dan informasi.

Makalah ini bertujuan untuk mengeksplorasi sejauh mana teori bilangan dapat diterapkan dalam kriptografi untuk meningkatkan keamanan data dan informasi. Melalui analisis ini, diharapkan makalah ini dapat memberikan wawasan baru bagi pembaca mengenai penerapan teori bilangan dalam sistem keamanan informasi.

II. KAJIAN TEORI

2.1. Teori Bilangan

Teori bilangan adalah salah satu cabang dari matematika murni yang berfokus pada kajian tentang bilangan bulat (integer) dan fungsi yang bernilai bilangan bulat [1].

2.1.1. Pembagi Bersama Terbesar (PBB)

PBB adalah bilangan bulat terbesar a yang merupakan pembagi bersama dari b dan c, yaitu a membagi b $(a|b)$ dan a membagi c $(a|c)$ [2]. Dalam hal ini dapat dinyatakan bahwa PBB$(b, c) = a$.

Menurut teorema dari sumber [1], misalkan m dan n adalah bilangan bulat dengan $n > 0$, sehingga berlaku:

$$m = nq + r \text{ dengan } 0 \leq r < n$$

Maka, $PBB(m,n) = PBB(n,r)$.

2.1.2. Algoritma Euclidean

Algoritma Euclidean adalah algoritma yang ditemukan oleh seorang matematikawan Yunani yang bernama Euclides untuk menentukan PBB dari dua bilangan bulat [1].

Berdasarkan sumber [1], cara mencari PBB dari bilangan bulat tak-negatif m dan n serta memenuhi syarat $m \geq n$ menggunakan Algoritma Euclidean adalah sebagai berikut.

1. Jika $n = 0$, makan m adalah $PBB(m,n)$; hentikan proses. Namun, jika $n \neq 0$, lanjutkan ke langkah berikutnya.
2. Bagi m dengan n dan sisa bagi dari pembagian tersebut misalkan r.
3. Perbarui m menjadi n dan n menjadi r, lalu ulangi dari langkah pertama.

Misalkan kita mempunyai $m = 56$ dan $n = 12$ dan dipenuhi syarat $m \geq n$. Kita dapat mencari PBB(56,12) adalah sebagai berikut.

$$56 = 4 \,.\, 12 + 8$$
$$12 = 1 \,.\, 8 + 4$$
$$8 = 2 \,.\, 4 + 0$$

Sisa pembagian terakhir sebelum 0 adalah 4, maka $\text{PBB}(56,12) = 4$.

2.1.3. Relatif Prima

Berdasarkan sumber [1], dua bilangan bulat a dan b disebut relatif prima jika $PBB(a,b) = 1$. Jika a dan b relatif prima, maka terdapat bilangan bulat m dan n yang memenuhi persamaan:

$$ma + nb = 1$$

2.1.4. Aritmetika Modulo

Misalkan a dan m adalah bilangan bulat dengan $m > 0$, hasil dari $a \bmod m$ adalah sisa pembagian a oleh m[1]. Notasinya adalah:

$$a \bmod m = r$$

Dengan, $a = mq + r$ dan $0 \leq r < m$

Kekongruenan modulo n berarti dua bilangan a dan b memiliki sisa bagi yang sama ketika dibagi oleh n [3]. Dalam hal ini, a dikatakan kongruen dengan b modulo n, yang dapat ditulis sebagai berikut:

$$a \equiv b \pmod n$$

Misalnya, $40 \bmod 3 = 1$ dan $13 \bmod 3 = 1$, maka dapat ditulis sebagai berikut:

$$40 \equiv 13 \pmod 3$$

2.2. Kriptografi

Kriptografi adalah metode untuk melindungi informasi dalam komunikasi, sehingga hanya pengirim dan penerima yang dapat mengakses pesan tersebut [4]. Teknik ini berkaitan erat dengan enkripsi, yaitu proses mengacak data agar hanya pihak tertentu yang dapat memahaminya. Secara teknis, enkripsi mengubah teks biasa (plainteks) menjadi teks yang terenkripsi (cipherteks) yang hanya dapat dipahami dengan kunci khusus.

Berdasarkan sumber [4], kata "kriptografi" berasal dari bahasa Yunani kuno yaitu *kryptos* yang berarti "tersembunyi" dan *graphia* yang berarti "tulisan". Salah satu penggunaan awal kriptografi dilakukan oleh Julius Caesar, seorang pemimpin militer Romawi, untuk menyampaikan pesan rahasia kepada pasukannya.

Kriptografi memiliki beberapa fungsi sebagai berikut.

1. Melindungi informasi penting
   Kriptografi digunakan untuk menjaga kerahasiaan data dari pihak yang tidak berwenang. Proses ini melibatkan algoritma dengan sistem kunci, enkripsi, dan dekripsi.
   - **Enkripsi**: mengonversi plainteks menjadi cipherteks.
   - **Dekripsi:** mengubah cipherteks kembali menjadi plainteks.
2. Meningkatkan privasi dan keamanan data
   Aplikasi atau perangkat dengan sistem kriptografi dapat membantu meningkatkan privasi dan melindungi data pengguna. Keamanan ini didasarkan pada beberapa aspek utama:
   - **Kerahasiaan**: memastikan hanya pihak tertentu yang dapat mengakses informasi.
   - **Otentikasi**: menjamin keaslian informasi yang diterima.
   - **Integritas**: memastikan pesan yang dikirim tetap utuh tanpa perubahan selama proses pengiriman.
   - **Nir penyangkalan**: mencegah pengirim atau penerima menyangkal keterlibatan mereka dalam proses pengiriman atau penerimaan pesan.

Salah satu penerapan kriptografi modern adalah dengan menggunakan algoritma RSA (Rivest-Shamir-Adleman) dan Diffie-Hellman.

- **RSA**: digunakan untuk mengamankan data dengan metode enkripsi kunci publik, sehingga pesan hanya dapat didekripsi oleh penerima dengan kunci privat.
- **Diffie-Hellman**: berfungsi untuk pertukaran kunci secara aman antara dua pihak.

2.3. Algoritma RSA

RSA (Rivest-Shamir-Adleman) adalah metode kriptografi kunci publik yang bekerja dengan menggunakan operasi eksponensial pada modulo bilangan bulat $N(Z_N)$, di mana N adalah bilangan bulat yang dihasilkan dari perkalian dua faktor besar, yaitu *semi-prime* [5, seperti dikutip dalam Kiviharju, 2017].

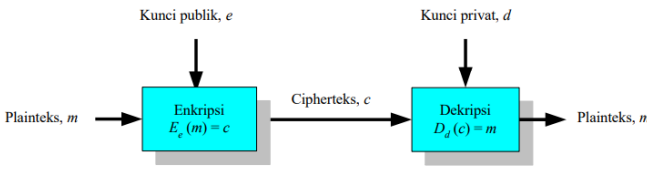
Dalam algoritma RSA untuk mengamankan pesan, terdapat beberapa langkah yang perlu dilakukan. Salah satu langkah yang paling dasar adalah proses ekspansi kunci. RSA menggunakan dua jenis kunci, yaitu kunci publik (untuk mengenkripsi atau menyandikan pesan) dan kunci privat (untuk mendekripsi atau mengembalikan pesan ke

bentuk aslinya). Proses ini menghasilkan sepasang kunci yang saling terhubung.

Berdasarkan sumber [6], langkah-langkah dalam melakukan ekspansi kunci adalah sebagai berikut:

1. Pilih dua bilangan prima besar p dan q, keduanya harus berbeda dan bersifat rahasia. Untuk mencapai tingkat keamanan yang tinggi, bilangan p dan q sebaiknya memiliki ukuran besar, misalnya 1024 bit.
2. Hitung $n = p \times q$ (bersifat publik)
3. Hitung $m = (p-1)(q-1)$ (bersifat rahasia)
4. Pilih nilai e untuk kunci publik yang relatif prima terhadap m, yaitu $PBB(e,m) = 1$. Proses ini dapat dilakukan menggunakan Algoritma Euclidean.
5. Cari d untuk kunci privat dengan rumus:
$$e \times d \bmod m = 1$$
$$ed \equiv 1 \ (mod \ m)$$
6. Kemudian, didapatlah sepasang kunci yang terdiri dari kunci publik = (e,n) dan kunci privat = (d,n).

Alur dari proses kriptografi dengan menggunakan algortima RSA dapat dilihat pada gambar di bawah ini.



Gambar 1. Diagram Proses Kriptografi RSA
Sumber:https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/17-Teori-Bilangan-Bagian3-2024.pdf

2.3.1. Enkripsi

Enkripsi adalah mengubah pesan asli menjadi bentuk sandi yang tidak dapat dibaca. Oleh karena itu, tujuan dari ekspansi kunci adalah memungkinkan pesan dienkripsi menggunakan kunci publik yang telah dihasilkan. Rumus dari enkripsi adalah sebagai berikut:

$$c = p^e \ (mod \ n)$$

Dengan, c = cipherteks
p = plainteks

2.3.2. Dekripsi

Dekripsi bertujuan untuk mengembalikan sandi yang telah dienkripsi menjadi pesan asli. Rumus dari dekripsi adalah sebagai berikut:

$$p = c^d \ (mod \ n)$$

2.4. Algoritma Diffie-Hellman

Metode pertukaran kunci Diffie-Hellman merupakan teknik yang aman untuk melakukan pertukaran kunci kriptografi [7]. Dengan metode ini, dua pihak yang sebelumnya tidak saling mengenal dapat menciptakan kunci rahasia bersama, meskipun komunikasi dilakukan melalui saluran komunikasi yang tidak aman.

Konsep ini menggunakan operasi perkalian dengan bilangan bulat dalam sistem modulo. Tanpa mengetahui kunci rahasia salah satu pihak, akan sangat sulit bagi orang lain untuk membongkar kunci tersebut secara matematis.

Algoritma Diffie-Hellman memungkinkan pertukaran kunci yang dapat digunakan untuk proses enkripsi dan dekripsi melalui beberapa pertukaran data di jaringan publik. Berdasarkan sumber [8], langkah-langkah melakukan pertukaran kunci menggunakan algoritma Diffie-Hellman adalah sebagai berikut:

1. Alice dan Bob adalah dua pihak yang ingin melakukan komunikasi aman dengan menggunakan pesan yang terenkripsi. Untuk itu, mereka perlu berbagi kunci rahasia yang akan digunakan untuk proses enkripsi dan dekripsi.
2. Alice dan Bob sepakat untuk memilih sebuah bilangan prima publik P dan bilangan dasar G, di mana $P > G$. Kedua bilangan ini bersifat publik dan digunakan sebagai parameter dasar dalam perhitungan dengan G adalah akar primitif dari P.
3. **Proses Alice:**
   - Alice memilih bilangan bulat acak X_A dengan syarat $1 \leq X_A \leq p-1$ sebagai kunci privatnya dan menjaganya agar tetap rahasia.
   - Alice menghitung nilai publik Y_A dengan menggunakan rumus:
   $$Y_A = G^{X_A} \bmod P$$
   - Alice mengirimkan Y_A kepada Bob.
4. **Proses Bob:**
   - Bob memilih bulangan bulat acak X_B dengan syarat $1 \leq X_B \leq p-1$ sebagai kunci privatnya dan menjaganya agar tetap rahasia.
   - Bob menghitung nilai publik Y_B dengan menggunakan rumus:
   $$Y_B = G^{X_B} \bmod P$$
   - Bob mengirimkan Y_B kepada Alice.
5. Setelah bertukar nilai Y_A dan Y_B, masing-masing pihak dapat menghitung kunci rahasia bersama tanpa membocorkan kunci privat mereka:
   - Alice menghitung kunci rahasia:
   $$K_A = Y_B^{X_A} \bmod P$$
   - Bob menghitung kunci rahasia:
   $$K_B = Y_A^{X_B} \bmod P$$
6. Berdasarkan sifat eksponensial modular, nilai K_A yang dihitung alice akan sama dengan nilai K_B yang dihitung Bob, yaitu:
$$K = K_A = K_B = G^{X_A \cdot X_B} \bmod P$$
7. Setelah mendapatkan kunci rahasia K, Alice dan Bob dapat menggunakan kunci ini untuk mengenkripsi dan mendekripsi pesan secara aman.

Jika ada pihak ketiga yang menyadap komunikasi antara Alice dan Bob, pihak tersebut hanya dapat mengetahui nilai $G, P, Y_A,$ dan Y_B. Namun, tanpa mengetahui nilai X_A dan X_B, penyadap tidak akan dapat menghitung kunci rahasia dari nilai-nilai tersebut.

Dengan algoritma ini, Alice dan Bob dapat saling bertukar pesan yang terenkripsi tanpa harus secara langsung mengirimkan kunci rahasia, menjadikannya metode yang aman untuk pertukaran kunci.

III. HASIL DAN PEMBAHASAN

3.1. Implementasi Algoritma RSA

Misalkan kita memiliki sebuah pesan dan ingin melakukan enkripsi pada pesan tersebut dengan isi pesan sebagai berikut.

Pesan = "MATEMATIKA DISKRIT"

Langkah pertama yang harus kita lakukan adalah memilih dua bilangan prima yang bersifat privat sebagai p dan q. Misal, kita pilih $p = 59$ dan $q = 73$.

Setelah memilih p dan q, kita dapat menghiutng nilai n dan m berdaasarkan rumus yang telah dijelaskan sebelumnya.

- Nilai n

$$\begin{aligned} n &= p \times q \\ &= 59 \times 73 \\ &= 4307 \end{aligned}$$

- Nilai m

$$\begin{aligned} m &= (p-1)(q-1) \\ &= (59-1)(73-1) \\ &= 58 \times 72 \\ &= 4176 \end{aligned}$$

Langkah selanjutnya adalah memilih nilai e sebagai kunci publik dengan syarat e relatif prima dengan m. Misal, dipilih $e = 23$ karena $PBB(23{,}4176) = 1$. Setelah memilih kunci publik, kita dapat menghitung kunci privat d dengan perhitungan sebagai berikut.

$$ed \equiv 1 \ (mod\ m)$$

$$d = \frac{1 + km}{e} = \frac{(1 + 4176k)}{23}$$

Dengan mencoba $k = 0, 1, 2, \ldots$, diperoleh nilai d bilangan bulat ada pada $k = 7$ yaitu $d = 1271$.

Langkah selanjutnya adalah mengubah setiap huruf menjadi angka dengan menggunakan representasi ASCII.

- M = 77
- A = 65
- T = 84
- E = 69
- I = 73
- K = 75
- D = 68
- S = 83
- R = 82

Langkah selanjutnya adalah mengenkripsi setiap huruf dengan rumus enkripsi RSA.

1. Huruf M: $c = 77^{23}\ mod\ 4307 = \ 3725$
2. Huruf A: $c = 65^{23}\ mod\ 4307 = \ 1761$
3. Huruf T: $c = 84^{23}\ mod\ 4307 = \ 744$
4. Huruf E: $c = 69^{23}\ mod\ 4307 = \ 2845$
5. Huruf I: $c = 73^{23}\ mod\ 4307 = \ 1825$
6. Huruf K: $c = 75^{23}\ mod\ 4307 = \ 3390$
7. Huruf D: $c = 68^{23}\ mod\ 4307 = \ 3736$
8. Huruf S: $c = 83^{23}\ mod\ 4307 = \ 533$
9. Huruf R: $c = 82^{23}\ mod\ 4307 = \ 4226$
10. Spasi: $c = 32^{23}\ mod\ 4307 = \ 3924$

Setelah memperoleh kode sandi untuk setiap huruf, susunlah kode tersebut sesuai urutan isi pesan. Jadi, hasil enkripsi yang telah didapat dari pesan "MATEMATIKA DISKRIT" adalah sebagai berikut:

- **Cipherteks** = [3725, 1761, 744, 2845, 3725, 1761, 744, 1825, 3390, 1761, 3924, 3736, 1825, 533, 3390, 4226, 1825, 744]

Langkah terakhir adalah melakukan dekripsi untuk mengembalikan sandi tersebut menjadi pesan asli dengan menggunakan rumus dekripsi RSA.

1. $p1 = 3725^{1271}\ mod\ 4307 = \ 77$ (M)
2. $p2 = 1761^{1271}\ mod\ 4307 = \ 65$ (A)
3. $p3 = 744^{1271}\ mod\ 4307 = \ 84$ (T)
4. $p4 = 2845^{1271}\ mod\ 4307 = \ 69$ (E)
5. $p5 = 1825^{1271}\ mod\ 4307 = \ 73$ (I)
6. $p6 = 3390^{1271}\ mod\ 4307 = \ 75$ (K)
7. $p7 = 3736^{1271}\ mod\ 4307 = \ 68$ (D)
8. $p8 = 533^{1271}\ mod\ 4307 = \ 83$ (S)
9. $p9 = 4226^{1271}\ mod\ 4307 = \ 82$ (R)
10. $p10 = \ 3924^{1271}\ mod\ 4307 = \ 32$ (spasi)

Dengan mencocokkan hasil dekripsi dengan daftar angka yang diperoleh dari proses enkripsi (cipherteks), pesan yang berhasil diungkap adalah "MATEMATIKA DISKRIT".

- **Plainteks** = MATEMATIKA DISKRIT

Implementasi pada python untuk algoritma RSA ini adalah dengan membuat fungsi `generate_rsa_keys` yaitu fungsi untuk menghitung nilai $n, m,$ dan d dari $p, q,$ dan e yang telah didefinisikan. Cara menghitung $m, n,$ dan d adalah dengan menggunakan rumus $d = e^{-1}\ mod\ m$.

```python
# Generate kunci RSA
def generate_rsa_keys():
    p = 59
    q = 73
    n = p * q
    m = (p - 1) * (q - 1)

    # kunci publik e
    e = 23

    # kunci privat d
    d = pow(e, -1, m)

    return (e, n), (d, n)  # kunci publik dan kunci privat
```

Gambar 2. Implementasi RSA pada python
Sumber: https://github.com/anellautari/Makalah-Matdis.git

Jika ingin menggunakan nilai dari $p, q,$ dan e secara acak oleh program tersebut, kita dapat membuat fungsi `is_prime` untuk memeriksa apakah suatu bilangan `num` adalah bilangan prima dengan mengecek apakah `num` habis dibagi oleh bilangan dari 2 hingga akar kuadrat `num`, jika

ada maka `num` bukan bilangan prima dan sebaliknya, serta fungsi `next_prime` untuk mencari bilangan prima berikutnya setelah bilangan n dengan menaikkan n sebesar 1 jika n bukan bilangan prima hingga ditemukan bilangan prima berikutnya. Implementasi dari kedua fungsi tersebut adalah sebagai berikut.

```python
import random
from math import gcd

# Fungsi untuk mencari bilangan prima berikutnya
def is_prime(num):
    if num < 2:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True

def next_prime(n):
    while not is_prime(n):
        n += 1
    return n
```

Gambar 3. Implementasi Fungsi is_prime dan next_prime
Sumber: https://github.com/anellautari/Makalah-Matdis.git

Cara menentukan nilai $p, q,$ dan e secara acak adalah dengan memanfaatkan kedua fungsi diatas dengan membuat fungsi bernama `generate_rsa_keys` serta menggunakan *library* dari python untuk mengambil angka secara *random* pada rentang yang dingiinkan (pada contoh implementasi, saya menggunakan rentang 50 hingga 100) serta memanggil fungsi `gcd` untuk menentukan nilai e yang relatif prima terhadap m sehingga dipenuhi syarat $PBB(e, m) = 1$. Implementasi programnya adalah sebagai berikut.

```python
def generate_rsa_keys():
    p = next_prime(random.randint(50, 100))
    q = next_prime(random.randint(50, 100))
    n = p * q
    m = (p - 1) * (q - 1)

    # kunci publik e yang relatif prima terhadap m
    e = random.randint(2, m - 1)
    while gcd(e, m) != 1:
        e = random.randint(2, m - 1)
```

Gambar 4. Modifikasi Fungsi generate_rsa_keys
Sumber: https://github.com/anellautari/Makalah-Matdis.git

Setelah membuat fungsi untuk menghitung nilai $p, q, n, m, e,$ dan d, langkah selanjutnya adalah dengan membuat fungsi `rsa_encrypt` yang berfungsi untuk mengubah pesan asli atau plainteks menjadi cipherteks menggunakan kunci publik (e, n) dengan rumus $c = p^e \ (mod\ n)$ serta fungsi `rsa_decrypt` yang berfungsi untuk mengembalikan cipherteks ke bentuk pesan asli atau plainteks menggunakan kunci privat (d, n) dengan rumus $p = c^d \ (mod\ n)$. Algoritma dari kedua fungsi tersebut adalah sebagai berikut.

```python
# Fungsi untuk enkripsi dan dekripsi
def rsa_encrypt(message, public_key):
    e, n = public_key
    return [pow(ord(char), e, n) for char in message]

def rsa_decrypt(ciphertext, private_key):
    d, n = private_key
    return ''.join([chr(pow(char, d, n)) for char in ciphertext])
```

Gambar 5. Fungsi Enkripsi dan Dekripsi pada Python
Sumber: https://github.com/anellautari/Makalah-Matdis.git

Dengan mencoba menginput *message* dengan "MATEMATIKA DISKRIT" serta mengatur nilai $p = 59$, $q = 73$, dan $e = 23$ agar hasilnya dapat dibandingkan dengan perhitungan manual, program RSA tersebut akan mengeluarkan hasil dari cipherteks dan plainteks. Hasil keluaran tersebut adalah sebagai berikut.

```
Pesan asli: MATEMATIKA DISKRIT
Cipherteks: [3725, 1761, 744, 2845, 3725, 1761, 744, 1825, 3390, 1761,
3924, 3736, 1825, 533, 3390, 4226, 1825, 744]
Plainteks: MATEMATIKA DISKRIT
```

Gambar 6. Hasil dari Algoritma RSA
Sumber: https://github.com/anellautari/Makalah-Matdis.git

3.2. Implementasi Algoritma Diffie-Hellman

Misalkan terdapat dua orang, yaitu Alice dan Bob, ingin melakukan pertukaran kunci dengan Diffie-Hellman. Mereka berdua sepakat untuk memilih bilangan prima publik $P = 71$ dan bilangan dasar $G = 43$ karena kedua bilangan tersebut telah memenuhi syarat $P > G$ dan 43 adalah akar primitif dari 71.

Langkah selanjutnya adalah memilih kunci privat dan menghitung kunci publik milik Alice. Misalkan Alice memilih kunci privat $X_A = 18$ karena angka tersebut memenuhi syarat $1 \leq X_A \leq p - 1$. Selanjutnya, Alice dapat menghitung kunci publik miliknya dengan perhitungan sebagai berikut:

$$Y_A = G^{X_A} \ mod\ P$$
$$Y_A = 43^{18} \ mod\ 71$$
$$= 16$$

Jadi, didapatkan kunci publik milik Alice (Y_A) adalah 16.

Cara yang sama dilakukan untuk memilih kunci privat (X_B) dan menghitung kunci publik (Y_B) milik Bob. Misalkan, ia memilih kunci privat $X_B = 25$ karena angka tersebut telah memenuhi syarat $1 \leq X_B \leq p - 1$. Perhitungan kunci publik Y_B adalah sebagai berikut:

$$Y_B = G^{X_B} \ mod\ P$$
$$Y_B = 43^{25} \ mod\ 71$$
$$Y_B = 45$$

Jadi, didapatkan kunci publik milik Bob (Y_B) adalah 45.

Setelah menghitung $X_A, Y_A, X_B,$ dan Y_B, Alice dan Bob saling bertukar kunci publik. Alice mendapatkan nilai Y_B dan Bob mendapatkan nilai Y_A. Setelah itu, mereka pun dapat menghitung kunci rahasia mereka dengan perhitungan sebagai berikut:

- Kunci rahasia Alice:

$$K_A = Y_B^{X_A} \ mod \ P$$
$$K_A = 45^{18} \ mod \ 71$$
$$K_A = 20$$

- Kunci rahasia Bob:

$$K_B = Y_A^{X_B} \ mod \ P$$
$$K_B = 16^{25} \ mod \ 71$$
$$K_B = 20$$

Kunci rahasia dari keduanya bernilai sama yaitu 20 yang artinya perhitungan mereka benar, yaitu $K = K_A = K_B = 20$. Jadi, mereka pun mendapatkan kunci rahasia bersama yang bernilai 20.

Implementasi Diffie-Hellman pada python sama seperti cara matematis yang telah kita lakukan. Kita dapat mengatur nilai P, G, X_A, dan X_B dalam sebuah variabel. Kemudian kita menghitung nilai Y_A dan Y_B dengan rumus yang sama seperti cara matematis. Lalu, kita dapat menghitung nilai kunci bersama dengan cara yang sama dan kita bandingkan hasil dari K_A dan K_B yang telah didapat. Jika kedua nilai tersebut berbeda, maka pertukaran kunci gagal karena kunci bersama tidak cocok. Jika kedua nilai tersebut sama, maka pertukaran kunci Diffie-Hellman berhasil dan kunci itulah yang dapat dipakai oleh Alice dan Bob untuk mengenkripsi dan dekripsi pesan.

```python
P = 71                      # Bilangan prima publik
G = 43                      # Bilangan dasar

X_A = 18                    # Kunci privat Alice
Y_A = pow(G, X_A, P)        # Kunci publik Alice

X_B = 25                    # Kunci privat Bob
Y_B = pow(G, X_B, P)        # Kunci publik Bob

# Pertukaran kunci publik
K_A = pow(Y_B, X_A, P)      # Kunci bersama (Alice)

K_B = pow(Y_A, X_B, P)      # Kunci bersama (Bob)
```

Gambar 7. Implementasi Diffie-Hellman pada python
Sumber: https://github.com/anellautari/Makalah-Matdis.git

Jika kita tidak ingin menentukan nilai dari P, G, X_A, dan X_B secara acak, kita dapat memodifikasi kode python tersebut untuk dapat memilih nilai P, G, X_A, dan X_B secara acak dengan memanfaatkan *library* pada python. Caranya adalah dengan membuat tiga fungsi, yang pertama adalah fungsi `is_prime(num)` yang berfungsi untuk mengecek apakah bilangan num adalah bilangan prima. Fungsi kedua adalah `generate_random_prime(start,end)` yaitu untuk menghasilkan bilangan prima secara acak dalam rentang `start` hingga `end`. Fungsi terakhir adalah `find_random_g(p)` yaitu untuk mencari bilangan g yang merupakan akar primitif dari p menggunakan syarat PBB$(g, p-1) = 1$ dengan memanfaatkan fungsi `gcd` dari *library* math pada python.

```python
import random
from math import gcd

def is_prime(num):
    if num < 2:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True

# Fungsi untuk mencari bilangan prima p secara acak
def generate_random_prime(start, end):
    while True:
        candidate = random.randint(start, end)
        if is_prime(candidate):
            return candidate

# Fungsi untuk mencari bilangan dasar g yang merupakan akar primitif dari p
def find_random_g(p):
    phi_p = p - 1
    while True:
        g = random.randint(2, p - 1)
        if gcd(g, phi_p) == 1:
            return g
```

Gambar 8. Modifikasi kode Diffie-Hellman pada Python
Sumber: https://github.com/anellautari/Makalah-Matdis.git

Setelah memodifikasi kode Diffie-Hellman, cara menggunakan ketiga fungsi tersebut untuk mencari nilai p adalah dengan memanggil fungsi generate_random_prime(start,end) dengan mengisi start dengan angka awal rentang bilangan prima dan end adalah angka akhir rentang bilangan prima. Untuk mendapatkan nilai g, kita hanya perlu memanggil fungsi find_random_g(p). Untuk mendapatkan kunci privat dari keduanya, kita gunakan *library* python yaitu fungsi random.randint(1, p-1) untuk menghasilkan bilangan bulat secara acak pada rentang 1 – (p-1). Berikut adalah implementasi secara lengkap.

```python
p = generate_random_prime(50, 100)
g = find_random_g(p)


X_A = random.randint(1, p-1)    # Kunci privat Alice
Y_A = pow(g, X_A, p)            # Kunci publik Alice

X_B = random.randint(1, p-1)    # Kunci privat Bob
Y_B = pow(g, X_B, p)            # Kunci publik Bob
```

Gambar 9. Mencari Nilai Kunci Privat Secara Acak
Sumber: https://github.com/anellautari/Makalah-Matdis.git

Hasil keluaran dari program tersebut akan menampilkan informasi nilai $P, G, X_A, X_B, Y_A, Y_B, K_A, K_B$, dan apakah kedua kunci rahasia bersama tersebtu cocok atau tidak. *Output*-nya adalah sebagai berikut.

```
Bilangan prima (P): 71
Bilangan dasar (G): 43
Kunci privat Alice: 18 , Kunci publik Alice: 16
Kunci privat Bob: 25 , Kunci publik Bob: 45
Kunci bersama (Alice): 20
Kunci bersama (Bob): 20
Kunci bersama berhasil dihitung dan cocok!
```

Gambar 10. *Output* dari Program Diffie-Hellman
Sumber: https://github.com/anellautari/Makalah-Matdis.git

3.3. Peran RSA dan Diffie-Hellman dalam Pengamanan Data

3.3.1. Kelebihan dan Kekurangan RSA dalam Pengamanan Data

Algoritma RSA memiliki keunggulan utama pada kemampuannya untuk melakukan enkripsi dan dekripsi secara asimetris, yaitu dengan menggunakan kunci publik untuk enkripsi dan kunci privat untuk dekripsi. Hal ini membuat RSA sangat cocok digunakan dalam situasi di mana keamanan kunci sangat penting. RSA juga sangat aman karena didasarkan pada kesulitan matematis dalam memfaktorkan bilangan besar.

Namun, RSA memiliki kelemahan dalam hal kecepatan. Proses enkripsi dan dekripsi dengan RSA memerlukan waktu yang cukup lama dibandingkan dengan algoritma lainnya, terutama ketika digunakan untuk data yang ukurannya besar. Oleh karena itu, RSA lebih efektif jika digunakan untuk mengenkripsi data kecil, seperti kunci simetris, daripada mengenkripsi data besar secara langsung.

3.3.2. Kelebihan dan Kekurangan Diffie-Hellman dalam Pengamanan Data

Diffie-Hellman memiliki keunggulan dalam pertukaran kunci rahasia melalui saluran komunikasi yang kurang aman. Algoritma ini memungkinkan dua pihak untuk berbagi kunci rahasia tanpa harus mengirimkan kunci privat, sehingga kunci tersebut tetap aman meskipun komunikasi mereka dapat dipantau oleh pihak ketiga.

Namun, Diffie-Hellman tidak bisa digunakan untuk mengenkripsi atau mendekripsi data secara langsung. Algoritma ini hanya menghasilkan kunci bersama, yang kemudian dapat digunakan oleh algoritma lain, seperti AES atau *Advanced Encryption Standard*, untuk melakukan proses enkripsi dan dekripsi. Selain itu, keamanan Diffie-Hellman sangat bergantung pada pemilihan parameter, seperti bilangan prima yang besar dan bilangan dasar yang harus memenuhi syarat.

3.3.3. Integrasi RSA dan Diffie-Hellman dalam Sistem Keamanan Data

RSA dan Diffie-Hellman sebenarnya dapat digunakan bersama untuk menciptakan sistem keamanan data yang lebih kuat. Diffie-Hellman digunakan untuk menghasilkan kunci rahasia bersama, sementara RSA dapat digunakan untuk mengenkripsi kunci tersebut agar lebih aman ketika dikirimkan melalui jaringan. Dengan cara ini, kedua algoritma dapat saling melengkapi untuk mengingkatkan keamanan sistem. Contoh kombinasi antara RSA dan Diffie-Hellman adalah sebagai berikut.

1. Alice dan Bob menggunakan Diffie-Hellman untuk membuat kunci rahasia bersama.
2. Kunci publik yang dimiliki Alice dan Bob dapat dienkripsi menggunakan RSA sebelum saling mengirimkan kunci publik tersebut.
3. Alice dan Bob melakukan dekripsi terhadap kunci publik yang diterimanya menggunakan RSA.
4. Setelah itu, kunci rahasia yang telah dihasilkan dapat digunakan untuk mengenkripsi dan dekripsi data dengan algoritma seperti AES.

Dengan kombinasi antara RSA dan Diffie-Hellman tersebut, sistem keamanan data dan informasi dapat menjadi semakin kuat. Kerja sama antara kedua algoritma dapat melindungi data secara menyeluruh. RSA memastikan keamanan kunci saat dikirimkan, sementara Diffie-Hellman membantu menciptakan kunci yang aman.

Pilihan menggunakan RSA dan Diffie-Hellman bergantung pada kebutuhan dan situasi, seperti:

- RSA lebih cocok digunakan ketika sistem memerlukan autentikasi di antara banyak pihak. Hal ini karena RSA menggunakan pasangan kunci publik dan privat yang unik untuk setiap pihak.
- Diffie-Hellman lebih cocok digunakan ketika hanya dua pihak yang perlu berbagi kunci secara aman tanpa perlu pengelolaan kunci publik yang rumit.

Dari analisis dan eksperimen yang telah dilakukan, dapat kita simpulkan bahwa algoritma RSA dan Diffie-Hellman memiliki peran penting masing-masing dalam membangun sistem keamanan data. Algoritma RSA unggul dalam memberikan keamanan selama proses pengiriman kunci, sementara algoritma Diffie-Hellman efektif dalam menciptakan kunci rahasia bersama untuk digunakan dalam komunikasi terenkripsi. Dengan memilih algoritma yang sesuai berdasarkan kebutuhan dan situasi tertentu, sistem keamanan dapat dirancang untuk memenuhi berbagai skenario, seperti autentikasi antar banyak pihak atau pertukaran kunci rahasia antara dua pihak.

IV. KESIMPULAN

Dalam era digital saat ini, kebutuhan akan sistem keamanan data yang kuat menjadi semakin penting. Algoritma RSA dan Diffie-Hellman memiliki cara yang berbeda untuk melindungi data dari penyadapan atau pencurian data. RSA digunakan untuk mengenkripsi dan mendekripsi data dengan kunci publik dan privat, sedangkan Diffie-Hellman memungkinkan pertukaran kunci rahasia antar dua pihak dengan aman, meskipun melalui saluran komunikasi yang rentan. Kombinasi keduanya dapat menciptakan sistem keamanan yang saling melengkapi, di mana RSA dapat mengenkripsi kunci simetris yang dihasilkan oleh Diffie-Hellman. Dengan penerapan yang tepat, kedua algoritma ini dapat melindungi data secara menyeluruh dan meningkatkan keamanan dalam berbagai aplikasi digital.

V. SARAN

Pengembangan lebih lanjut dapat difokuskan pada penerapan algoritma RSA dan Diffie-Hellman pada dataset yang lebih besar dan kompleks untuk mengevaluasi performa dari kedua algoritma tersebut. Selain itu, integrasi kedua algoritma dengan teknologi enkripsi modern seperti *Advanced Encryption Standard* (AES)

dapat dipertimbangkan untuk meningkatkan keamanan data pada tingkat aplikasi yang lebih luas. Edukasi mengenai penggunaan algoritma ini juga penting dilakukan agar lebih banyak pengguna memahami bagaimana cara melindungi data dan informasi mereka dengan teknologi yang tersedia.

VI. LAMPIRAN

Tautan repository:
https://github.com/anellautari/Makalah-Matdis.git

Tautan video youtube:
https://youtu.be/p9voBbkpnpc

VII. UCAPAN TERIMA KASIH

Penulis mengucapkan syukur kepada Tuhan Yang Maha Esa atas rahmat dan kemudahan yang diberikan sehingga makalah ini dapat diselesaikan. Penulis juga mengucapkan terima kasih kepada keluarga tercinta atas dukungan dan doanya, serta kepada Bapak Ir. Rila Mandala, M.Sc., Ph.D sebagai dosen mata kuliah Matematika Diskrit yang telah memberikan ilmu dan arahan selama perkuliahan. Penulis berharap makalah ini dapat memberikan manfaat bagi pembaca.

Referensi

[1] R. Munir, *Teori Bilangan Bagian 1*, 2024. [Daring]. Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/15-Teori-Bilangan-Bagian1-2024.pdf. [Diakses: 5-Jan-2025].

[2] Blogaritmaa, "Pembagi Bersama Terbesar dan Pembahasan Soal," 2016. [Daring]. Tersedia: https://blogaritmaa.blogspot.com/2016/04/pembagi-bersama-terbesar-dan-pembahasan.html. [Diakses: 5-Jan-2025].

[3] D. H. Ningsih, "Simulasi Aritmatika Modulo Pada Perhitungan Penanggalan Jawa," [Daring]. Tersedia: https://repository.unej.ac.id/handle/123456789/20247#:~:text=Kekongruenan%20modulo%20n%2C%20jika%20dua%20bilangan%20a,modulo%20n%20(%20ba%20%E2%89%A1%20(mod%20n)). [Diakses: 5-Jan-2025].

[4] Kaspersky, "What is Cryptography," [Daring]. Tersedia: https://www.kaspersky.com/resource-center/definitions/what-is-cryptography. [Diakses: 5-Jan-2025].

[5] KomputerKata, "Algoritma RSA," [Daring]. Tersedia: https://komputerkata.com/algoritma-rsa/. [Diakses: 5-Jan-2025].

[6] R. Munir, *Teori Bilangan Bagian 3*, 2024. [Daring]. Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/17-Teori-Bilangan-Bagian3-2024.pdf. [Diakses: 5-Jan-2025].

[7] R. Mulyawan, "Diffie-Hellman Key Exchange," [Daring]. Tersedia: https://rifqimulyawan.com/literasi/diffie-hellman-key-exchange/. [Diakses: 5-Jan-2025].

[8] R. Munir, *Algoritma Diffie-Hellman*, 2023. [Daring]. Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/22-Algoritma-Diffie-Hellman-2023.pdf. [Diakses: 5-Jan-2025].

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, 6 Januari 2025

Anella Utari Gunadi 13523078