

**ANALISIS PERBANDINGAN KECEPATAN WAKTU *ENKRIPSI*  
*DEKRIPSI* PADA ALGORITMA RSA DAN  
*VINCENT- SATHIYAMOORTHY***

**SKRIPSI**

**CHINDY BRIGITA BR SIHOTANG  
201401111**



**PROGRAM STUDI S-1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2024**

**ANALISIS PERBANDINGAN KECEPATAN WAKTU *ENKRIPSI*  
*DEKRIPSI* PADA ALGORITMA RSA DAN  
*VINCENT- SATHIYAMOORTHY***

**SKRIPSI**

**Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah  
Sarjana Ilmu Komputer (S.Kom)**

**CHINDY BRIGITA BR SIHOTANG  
201401111**



**PROGRAM STUDI S-1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2024**

**LEMBAR PERSETUJUAN**

Judul : ANALISIS PERBANDINGAN KECEPATAN WAKTU  
ENKRIPSI DEKRIPSI PADA ALGORITMA RSA DAN  
VINCENT - SATHIYAMOORTHY

Kategori : SKRIPSI

Nama : CHINDY BRIGITA BR SIHOTANG

Nomor Induk Mahasiswa : 201401111

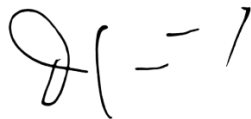
Progam Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA

Tanggal Sidang : 27 Mei 2024

Komisi Pembimbing :

Dosen Pembimbing II



Dian Rachmawati S.Si., M.Kom.  
NIP. 198307232009122004

Dosen Pembimbing I



Dr. Mohammad Andri Budiman  
S.T.,M.Comp.Sc., M.E.M.  
NIP. 197510082008011011

Diketahui/disetujui oleh  
Program Studi S Ilmu Komputer  
Ketua,



Dr. Amalia S.T., M.T.  
NIP. 197812212014042001

**LEMBAR PERNYATAAN****ANALISIS PERBANDINGAN KECEPATAN WAKTU ENKRIPSI DEKRIPSI  
PADA ALGORITMA RSA DAN VINCENT- SATHIYAMOORTHY****SKRIPSI**

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah dicantumkan sumbernya.

Medan, 27 Mei 2024

Chindy Brigita Br Sihotang  
201401111

## PENGHARGAAN

Segala puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas limpahan rahmat dan karunia-Nya, sehingga penulis dapat melakukan dan menyelesaikan penelitian ini dengan judul *"Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi Pada Algoritma RSA dan Vincent-Sathiyamoorthy"* sebagai persyaratan untuk meraih gelar Sarjana (S-1) Ilmu Komputer di Universitas Sumatera Utara. Penelitian ini tidak bisa dipisahkan dari dukungan berbagai pihak, dan penulis ingin mengucapkan terima kasih kepada mereka yang telah memberikan nasihat, motivasi, dan hiburan selama proses penelitian, yaitu:

1. Bapak Dr. Muryanto Amin, S.Sos., M.Si yang menjabat sebagai Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvia Lydia, B. Sc., M. Sc., yang menjabat sebagai Dekan Fasilkom-TI di Universitas Sumatera Utara.
3. Ibu Dr. Amalia, S.T., M . T., yang menjabat sebagai Ketua Program Studi S- 1 Ilmu Komputer di Universitas Sumatera Utara.
4. Ibu Dr. Elviawaty Muisa Zamzami S.T., M.T., MM yang berperan sebagai dosen pembimbing akademik yang sudah memberikan dukungan dan arahan ketika melakukan konsultasi akademik.
5. Bapak Dr. Mohammad Andri Budiman S.T., M.Comp.Sc., M.E.M. dan Ibu Dian Rachmawati S.Si., M.Kom. yang berperan sebagai dosen pembimbing skripsi penulis yang dengan kesabaran, kebijaksanaan, dan arahan yang diberikan dalam membantu penulis melewati setiap tahap penelitian dengan baik.
6. Seluruh dosen pengajar prodi Ilmu Komputer dan Teknologi Informasi di Fakultas Ilmu Komputer dan Teknologi Informasi di Universitas Sumatera Utara.
7. Bapak Timbul Marudut Sihotang dan Ibu Pesta Natalina Simbolon, sebagai orang tua dari penulis, yang sudah memberikan motivasi dan cinta kepada penulis dalam menempuh pendidikan di Fasilkom-TI USU terkhususnya dalam pengerjaan tugas akhir ini.
8. Putri Panduwinata Sihotang dan Dani-Danianto Sihotang selaku adik dari penulis yang sudah mengingatkan dan memberikan dorongan penuh kepada penulis untuk tidak

menunda-nunda menyelesaikan tugas akhir ini.

9. Petrus Marcelino H. Tampubolon, Yogi Piranda Limbongn dan Venerio Uvandy selaku teman penulis yang telah membantu dalam memperbaiki dan merapikan tugas akhir ini.
10. Audrey Malika Sitepu, Wina Octaria Saragih, Hasnah Dewi, dan Rizka Aulia Putri selaku teman-teman dekat penulis yang memberi motivasi, semangat, dan menghibur penulis disaat penulis merasa sedih, serta seluruh teman KOM C Stambuk 2022 yang telah bersama penulis selama menempuh pendidikan kurang lebih selama 4 tahun.
11. Sri Nova Theresia Sihotang dan Cindy Jose Maharani Ginting selaku sahabat penulis yang sudah memberikan penghiburan dan dukungan moral kepada penulis dalam penyelesain tugas akhir ini.
12. Semua yang turut membantu dalam menyelesaikan tugas akhir ini yang tidak dapat disebutkan satu per satu oleh penulis.
13. Terakhir, terima kasih untuk diri saya sendiri Chindy Brigita Br Sihotang atas dedikasi, ketekunan, dan ketabahan dalam menyelesaikan tugas akhir ini. Perjalanan ini penuh dengan rintangan, namun dengan tekad yang kuat, saya berhasil mengatasi setiap tantangan. Semoga keberhasilan ini menjadi motivasi untuk terus berkembang serta mencapai prestasi yang lebih baik dan tinggi di masa depan.

Penulis mengakui bahwa tugas akhir ini masih mempunyai kelemahan. Oleh karena itu, penulis sangat mengapresiasi masukan dan saran yang dapat membantu menyempurnakan dan menyelesaikan karya ini.

Medan, 27 Mei 2024

Chindy Brigita Br Sihotang  
201401111

## ABSTRAK

Keamanan mempunyai peran yang sangat penting dalam bidang ilmu komputer dan teknik, terutama di dalam menangani isu-isu kritis terkait dengan keamanan data. Aspek keamanan ini melibatkan berbagai teknologi dan metode, di mana salah satu teknologi yang paling terjamin adalah kriptografi. Kriptografi merupakan suatu proses di mana teks biasa diubah menjadi teks sandi untuk mentransfer data kepada pengguna yang memiliki hak akses yang sah. Pemilihan algoritma yang sesuai dengan aplikasi tertentu selalu merupakan tantangan karena mempertimbangkan berbagai faktor seperti latensi, ukuran kunci, dan permasalahan keamanan. Penelitian ini bertujuan untuk melakukan analisis perbandingan kecepatan waktu enkripsi dan dekripsi kedua algoritma kriptografi yang akan diteliti, yaitu RSA dan Vincent-Sathiyamoorthy. RSA, sebagai algoritma kriptografi kunci publik yang sudah mapan, akan dibandingkan dengan algoritma Vincent-Sathiyamoorthy yang muncul sebagai alternatif baru. Metodologi penelitian ini melibatkan implementasi kedua algoritma pada lingkungan komputasi dan pengukuran waktu yang dibutuhkan untuk proses pengenkripsian dan pendekripsian. Hasil analisis memperlihatkan perbandingan performa waktu antara RSA dan Vincent-Sathiyamoorthy, memberikan wawasan yang berharga terkait kecepatan eksekusi keduanya.

**Kata Kunci :** Kriptografi, RSA, Vincent-Sathiyamoorthy, Enkripsi, Dekripsi

## **COMPARATIVE ANALYSIS OF ENCRYPTION DECRYPTION TIME SPEED IN THE RSA AND VINCENT-SATHIYAMOORTHY ALGORITHM**

### **ABSTRACT**

Security plays a vital role in the field of computer science and engineering, especially in handling critical issues related to data security. This security aspect involves various technologies and methods, with one of the most reliable technologies being cryptography. Cryptography is a process where plaintext is transformed into ciphertext to transfer data to users with legitimate access. Selecting the appropriate algorithm for a particular application is always a challenge, considering various factors such as latency, key size, and security issues. This research aims to conduct a comparative analysis of the encryption and decryption speed of both cryptography algorithms under study, RSA and Vincent-Sathiyamoorthy. RSA, as a well-established public key cryptography algorithm, will be compared with the Vincent-Sathiyamoorthy algorithm, which emerges as a new alternative. The research methodology involves implementing both algorithms in a computational environment and measuring the time required for the encryption and decryption processes. The analysis results show a comparison of time performance between RSA and Vincent-Sathiyamoorthy, providing valuable insights into their execution speeds.

**Keywords :** Cryptography, RSA, Vincent-Sathiyamoorthy, Encryption, Decryption



## DAFTAR ISI

<b>LEMBAR PERSETUJUAN .....</b>	<b>ii</b>
<b>LEMBAR PERNYATAAN .....</b>	<b>iii</b>
<b>PENGHARGAAN.....</b>	<b>iv</b>
<b>ABSTRAK .....</b>	<b>vi</b>
<b>ABSTRACT.....</b>	<b>vii</b>
<b>DAFTAR ISI.....</b>	<b>viii</b>
<b>DAFTAR GAMBAR.....</b>	<b>xi</b>
<b>DAFTAR TABEL .....</b>	<b>xii</b>
<b>BAB I PENDAHULUAN.....</b>	<b>1</b>
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah.....	3
1.3 Batasan Masalah .....	3
1.4 Tujuan Penelitian .....	4
1.5 Manfaat Penelitian .....	4
1.6 Metode Penelitian .....	4
1.7 Sistematika Penulisan .....	5
<b>BAB 2 LANDASAN TEORI .....</b>	<b>6</b>
2.1 Teori Bilangan.....	6
2.1.1 Bilangan Prima.....	6
2.1.2 <i>Greatest Common Divisor (GCD)</i> .....	8
2.1.3 Relatif Prima .....	9
2.1.4 <i>Least Common Multiple ( LCM )</i> .....	9
2.1.5 Aritmatika Modulo.....	10
2.1.6 Modulo Eksponensial ( <i>Modular Exponentiation</i> ).....	11
2.1.7 <i>Invers</i> Modulo .....	12
2.1.8 <i>Totient Euler (<math>\phi</math>)</i> .....	13
2.2 Kriptografi .....	14
2.3 Tujuan Kriptografi .....	15
2.4 Jenis - Jenis Kriptografi .....	15

2.5	Algoritma Rsa .....	17
2.6	Algoritma Vincent-Sathiyamoorthy .....	19
2.7	Kompleksitas Algoritma .....	20
2.8	Peneletian Relevan.....	21
<b>BAB 3</b>	<b>ANALISIS DAN PERANCANGAN SISTEM .....</b>	<b>23</b>
3.1	Analisis Sistem .....	23
3.1.1	Analisis Masalah .....	23
3.1.2	Analisis Kebutuhan .....	24
3.2	Diagram Umum Sistem .....	25
3.3	Perancangan Sistem .....	26
3.3.1	<i>Use Case Diagram</i> .....	26
3.3.2	<i>Activity Diagram</i> .....	27
3.3.3	<i>Sequence diagram</i> .....	29
3.4	Flowchart .....	32
3.4.1	<i>Flowchart</i> Pembangkit Bilangan Prima .....	32
3.4.2	<i>Flowchart</i> Pembangkit Kunci Algoritma Vincent- Sathiyamoorthy .	33
3.4.3	<i>Flowchart</i> Enkripsi Algoritma Vincent-Sathiyamoorthy .....	34
3.4.4	<i>Flowchart</i> Dekripsi Algoritma Vincent-Sathiyamoorthy .....	35
3.4.5	<i>Flowchart</i> Pembentukan Kunci Algoritma RSA .....	36
3.4.6	<i>Flowchart</i> Proses Enkripsi Algoritma RSA.....	37
3.4.7	<i>Flowchart</i> Proses Dekripsi Algoritma RSA .....	38
3.5	Perancangan <i>User Interface</i> .....	39
3.5.1	Perancangan halaman <i>generate</i> bilangan prima.....	39
3.5.2	Perancangan halaman algoritma RSA .....	40
3.5.3	Perancangan halaman algoritma Vincent-Sathiyamoorthy .....	41
<b>BAB 4</b>	<b>IMPLEMENTASI DAN PENGUJIAN.....</b>	<b>43</b>
4.1	Implementasi Perhitungan .....	43
4.1.1	Algoritma Vincent-Sathiyamoorthy .....	43
4.1.2	Algoritma RSA .....	46
4.2	Implementasi Sistem.....	48
4.2.1	<i>Generate</i> Bilangan Prima.....	48
4.2.2	Algoritma Vincent-Sathiyamoorthy.....	49
4.2.3	Algoritma RSA .....	50

4.3	Pengujian Sistem.....	51
4.3.1	Perbandingan Waktu Pembuatan Kunci Berdasarkan Nilai $p$ dan $q$ yang sama .....	51
4.3.2	Perbandingan Rata-rata Waktu Enkripsi Dekripsi Berdasarkan Persamaan Nilai $p$ dan $q$ yang sama dan Perbedaan Panjang Plaintext .....	53
4.4	Perhitungan Kompleksitas Algoritma.....	64
4.4.1	Algoritma Vincent-Sathiyamoorthy.....	64
4.4.2	Algoritma RSA .....	66
<b>BAB 5</b>	<b>KESIMPULAN DAN SARAN .....</b>	<b>68</b>
5.1	Kesimpulan .....	68
5.2	Saran .....	68
<b>DAFTAR PUSTAKA .....</b>		<b>70</b>

## DAFTAR GAMBAR

Gambar 2. 1 Kriptografi Simetris .....	16
Gambar 2. 2 Kriptografi Asimetris .....	16
Gambar 3. 1 Fishbone Diagram Enkripsi Dekripsi .....	24
Gambar 3. 2 Diagram Umum Sistem Algoritma Vincent-Sathiyamoorthy .....	25
Gambar 3. 3 Diagram Umum Sistem Algoritma RSA.....	26
Gambar 3. 4 Use Case Diagram Enkripsi Dekripsi.....	27
Gambar 3. 5 Activity Diagram Pembuatan Kunci .....	28
Gambar 3. 6 Activity Diagram Enkripsi .....	28
Gambar 3. 7 Activity Diagram Dekripsi.....	29
Gambar 3. 8 Sequence Diagram Dekripsi Algoritma Vincent-Sathiyamoorthy .....	30
Gambar 3. 9 Sequence Diagram Enkripsi Algoritma Vincent-Sathiyamoorthy .....	30
Gambar 3. 10 Sequence Diagram Dekripsi Algoritma RSA.....	31
Gambar 3. 11 Sequence Diagram Enkripsi Algoritma RSA.....	31
Gambar 3. 12 Flowchart Pembangkit Bilangan Prima.....	32
Gambar 3. 13 Flowchart Pembangkit Kunci Algoritma Vincent-Sathiyamoorthy .....	33
Gambar 3. 14 Flowchart Enkripsi Algoritma Vincent-Sathiyamoorthy .....	34
Gambar 3. 15 Flowchart Dekripsi Algoritma Vincent-Sathiyamoorthy .....	35
Gambar 3. 16 Flowchart Pembentukan Kunci Algoritma RSA .....	36
Gambar 3. 17 Flowchart Proses Enkripsi Algoritma RSA .....	37
Gambar 3. 18 Flowchart Proses Dekripsi Algoritma RSA .....	38
Gambar 3. 19 Perancangan halaman generate bilangan prima .....	39
Gambar 3. 20 Perancangan halaman algoritma RSA.....	40
Gambar 3. 21 Perancangan halaman algoritma RSA.....	41
Gambar 4. 1 Tampilan Sistem Generate Bilangan Prima .....	48
Gambar 4. 2 Tampilan Sistem Algoritma Vincent-Sathiyamoorthy.....	49
Gambar 4. 3 Tampilan Sistem Algoritma RSA .....	50
Gambar 4. 4 Grafik waktu pembuatan kunci algoritma VS dan RSA .....	52
Gambar 4. 5 Grafik Perbandingan Waktu Enkripsi Algoritma VS dan RSA .....	61
Gambar 4. 6 Grafik Perbandingan Waktu Dekripsi Algoritma VS dan RSA .....	63

## DAFTAR TABEL

Tabel 4. 1 Waktu Pembuatan Kunci Algoritma VS dan RSA .....	52
Tabel 4. 2 Waktu Enkripsi dan Dekripsi Algoritma Vincent-Sathiyamoorthy .....	53
Tabel 4. 3 Waktu Enkripsi dan Dekripsi Algoritma RSA.....	54
Tabel 4. 4 Waktu Enkripsi dan Dekripsi Algoritma Vincent-Sathiyamoorthy .....	55
Tabel 4. 5 Waktu Enkripsi dan Dekripsi Algoritma RSA.....	57
Tabel 4. 6 Waktu Enkripsi dan Dekripsi Algoritma Vincent-Sathiyamoorthy .....	58
Tabel 4. 7 Waktu Enkripsi dan Dekripsi Algoritma RSA.....	59
Tabel 4. 8 Waktu Enkripsi Algoritma Vincent-Sathiyamoorthy dan RSA.....	60
Tabel 4. 9 Waktu Enkripsi Algoritma Vincent-Sathiyamoorthy dan RSA.....	62
Tabel 4. 10 Kompleksitas Waktu Proses Pembuatan Kunci Algoritma Vincent-Sathiyamoorthy .....	64
Tabel 4. 11 Kompleksitas Waktu Proses Enkripsi Algoritma Vincent-Sathiyamoorthy .....	65
Tabel 4. 12 Kompleksitas Waktu Proses Dekripsi Algoritma Vincent-Sathiyamoorthy .....	65
Tabel 4. 13 Kompleksitas Waktu Pembuatan Kunci Algoritma RSA .....	66
Tabel 4. 14 Kompleksitas Waktu Proses Enkripsi Algoritma RSA.....	67
Tabel 4. 15 Kompleksitas Waktu Proses Dekripsi Algoritma RSA .....	67

# **BAB I**

## **PENDAHULUAN**

Bab ini membahas tentang pendahuluan dari penelitian dengan judul "Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi antara Algoritma RSA dan Algoritma Vincent-Sathiyamoorthy". Pada bagian ini berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian, dan sistematika penulisan.

### **1.1 Latar Belakang**

Kemajuan teknologi memiliki peran yang sangat penting dalam mendukung kehidupan manusia, terutama dalam hal komunikasi. Komunikasi melalui jaringan memegang peranan penting di era digital, di mana seseorang dapat melakukan berbagai transaksi dan pertukaran data dengan praktis dan cepat (Aulia, *et al.* 2023). Namun, dari perkembangan tersebut dapat menimbulkan resiko besar terhadap keamanan informasi, mulai dari penyalahgunaan akses yang tidak berwenang, modifikasi atau perubahan informasi, perusakan hingga tindakan pencurian yang dilakukan oleh pihak yang tidak memiliki tanggung jawab (Yel & Nasution, 2022). Oleh karena itu, dibutuhkan suatu cara yang mampu menyelesaikan kejahatan yang terjadi pada saat ini. Metode yang dapat dilakukan adalah kriptografi (Merliana, 2020). Kriptografi merupakan disiplin ilmu atau seni yang bertujuan untuk memastikan pesan yang dikirim oleh pengirim dapat diterima secara aman oleh penerima (Putri, *et al.* 2019). Prinsip utama dari kriptografi yaitu *confidentiality* (kerahasiaan), *data integrity* (integritas data), *authentication* (autentikasi), dan *non-repudiation* (ketiadaan penyangkalan) (Hermawati, *et al.* 2023). Keamanan data harus mematuhi prinsip-prinsip ini untuk memastikan kepercayaan individu bahwa data dan informasi dikirimkan sebagaimana mestinya. Prosedur untuk melindungi data dalam bidang kriptografi dikenal sebagai algoritma kriptografi. Algoritma yang digunakan dalam kriptografi dapat dibagi

menjadi dua jenis yaitu kriptografi simetris dan kriptografi asimetris (Purba, dkk. 2020).

Kriptografi kunci simetris disebut kriptografi kunci rahasia. Dalam kriptografi simetris, pengirim serta penerima memakai kunci yang serupa untuk melaksanakan proses pengenkripsian serta pendekripsian. Contoh algoritma kriptografi simetris antara lain *AES*, *DES*, *3DES*, dan *Blowfish*. Sedangkan kriptografi asimetris dikenal sebagai kriptografi kunci publik. Dalam kriptografi asimetris, pengirim dan penerima memakai kunci yang berbeda. Pengirim memanfaatkan kunci publik untuk melakukan pengenkripsian pesan, dan penerima memakai kunci privat untuk melakukan pendekripsian pesan. Contoh algoritma kriptografi asimetris antara lain *RSA*, *Diffie-Hellman*, *ECC*, dan sebagainya. (Anshori, *et al.* 2019). Algoritma *RSA* (*Rivest-Shamir-Adleman*) ditemukan pada tahun 1978 oleh para profesor yaitu Leonard Adleman, Adi Shamir, dan Ron Rivest di laboratorium Massachusetts (MIT). Algoritma ini mempunyai 2 kunci yang nilainya berbeda-beda, satu adalah kunci publik yang dapat diakses semua orang, dan dipakai dalam proses enkripsi. Kunci lainnya adalah kunci pribadi, yang digunakan untuk mendekripsi pesan terenkripsi (Pamungkas & Irawan, 2023). Kelebihan algoritma *RSA* adalah dapat memberikan enkripsi yang kuat sehingga menyebabkan pesan asli sulit diprediksi dan pihak yang tidak berkepentingan tidak dapat mengaksesnya.

Algoritma *RSA* sudah mengalami banyak modifikasi yang ditemukan oleh para profesor lainnya. Menurut penelitian (Dhakar, *et al.* 2012) modifikasi dilakukan pada *RSA* tradisional yang dimana kompleksitas diterapkan lebih lanjut dalam proses enkripsi dan dekripsi dengan operasi yang kompleks untuk menyediakan keamanan yang lebih baik. Namun, hasil modifikasi memakan waktu eksekusi lebih lama daripada *RSA* tradisional. Selain itu, (Li, *et al.* 2010) mengembangkan algoritma menjadi *Encrypt Assistant Multi Prime RSA* (*EAMRSA*). Tujuan dari algoritma ini adalah untuk mempercepat proses dekripsi.

Pada tahun 2016, P.M. Durai Raj Vincent dan E. Sathiyamoorthy juga memperkenalkan pendekatan baru yang memiliki kunci publik kriptosistem yang merupakan pengembangan dari algoritma *RSA*. Algoritma ini dikenal dengan nama algoritma *Vincent- Sathiyamoorthy*. Algoritma ini memiliki kemiripan dengan *RSA* yaitu memiliki kunci publik yang sama, namun dapat melakukan pengurangan waktu pada saat melakukan enkripsi dan dekripsi.

Pada penelitian yang sudah dilaksanakan oleh (Hidayat, *et al.* 2020) yang

berjudul *“Implementation of RSA and RSA-CRT Algorithms for Compariosn of Encryption and Decryption Time in Android-bases Instant Message Applications”* menyampaikan bahwa perbandingan waktu pada proses enkripsi yang terjadi antara algoritma RSA dan RSA-CRT tidak menunjukkan perbedaan yang signifikan, meskipun keduanya menggunakan modulus yang berbeda. Namun, pada proses dekripsi, penggunaan metode CRT pada RSA terbukti sangat efektif dalam mempercepat dekripsi pesan, bahkan ketika modulus yang digunakan berbeda. Pada RSA-CRT 1024 bit, jika dihitung kecepatan dekripsi pesan akan meningkat sekitar 2,6 kali lebih cepat dibandingkan RSA 1024 bit.

Pada penelitian yang telah dilaksanakan oleh (Marqas, *et al.* 2020) yang berjudul *“Comparing Symmetric and Asymmetric cryptography in message encryption and decryption by using AES and RSA algorithms”* menyampaikan bahwa AES memiliki waktu eksekusi yang lebih baik dibandingkan RSA. Selain itu, AES memiliki 3 kunci yang berbeda sedangkan RSA hanya memiliki 2 kunci yang berbeda yang menunjukkan bahwa AES lebih fleksibel dan berkinerja lebih unggul dan cepat jika dibandingkan dengan RSA.

Berdasarkan latar belakang yang sudah penulis uraikan di atas, maka dilaksanakan penelitian dengan judul “Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi Pada Algoritma RSA dan Algoritma Vincent-Sathiyamoorthy”.

## **1.2 Rumusan Masalah**

Dari latar belakang masalah yang sudah penulis uraikan sebelumnya, maka fokus penelitian difokuskan pada bagaimana perbandingan waktu kecepatan yang dihasilkan pada saat melakukan enkripsi dekripsi pada algoritma RSA dan Vincent-Sathiyamoorthy serta mengevaluasi kinerja efisien dan efektifitas kedua algoritma tersebut.

## **1.3 Batasan Masalah**

Dalam menjalankan penelitian ini, terdapat empat batasan permasalahan yang menjadi fokus penelitian yang akan diselidiki, yaitu sebagai berikut:

- a. Perbandingan waktu dihitung dari keberhasilan dilakukannya enkripsi dekripsi pada kedua algoritma tersebut.



- b. Penggunaan kode teks dalam penelitian ini mengacu pada tabel kode *ASCII* yang terdiri dari 8 bit.
- c. Uji bilangan prima menggunakan algoritma Lehman.
- d. Bahasa pemrograman yang diterapkan ialah *Python*.

#### **1.4 Tujuan Penelitian**

Tujuan dari penelitian ini adalah untuk membandingkan durasi yang diperlukan dalam proses pengenkripsian dan pendekripsian antara algoritma RSA dengan algoritma Vincent-Sathiyamoorthy, serta menilai efisiensi dan efektivitas kinerja kedua algoritma tersebut.

#### **1.5 Manfaat Penelitian**

Penelitian ini bermanfaat untuk memberikan pemahaman yang jelas tentang perbedaan dalam kecepatan dan kompleksitas antara algoritma RSA dan Vincent-Sathiyamoorthy sehingga dapat menghemat sumber daya komputasi yang berharga serta memudahkan pemilihan algoritma enkripsi-dekripsi yang lebih optimal.

#### **1.6 Metode Penelitian**

Metode penelitian yang digunakan dalam melaksanakan penelitian ini meliputi:

##### **1. Studi Literatur**

Pada bagian ini, data dikumpulkan dengan melakukan pencarian referensi dari beragam sumber *otoritatif* seperti melaksanakan penelusuran literatur pada artikel ilmiah, buku dan artikel penelitian lain yang diterbitkan pada jurnal akademik terkait Vincent-Sathiyamoorthy dan algoritma RSA.

##### **2. Analisis dan Perancangan Sistem**

Pada bagian ini, dilaksanakan evaluasi kebutuhan penelitian serta perancangan dalam bentuk diagram *Ishikawa*, diagram kasus pengguna, diagram urutan, serta diagram alir.

##### **3. Implementasi Sistem**

Pada bagian ini, sistem dijalankan dengan menerapkan bahasa pemrograman *Python* sesuai dengan rancangan yang telah disusun.

#### 4. Pengujian Sistem

Pada bagian ini, sistem dijalankan memakai bahasa pemrograman *Python* sesuai dengan *flowchart* yang telah disusun sebelumnya.

#### 5. Dokumentasi

Pada bagian ini, hasil analisis dari rancangan sistem disusun dalam format skripsi.

### 1.7 Sistematika Penulisan

Berikut adalah susunan dalam pembuatan skripsi yang terbagi menjadi beberapa bagian yang utama seperti yang dijabarkan di bawah ini :

#### **BAB 1 : PENDAHULUAN**

Bagian ini membahas tentang pendahuluan dari penelitian dengan judul "Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi antara Algoritma RSA dan Algoritma Vincent-Sathiyamoorthy".

#### **BAB 2 : LANDASAN TEORI**

Bagian ini membahas topik-topik seperti teori bilangan, konsep kriptografi, tujuan kriptografi, jenis-jenis kriptografi, algoritma RSA, algoritma Vincent- Sathiyamoorthy, analisis kompleksitas algoritma, serta penelitian sebelumnya yang relevan.

#### **BAB 3 : ANALISIS DAN PERANCANGAN**

Bagian ini mengulas analisis terkait dengan permasalahan yang diteliti serta merancang kerangka sistem yang direncanakan untuk menyelesaikan masalah tersebut.

#### **BAB 4 : IMPLEMENTASI DAN PENGUJIAN**

Bagian ini mengulas langkah-langkah dalam pembuatan sistem sebagai penerapan dari analisis dan perancangan sebelumnya, dan membahas tentang pengujian sistem serta hasil analisis dari hasil uji sistem.

#### **BAB 5 : KESIMPULAN DAN SARAN**

Bagian akhir ini memuat rangkuman keseluruhan dari hasil penelitian yang sudah dilaksanakan, dilengkapi dengan saran berdasarkan temuan akhir, yang diinginkan dapat memberikan kontribusi pada kemajuan perkembangan penelitian di masa depan.

## **BAB 2**

### **LANDASAN TEORI**

Bab ini membahas mengenai topik-topik yang membangun judul penelitian. Topik-topik yang dibahas seperti teori bilangan, konsep kriptografi, tujuan kriptografi, jenis-jenis kriptografi, algoritma RSA, algoritma Vincent- Sathiyamoorthy, analisis kompleksitas algoritma, serta penelitian sebelumnya yang relevan.

#### **2.1 Teori Bilangan**

Teori bilangan atau yang bisa disebut juga sebagai *number theory* merupakan cabang dalam ilmu matematika dasar yang memfokuskan untuk mempelajari bilangan bulat (*integer*) atau fungsi yang menghasilkan nilai bilangan bulat (Akbar, 2021). Teori bilangan adalah dasar yang penting saat memahami kriptografi. Adapun bilangan yang diinginkan di sini adalah bilangan bulat positif, yang di mana tidak memiliki nilai desimal. Contohnya adalah -3, -10, 5, 9, dan 0. Sebaliknya, bilangan *riil* adalah bilangan yang mempunyai titik desimal, seperti 0.15, 3.79, 108.25. Dalam penulisan skripsi ini, akan dipelajari konsep-konsep dasar yang menjadi dasar bagi algoritma RSA.

##### *2.1.1 Bilangan Prima*

Bilangan prima ialah bilangan bulat yang tidak memiliki pembagi lain selain nilai positif dan negatif dari dirinya sendiri, serta angka 1 tanpa menyisakan sisa pembagian. Ciri – ciri bilangan prima :

- Bilangan prima hanya mempunyai 2 faktor pembagi yakni bilangan dirinya sendiri dan 1.
- Bilangan prima tidak dapat ditulis sebagai perkalian dari bilangan lain.
- Bilangan prima selalu memiliki nilai yang lebih besar dari 1.
- Bilangan genap yang memiliki nilai lebih dari 2 dapat dinyatakan sebagai hasil penjumlahan dua bilangan prima.

Adapun aturan untuk mencari bilangan prima antara 1- 100 adalah sebagai berikut :

- 1 bukanlah bilangan prima dan komposit.
- Bilangan 2 adalah sebuah bilangan prima, namun tidak semua bilangan yang dapat dibagi habis oleh 2 juga merupakan bilangan prima.
- Bilangan 3 adalah sebuah bilangan prima, namun tidak semua bilangan yang dapat dibagi habis oleh 3 juga merupakan bilangan prima.
- Bilangan 5 adalah sebuah bilangan prima, namun tidak semua bilangan yang dapat dibagi habis oleh 5 juga merupakan bilangan prima.
- Bilangan 7 adalah sebuah bilangan prima, namun tidak semua bilangan yang dapat dibagi habis oleh 7 juga merupakan bilangan prima.

Cara untuk memastikan apakah sebuah bilangan tertentu ialah bilangan prima atau tidak, dapat menggunakan metode algoritma *Lehman*. Algoritma *Lehman* merupakan salah satu metode yang efisien untuk melakukan pengujian keprimaan suatu bilangan. Berikut adalah syarat-syarat dalam menggunakan Algoritma *Lehman* untuk membangkitkan bilangan prima:

1. Bangkitkan bilangan  $a$  sesuai ketentuan bahwa  $1 < a < p$ , di mana  $p$  merupakan bilangan prima.
2. Hitunglah nilai  $L$  (*Legendre*) dengan  $L = a^{(p-1)/2} \bmod p$ .
3. Jika nilai  $L$  tidak sama dengan 1 atau selisih  $L$  dengan  $p$  bukan sama dengan -1, jadi  $p$  bukanlah sebuah bilangan prima.
4. Jika nilai  $L$  sama dengan 1 atau selisih  $L$  dengan  $p$  sama dengan -1, jadi *probabilitas* bahwa  $p$  bukan bilangan prima lebih besar dari 50%.
5. Lakukan pengujian sejumlah  $t$  dengan variasi nilai  $a$  yang tidak sama dengan satu sama lain. Jika hasil pembagian  $L = 1$  atau  $L = -1$ , namun tidak konsisten dengan nilai 1 setiap kali, maka *probabilitas*  $p$  sebagai bilangan prima memiliki tingkat kesalahan yang tidak lebih dari  $1/2^t$

Contoh :

Lakukan pengujian keprimaan pada bilangan  $p = 11$  menggunakan algoritma *Lehman*.

Hitunglah  $L = a^{(11-1)/2} \bmod 11$  dengan menguji nilai  $a$ , di mana  $1 \leq a \leq 11$ .

$a = 1$	$1^{(11-1)/2} \bmod 11 \equiv 1$
$a = 2$	$2^{(11-1)/2} \bmod 11 \equiv / 1$
$a = 3$	$3^{(11-1)/2} \bmod 11 \equiv 1$
$a = 4$	$4^{(11-1)/2} \bmod 11 \equiv 1$
$a = 5$	$5^{(11-1)/2} \bmod 11 \equiv 1$
$a = 6$	$6^{(11-1)/2} \bmod 11 \equiv / 1$
$a = 7$	$7^{(11-1)/2} \bmod 11 \equiv / 1$
$a = 8$	$8^{(11-1)/2} \bmod 11 \equiv / 1$
$a = 9$	$9^{(11-1)/2} \bmod 11 \equiv 1$
$a = 10$	$10^{(11-1)/2} \bmod 11 \equiv / 1$
$a = 11$	$11^{(11-1)/2} \bmod 11 \equiv / 1$

Dengan demikian, bilangan 11 dapat disebut sebagai bilangan prima karena memenuhi persyaratan yaitu  $a^{(p-1)/2} \equiv 1$  sebanyak 5 kali, yang sama dengan setengah dari jumlah pengujian yaitu  $11/2 = 5$ . Tingkat kesalahan adalah  $1/2^{11} = 0,0488\%$ , sehingga tingkat kebenarannya mencapai 99,95%.

### 2.1.2 Greatest Common Divisor (GCD)

*Greatest Common Divisor (GCD)*, atau dikenal sebagai Faktor Persekutuan Terbesar (FPB) merupakan bilangan bulat yang paling besar yang dapat membagi kedua bilangan secara penuh tanpa menyisakan sisa. Ada berbagai metode untuk menghitung FPB dua angka, dan salah satunya adalah melalui faktorisasi prima dasar. Metode ini melibatkan pemfaktoran bilangan-bilangan ke dalam pangkat bilangan prima dan penemuan faktor persekutuan dari bilangan prima tersebut. Meski begitu, perhitungan

ini menjadi kompleks serta membutuhkan waktu signifikan. Oleh karena itu, terdapat empat algoritma yang telah disederhanakan untuk membantu dalam menghitung GCD. Keempat algoritma tersebut mencakup Algoritma *Euclidean*, algoritma *Biner GCD*, algoritma *Dijkstra*, dan algoritma *Lehmer* (Siddhartha, *et al.* 2020). Dalam skripsi ini, akan membahas mengenai algoritma *Euclidean*.

Algoritma *Euclidean* adalah algoritma sederhana yang dapat menghitung GCD dengan cepat (Suroso, 2018). Berikut adalah langkah-langkah yang digunakan dalam menghitung GCD 2 bilangan:

1. Jika nilai  $b = 0$ , maka nilai  $a$  merupakan GCD ( $a, b$ ); proses dihentikan. Jika tidak (artinya  $b \neq 0$ ), maka lanjutkan ke tahap selanjutnya.
2. Bagilah  $a$  dengan  $b$  dan misalkan  $r$  merupakan hasil sisa pembagian dari dua bilangan. Jika nilai  $r = 0$ , maka proses dihentikan.
3. Jika nilai  $r \neq 0$ , maka ubah nilai  $a$  dengan nilai  $b$  serta ubah nilai  $b$  dengan nilai  $r$ , kemudian kembali ke tahap pertama.

Contoh :

Berapakah GCD (54,12) ?

$$54 \bmod 12 = 6$$

$$12 \bmod 6 = 0$$

Jadi, GCD dari (54,12) adalah 6.

### 2.1.3 Relatif Prima

Dua bilangan bulat positif yakni  $a$  dan  $b$  disebut relatif prima jika dan hanya jika

$$\text{GCD}(a,b) = 1$$

Contoh :  $\text{GCD}(20,3) = 1$

### 2.1.4 Least Common Multiple ( LCM )

*Least Common Multiple (LCM)*, atau dikenal juga sebagai Kelipatan Persekutuan Terkecil (KPK), adalah hasil perkalian dua bilangan dengan Faktor Persekutuan Terbesar (FPB) dari keduanya. (Matahelumual, *et al.* 2020).

$$\text{LCM}(m,n) = \frac{m \times n}{\text{GCD}(m,n)}$$

Contoh : LCM (18, 24) ?

$$\text{LCM}(18, 24) = \frac{18 \times 24}{\text{GCD}(18, 24)}$$

$$\text{LCM}(18, 24) = \frac{432}{6}$$

$$\text{LCM}(18, 24) = 72$$

### 2.1.5 Aritmatika Modulo

Aritmatika modulo adalah elemen dasar yang dimanfaatkan oleh algoritma RSA. Konsep ini memanfaatkan sifat-sifat dalam bidang di mana sebuah bilangan, setelah melalui beberapa operasi, akan memberikan sisa yang membentuk suatu siklus ketika dibagi dengan bilangan tertentu. Dalam pola ini, siklusnya tetap konsisten untuk sejumlah operasi tertentu. Jenis transformasi ini dikenal sebagai *kongruensi* (Rhomdani & Ningtyas, 2021).

- Jika sebuah bilangan bulat  $a$  memiliki nilai yang sama dengan bilangan bulat  $b$  modulo  $n$  (dengan  $a$  dan  $b$  sebagai bilangan bulat, serta  $n$  sebagai bilangan bulat positif lainnya), jadi dapat dirumuskan :

$$a \equiv b \pmod{n}$$

Contoh :

$$98 \equiv 2 \pmod{8}$$

- Jika  $ac \equiv bc \pmod{n}$ , maka tidak harus benar bahwa  $a \equiv b \pmod{n}$

Contoh :

$$15 \times 2 \pmod{10} \equiv 20 \times 2 \pmod{10}, \text{ namun } 15 \not\equiv 20 \pmod{10}$$

- Jika  $x \equiv y \pmod{n}$  dan  $a \equiv b \pmod{n}$ , dapat dirumuskan sebagai berikut :

$$a + x \equiv b + y \pmod{n}$$

dan

$$ax \equiv by \pmod{n}$$

Contoh :

$$10 \equiv 3 \pmod{7} \text{ dan } 15 \equiv 1 \pmod{7}.$$

Jadi, dapat ditulis juga sebagai :  $150 \equiv 1 \pmod{7}$ .

### 2.1.6 Modulo Eksponensial ( Modular Exponentiation )

Modulo eksponensial melibatkan serangkaian operasi modular, yang mencakup operasi perkalian dan pembagian dengan angka modulo dalam skalayang besar. Dalam konteks ini, perhitungan modulo eksponensial memerlukan alokasi sumber daya yang signifikan baik dalam hal ruang maupun waktu. Jika  $a$  dan  $b$  merupakan 2 bilangan bulat positif, jadi untuk suatu modulus positif tertentu  $n$ , dapat dirumuskan sebagai berikut:

$$c = a^b \bmod n$$

Peranan signifikan dalam algoritma RSA terletak pada operasi eksponensial. Proses enkripsi dan dekripsi dalam RSA bergantung pada modulo eksponensial. Meskipun terdapat variasi algoritma eksponensial modular, algoritma yang paling umum digunakan dalam implementasi RSA adalah *Square and Multiply (SAM)*. Algoritma ini efektif dalam mengatasi kompleksitas perhitungan dengan mengonversi masalah panjang menjadi serangkaian langkah perkalian dan pengkuadratan modular.

Berikut langkah-langkah yang digunakan dalam algoritma *Square and Multiply (SAM)*:

1. Konversikan nilai eksponennya menjadi nilai biner.
2. Ambil nilai  $z = 1$ .
3. *Update* nilai  $z$  dengan cara mengamati eksponen yang telah diubah ke dalam biner dari sisi kiri ke sisi kanan dengan ketentuan :
  - a. Bila bertemu bit 0, maka *update*  $z$  menjadi :  $z^2 \bmod n$
  - b. Bila bertemu bit 1, maka *update*  $z$  menjadi :  $x \cdot z^2 \bmod n$
4. Nilai  $z$  terakhir adalah solusi untuk  $c = x^y \bmod n$ .

Contoh :

Berapa modulo eksponensial dari  $8^{15} \bmod 40$ ?

1. Konversi nilai eksponen menjadi nilai biner :

$$15_{10} = 1111_2$$

2. Ambil nilai  $z = 1$  dan  $x = 8$
3. Update nilai  $z$  dari kiri ke kanan :
  - a. Bit 1 :  $z = x \cdot z^2 \bmod n = 8 \cdot 1^2 \bmod 40 = 8$
  - b. Bit 1 :  $z = x \cdot z^2 \bmod n = 8 \cdot 8^2 \bmod 40 = 32$
  - c. Bit 1 :  $z = x \cdot z^2 \bmod n = 8 \cdot 32^2 \bmod 40 = 32$



d. Bit 1 :  $z = x \cdot z^2 \bmod n = 8 \cdot 32^2 \bmod 40 = 32$

4. Maka, modulo eksponensial dari  $8^{15} \bmod 40 = 32$ .

### 2.1.7 Invers Modulo

*Invers* modulo merupakan balikan dari modulo. Dalam menghitung *invers* pada modulo membutuhkan waktu yang lama karena sedikit rumit. Jika  $a$  dan  $n$  ialah bilangan relatif prima dan  $n > 1$ , jadi terdapat suatu *invers* dari  $a \pmod{n}$ . *Invers* dari  $a \pmod{n}$  ialah sebuah bilangan bulat  $x$  yang memenuhi syarat :

$$x \cdot a \equiv 1 \pmod{n}$$

Notasinya dapat dituliskan sebagai berikut :

$$a^{-1} \pmod{n} = x$$

Salah satu cara untuk menghitung *invers* modulo, dapat menggunakan algoritma *Extended Euclidean*. Berikut adalah langkah-langkah penyelesaiannya :

1. Jika kita ingin menghitung secara berurutan, maka kita dapat menggunakan algoritma pembagian. Caranya adalah dengan menghitung hasil bagi  $q_i$  dan sisa  $r_i$ :

$$a = b \cdot q_0 + r_0$$

$$b = r_0 \cdot q_1 + r_1$$

$$r_0 = r_1 \cdot q_2 + r_2$$

.

.

.

$$r_{i-1} = r_i \cdot q_{i+1} + r_{i+1}$$

2. Substitusi sisa dengan rumus yang melibatkan pendahulunya sampai nilai  $a$  dan  $b$  tercapai sebagai berikut:

.

.

.

$$r_2 = r_0 - q_2 \cdot r_1$$

$$r_1 = b - q_1 \cdot r_0$$

$$r_0 = a - q_0 \cdot b$$

3. Persamaan akhir menyatakan dalam bentuk *Identitas Bézout* yaitu :  $\text{GCD}(a, b) = s \cdot a + t \cdot b$ . Dalam kasus tertentu , dimana  $\text{GCD}(a, b) = 1$ , maka  $t$  merupakan *invers* perkalian dari  $b$  modulo  $a$ , serta  $s$  merupakan *invers* perkalian dari  $a$  modulo  $b$ .

Contoh :

Berapakah *invers* modulo dari 17 (mod 43)?

$$\text{GCD}(17, 43) = 1$$

- $17x \equiv 1 \pmod{43}$   
 $x \equiv 1/17 \pmod{43} \quad x \equiv 17^{-1} \pmod{43}$
- $43 = 17 \times 2 + 9 \longrightarrow 9 = 43 - 17 \times 2 \dots\dots\dots(1)$   
 $9 = 9 \times 1 + 8 \longrightarrow 8 = 17 - 9. \dots\dots\dots(2)$   
 $9 = 8 \times 1 + 1 \longrightarrow 1 = 9 - 8. \dots\dots\dots(3)$
- Substitusi persamaan 2 ke 3 :  $1 = 9 - 8$   
 $1 = 9 - (17 - 9)$   
 $1 = 9 - 17 + 9$   
 $1 = 2 \times 9 - 17 \dots\dots\dots(4)$
- Substitusi persamaan 1 ke 4 :  $1 = 2 \times 9 - 17$   
 $1 = 2(43 - 17 \times 2) - 17$   
 $1 = 2 \times 43 - 4 \times 17 - 17$   
 $1 = 2 \times 43 - 5 \times 17$   
 Dimana :  $-5 \equiv 38 \pmod{43}$ . Maka, kita dapat menyatakan bahwa  $17^{-1} \equiv 38 \pmod{43}$ .

### 2.1.8 Totient Euler ( $\phi$ )

*Totient euler* adalah total bilangan bulat yang sama dengan atau kurang dari bilangan bulat  $n$ , dan juga memiliki sifat relatif prima dengan bilangan  $n$ .

- Jika  $\text{GCD}(a, n) = 1$  untuk beberapa bilangan bulat  $a$  dan  $n$ , maka :

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

- Jika  $m$  ialah bilangan prima, jadi semua bilangan positif yang kurang dari  $m$  akan bersifat relatif prima dengan  $m$ , maka :

$$\phi(m) = m - 1$$

- Jika  $p$  dan  $q$  ialah 2 bilangan prima yang memiliki nilai yang berbeda-beda, maka :

$$\begin{aligned} \phi(pq) &= \phi(p) \cdot \phi(q) \\ \phi(pq) &= (p-1)(q-1) \end{aligned}$$

Contoh :

Berapakah  $\phi(7)$  ?

$$\phi(7) = 7 - 1$$

$$\phi(7) = 6$$

## 2.2 Kriptografi

Kriptografi adalah istilah yang berasal dari bahasa Yunani, yang terdiri dari 2 kata, yaitu "*kryptos*" yang memiliki arti menyembunyikan, dan "*graphein*" yang memiliki arti menulis. Kriptografi ialah disiplin ilmu dan seni yang berperan dalam menjaga kerahasiaan pesan saat proses pengiriman dari suatu lokasi ke lokasi lainnya, dengan cara mengubahnya menjadi format yang tidak dapat dipahami melalui penerapan algoritma tertentu. (Hafiz, 2019). Istilah umum yang digunakan dalam kriptografi meliputi:

### a. Plaintext

*Plaintext* ialah pesan awal yang belum melewati proses enkripsi sehingga masih dapat dibaca dan dimengerti.

### b. Ciphertext

*Ciphertext* ialah pesan awal yang sudah dienkripsi sehingga sulit untuk dibaca langsung dan dimengerti.

### c. Enkripsi

Enkripsi ialah proses mengonversi *plaintext* menjadi *ciphertext* dengan memanfaatkan kunci tertentu.

### d. Dekripsi

Dekripsi ialah proses mengonversi *ciphertext* menjadi *plaintext* sehingga pesan dapat dibaca dan dimengerti oleh penerima.

### e. Key

*Key* ialah parameter yang digunakan untuk melakukan teknik kriptografi yang bersifat rahasia. *Key* yang digunakan dapat berupa karakter, angka, dan spesial karakter.

## 2.3 Tujuan Kriptografi

Secara keseluruhan, kriptografi bertujuan untuk menjaga kerahasiaan data atau informasi dari akses oleh pihak yang tidak sah. Adapun tujuan dari kriptografi dapat dirinci sebagai berikut: (Hermawati *et al.*, 2023):

a. Otentikasi (*Authentication*)

Otentikasi ialah proses yang terkait dengan pengenalan, baik untuk memverifikasi keabsahan pihak yang terlibat dalam komunikasi maupun keabsahan sumber pesan.

b. Kerahasiaan (*Confidentiality*)

Kerahasiaan ialah proses yang memiliki tujuan untuk melindungi pesan dari akses oleh pihak yang tidak sah.

c. Integritas Data (*Data Integrity*)

Integritas data ialah proses yang memiliki tujuan untuk menyakinkan bahwa pesan tetap lengkap dan tidak mengalami perubahan atau manipulasi saat proses pengiriman.

d. Tidak ada penyangkalan (*Non-Repudiation*)

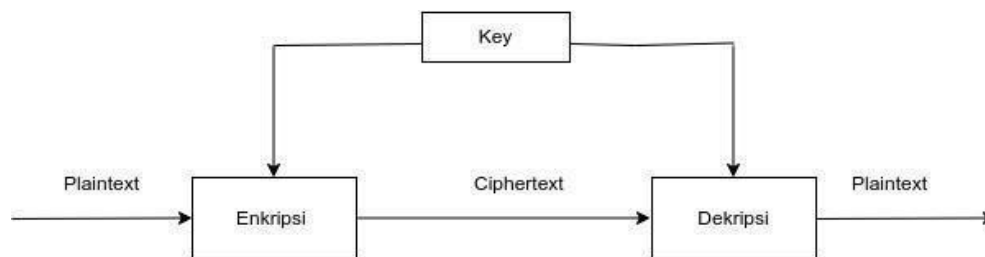
Layanan yang bertujuan untuk mencegah pihak yang terlibat dalam proses kriptografi tidak dapat melakukan penyangkalan.

## 2.4 Jenis-jenis Kriptografi

Kriptografi dapat dibagi menjadi 2 jenis berdasarkan kunci yang digunakan, yaitu kriptografi simetris (dengan kunci tunggal) dan kriptografi asimetris (dengan kunci publik) (Anshori, *et al.* 2019).

1. Kriptografi Simetris

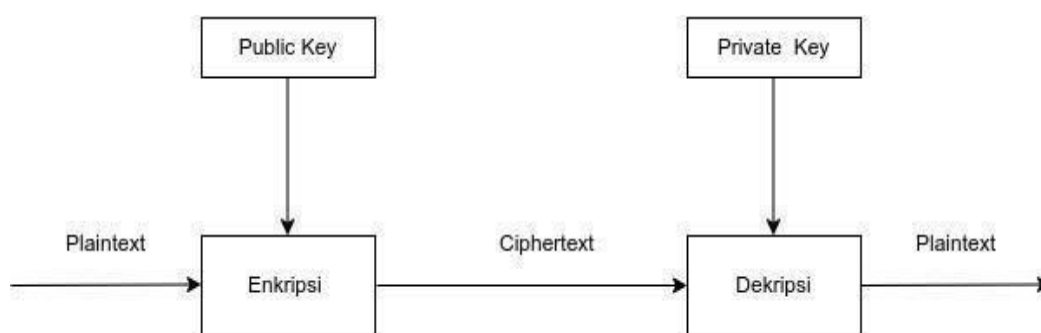
Kriptografi simetris, juga dikenal sebagai kriptografi kunci-rahasia atau *secret-key cryptography* adalah jenis kriptografi di mana proses pengenkripsian dan pendeskripsian menggunakan kunci yang identik yang harus dirahasiakan. Salah satu kelemahan utama dari kriptografi ini adalah bahwa pengirim pesan perlu menemukan cara yang terjamin untuk mengirimkan kunci kepada penerima. Contoh dari algoritma kriptografi ini mencakup AES (*Advanced Encryption Standard*), DES (*Data Encryption Standard*), dan lainnya. Gambaran skema kriptografi simetris dapat dilihat pada Gambar 2.1



*Gambar 2. 1 Kriptografi Simetris*

## 2. Kriptografi Asimetris

Kriptografi asimetris, atau dikenal sebagai kriptografi kunci publik, merupakan jenis kriptografi di mana proses penyandian dan pendekripsian memakai dua kunci yang berbeda. Kunci enkripsi, yang disebut kunci publik, dapat diakses oleh siapa saja, sedangkan kunci dekripsi, yang disebut kunci privat, hanya diketahui oleh penerima pesan yang sah. Ada dua keunggulan utama dari kriptografi ini, yakni kunci publik dapat disampaikan melewati jalur yang sama dengan yang dipakai untuk mengirim pesan, dan jumlah kunci yang diperlukan bisa dikurangi sehingga untuk berkomunikasi dengan sejumlah pihak yang bersifat rahasia, tidak harus menggunakan kunci privat sebanyak pihak tersebut. Contoh algoritma dari kriptografi ini mencakup *ElGamal*, *LUC*, *RSA (Rivest-Shamir-Adleman)*, *Schmidt-Samoa*, *ECC (Elliptic Curve Cryptography)*, *Vincent-Sathiyamoorthy*, dan lainnya. Gambaran skema kriptografi asimetris dapat dilihat pada Gambar 2.2.



*Gambar 2. 2 Kriptografi Asimetris*

## 2.5 Algoritma RSA

Algoritma RSA ialah algoritma tertua serta algoritma kriptografi kunci publik yang sudah umum digunakan. Pada tahun 1977, Ron Rivest, Adi Shamir, dan Leonard Adleman menemukan sebuah algoritma yang dinamakan algoritma RSA. Dalam algoritma kriptografi RSA, terdapat 2 kunci yang nilainya berbeda yang digunakan dalam proses penyandian dan pendeskripsian. Kunci publik dipakai saat proses penyandian, sementara kunci privat digunakan pada proses pendeskripsian. Algoritma RSA adalah algoritma yang paling aman dipakai dalam fitur kerahasiaan dan privasinya, dikarenakan memiliki kompleksitas dan kunci yang sangat besar sehingga penyerang seharusnya tidak dapat memecahkan pemfaktoran pada RSA. Namun, kelemahan pada RSA adalah membutuhkan waktu yang lama pada saat melakukan enkripsi. RSA umumnya digunakan dalam skema enkripsi *hybrid*, tanda tangan digital, aplikasi *chatting*, surat elektronik dan beberapa layanan lainnya yang memerlukan pengiriman informasi yang terjamin ke server atau individu. Beberapa parameter yang terdapat pada algoritma RSA adalah:

- a.  $p$  &  $q$  adalah bilangan prima (bersifat privat)
- b.  $n$  adalah hasil perkalian  $p$  dan  $q$  (bersifat publik)
- c.  $\phi(n)$  adalah hasil kali dari  $p - 1$  dengan  $q - 1$  (bersifat publik)
- d.  $e$  adalah kunci publik (bersifat publik)
- e.  $d$  adalah kunci privat (bersifat privat)
- f.  $m$  adalah *plaintext* (bersifat privat)
- g.  $c$  adalah *ciphertext* (bersifat publik)

Adapun proses pada algoritma RSA adalah sebagai berikut:

### 1. Pembangkitan Kunci

Dalam proses pembuatan sandi, RSA memiliki proses untuk membentuk kunci publik serta kunci privat:

- a. Bangkitkan 2 bilangan prima besar dengan sembarang untuk  $p$  &  $q$ , di mana nilai  $p$  tidak sama dengan  $q$ .
- b. Hitunglah *modulus* RSA dengan menggunakan rumus:  $n = p * q$ .
- c. Hitunglah fungsi *totient euler*  $\phi(n)$  menggunakan rumus :

$$\phi(n) = (p - 1) * (q - 1).$$

- d. Pilih kunci publik  $e$  sedemikian rupa sehingga  $e$  adalah relatif prima terhadap

$\phi(n)$ . Dengan demikian, nilai  $e$  harus memenuhi syarat  $1 < e < \phi(n)$  serta GCD  $e$  dan  $\phi(n)$  sama dengan 1.

- e. Hitung kunci privat memakai rumus persamaan berikut :

$$d \equiv e^{-1} \pmod{\phi(n)} \dots\dots\dots(1)$$

Dari langkah-langkah di atas, maka didapatkan pasangan kunci sebagai berikut :

- a. Kunci publik adalah  $e$  &  $n$
- b. Kunci privat adalah  $d$  &  $n$

## 2. Proses Enkripsi

Tahapan-tahapan dalam menjalankan proses enkripsi pesan pada algoritma RSA mencakup:

- a. Dapatkan kunci publik  $e$  dan juga *modulus*  $n$ .
- b. Konversi teks biasa  $M$  menjadi serangkaian blok, seperti  $m_1, m_2, \dots$ , di mana tiap blok mewakili nilai dalam rentang  $[0, n-1]$ .
- c. Semua blok  $M_i$  diubah menjadi blok  $C_i$  melalui proses enkripsi dengan menggunakan rumus :

$$C_i = M_i^e \pmod{n} \dots\dots\dots(2)$$

## 3. Proses Dekripsi

Tahapan-tahapan dalam menjalankan proses dekripsi pesan pada algoritma RSA mencakup:

- a. Dapatkan kunci privat yaitu  $d$ .
- b. Konversi teks biasa (*plaintext*) menjadi sejumlah blok, seperti  $c_1, c_2, \dots$ , di mana masing-masing blok melambangkan nilai dalam rentang  $[0, n-1]$ .
- c. Semua blok  $M_i$  diubah menjadi blok  $C_i$  melalui proses dekripsi dengan menggunakan rumus:

$$M_i = C_i^d \pmod{n} \dots\dots\dots(3)$$

## 2.6 Algoritma Vincent-Sathiyamoorthy

*Vincent-Sathiyamoorthy* merupakan algoritma perkembangan dari algoritma *RSA* (*Rivest-Shamir-Adleman*) yang ditemukan oleh P.M. Durai Raj Vincent dan E. Sathiyamoorthy pada tahun 2016. Algoritma ini memiliki kemiripan dengan RSA yaitu memiliki kunci publik yang sama, namun dapat melakukan pengurangan waktu pada saat melakukan proses enkripsi dekripsi. Pendekatan yang diusulkan ialah sebagai berikut :

### 1. Pembentukan kunci publik

Algoritma dimulai dengan pembentukan kunci publik karena kunci publik perlu dibagikan penerima untuk mengenkripsi. Pembentukan kunci publik ialah sebagai berikut :

- a. Pilih tiga angka secara sembarang, yakni  $p$ ,  $q$ , dan  $r$ , di mana  $p$  merupakan bilangan prima dan harus lebih kecil dari hasil perkalian  $q$  dan  $r$  ( $p < qr$ ).
- b. Berdasarkan kondisi pada langkah sebelumnya, nilai yang sesuai akan dipilih.

Setelah itu adalah menghitung nilai dari :

$$s = qr - p$$

$$e = ps + r$$

$$t = p^2s + q$$

- c. Kemudian hitung kunci publik dengan rumus:

$$n = \frac{et-p}{s}$$

Setelah menyelesaikan langkah-langkah di atas, kita akan mendapatkan berbagai nilai seperti  $s$ ,  $e$ ,  $t$  dan  $n$ . Sekarang kunci telah dibuat dan kunci publik akan menjadi  $\langle e, n \rangle$ . Nilai ini sekarang akan dibagikan dan pengirim melakukan proses enkripsi.

### 2. Enkripsi

Setelah mendapat pasangan kunci publik, pengirim akan mengenkripsi pesan. Pengirim sudah siap dengan *plaintext* 'M' dan melakukan langkah yang diberikan untuk melakukan proses enkripsi. Biarkan 'M' menjadi teks biasa, hitung teks sandi menggunakan rumus:

$$C = M * e \pmod n$$



### 3. Dekripsi

Setelah mengenkripsi *plaintext* pengirim akan mengirimkan *ciphertext* ke penerima. Sekarang penerima perlu membuat kunci pribadi untuk mendapatkan kembali teks asli. Dengan demikian proses dekripsi dimulai dengan pembuatan kunci privat.

#### a. Pembentukan kunci privat

Untuk mendapatkan kunci privat, pada langkah ini kita akan memecahkan persamaan berikut :

$$P * u \pmod{n} \equiv 1 \text{ dan } 0 \leq u \leq n$$

Setelah mendapatkan nilai  $u$  langkah yang selanjutnya adalah menghitung nilai “ $d$ ” dengan rumus :

$$d = u * t$$

Setelah menghitung nilai ' $d$ ', kunci privat selesai dan akan menjadi  $\langle d, n \rangle$ .

#### b. Dekripsi

Pada langkah terakhir untuk mendapatkan kembali teks asli dari *ciphertext* yang diterima dari pengirim dapat menggunakan rumus:

$$M = C * d \pmod{n}$$

## 2.7 Kompleksitas Algoritma

Kompleksitas algoritma menitik beratkan pada seberapa efisien atau lambat performa suatu algoritma. Efisiensi diukur melalui kompleksitas, yang dinyatakan sebagai fungsi  $T(n)$ , yang menunjukkan waktu yang dibutuhkan algoritma relatif terhadap jumlah inputnya. Pengukuran waktu suatu algoritma dapat dilakukan tanpa memperhatikan detail implementasinya. Meskipun algoritma dapat menunjukkan waktu yang bervariasi untuk inputan yang bernilai sama, hal ini bergantung pada faktor-faktor seperti kecepatan prosesor, instruksi yang digunakan, merek komponen, dan faktor lainnya. Metode yang tepat untuk menilai efisiensi algoritma adalah dengan menghitung kompleksitasnya dalam konteks *asimtotik*. Ketika waktu  $T(n)$  diinterpretasikan sebagai "total langkah", jadi setiap langkah seharusnya dilaksanakan dalam waktu tetap dan satuan yang serupa.

## 2.8 Penelitian Relevan

Beberapa studi yang berkaitan dengan riset yang dilaksanakan penulis adalah sebagai berikut:

1. Berdasarkan penelitian (Suhandinata, *et al.* 2019) yang berjudul “*Analisis Performa Kriptografi Hybrid Algoritma Blowfish dan Algoritma RSA*” hasil penelitian menunjukkan bahwa proses pengenkripsian dan pendekripsian menggunakan metode RSA memerlukan waktu yang lebih lama dibandingkan dengan *Blowfish* atau *hybrid*. Namun, perbedaan waktu enkripsi dekripsi antara *hybrid* dan *Blowfish* hanya sedikit yaitu beberapa saat aja. Selain itu, algoritma *hybrid* menunjukkan kinerja yang cukup cepat dalam proses enkripsi dan dekripsi, meskipun tingkat kecepatannya tidak secepat algoritma yang menjadi komponennya.
2. Berdasarkan penelitian (Vahdati, *et al.* 2019) yang berjudul “*Comparison Of ECC and RSA Algorithms in IoT Device* ” disimpulkan bahwa penggunaan algoritma ECC pada perangkat *IoT* memiliki potensi yang lebih besar dikarenakan ukuran kunci yang lebih kecil dan kinerja yang lebih cepat serta efisien. Sebaliknya, RSA lebih cocok digunakan dalam beberapa aplikasi yang lebih menekankan pada verifikasi pesan dari pada pembuatan tanda tangan.
3. Berdasarkan penelitian (Yousif, 2023) yang berjudul “ *Performance Comparison between RSA and El-Gamal Algorithms for Speech Data Encryption and Decryption* ” memperoleh hasil bahwa kedua algoritma ini kuat dan dapat menyandikan serta menguraikan data dengan tingkat kepercayaan, keamanan dan privasi yang tinggi. Namun, kinerja RSA lebih unggul daripada El-Gamal dalam sebagian besar analisis yang dilakukan.
4. Berdasarkan penelitian (Jnr, *et al.* 2023) yang berjudul “*A Comparative Study of RSA and Elgamal Cryptosystems*” menyatakan bahwa RSA cenderung memiliki kecepatan yang lebih tinggi dibandingkan Elgamal dalam proses enkripsi dan dekripsi, tetapi terdapat beberapa pertimbangan yang harus dipertimbangkan dalam proses pemilihan algoritma kriptografi yang sesuai, seperti tingkat keamanan yang dibutuhkan, ukuran kunci, jenis aplikasi, dan kebutuhan kinerja.
5. Berdasarkan penelitian (Marqas, *et al.* 2020) yang berjudul “*Comparing Symmetric and Asymmetric cryptography in message encryption and decryption by using AES and RSA algorithms*” memperoleh hasil bahwa AES lebih fleksibel daripada RSA

dan menunjukkan kinerja yang lebih baik serta lebih cepat dikarenakan AES memiliki tiga ukuran kunci yang berbeda (128, 192, 256), sedangkan RSA hanya memiliki dua ukuran kunci yang berbeda (1024, 4096).

6. Berdasarkan penelitian (Anwar, *et al.* 2019) yang berjudul “*Comparative Study of Cryptography Algorithms and Its’ Applications*” memperoleh hasil bahwa algoritma kriptografi simetris cocok untuk aplikasi seperti komunikasi nirkabel, pengolahan gambar, atau *e-commerce*. Sementara itu, algoritma kriptografi asimetris lebih baik digunakan untuk aplikasi web, verifikasi *email*, dan perangkat seluler.
7. Berdasarkan penelitian (Sulistiyorini, *et al.* 2019) yang berjudul “*Perbandingan Efisiensi Algoritma RSA dan RSA CRT Dengan Data Teks Berukuran Besar*” disimpulkan bahwa algoritma RSA-CRT memiliki kecepatan waktu sekitar 50% lebih tinggi daripada algoritma RSA dalam melakukan proses dekripsi data teks berukuran besar.
8. Berdasarkan penelitian (Hidayat, *et al.* 2020) yang berjudul “*Implementation of RSA and RSA-CRT Algorithms for Compariosn of Encryption and Decryption Time in Android-bases Instant Message Applications*” menyampaikan perbandingan waktu enkripsi antara RSA dan RSA-CRT tidak menunjukkan perbedaan yang signifikan. Namun, dalam dekripsi, metode CRT pada RSA terbukti sangat efektif, bahkan dengan *modulus* yang berbeda. Pada RSA-CRT 1024 bit, kecepatan dekripsi meningkat sekitar 2,6 kali lebih cepat dibanding RSA 1024 bit.
9. Berdasarkan penelitian (Mallouli, *et al.* 2019) yang berjudul “*A Survey on Cryptography : comparative study between RSA vs ECC Algorithms, and RSA vs El-Gamal Algorithms*” memperoleh hasil bahwa algoritma El-Gamal dianggap lebih aman dibandingkan dengan RSA, namun El-Gamal juga memiliki kekurangan seperti penggunaan satu bilangan acak dan teks sandi yang lebih panjang. Selain itu, ECC juga lebih efisien dan aman digunakan daripada RSA, karena ECC memiliki keunggulan dalam pertukaran kunci dengan panjang kunci lebih pendek untuk tingkat keamanan yang setara RSA.

## **BAB 3**

### **ANALISIS DAN PERANCANGAN SISTEM**

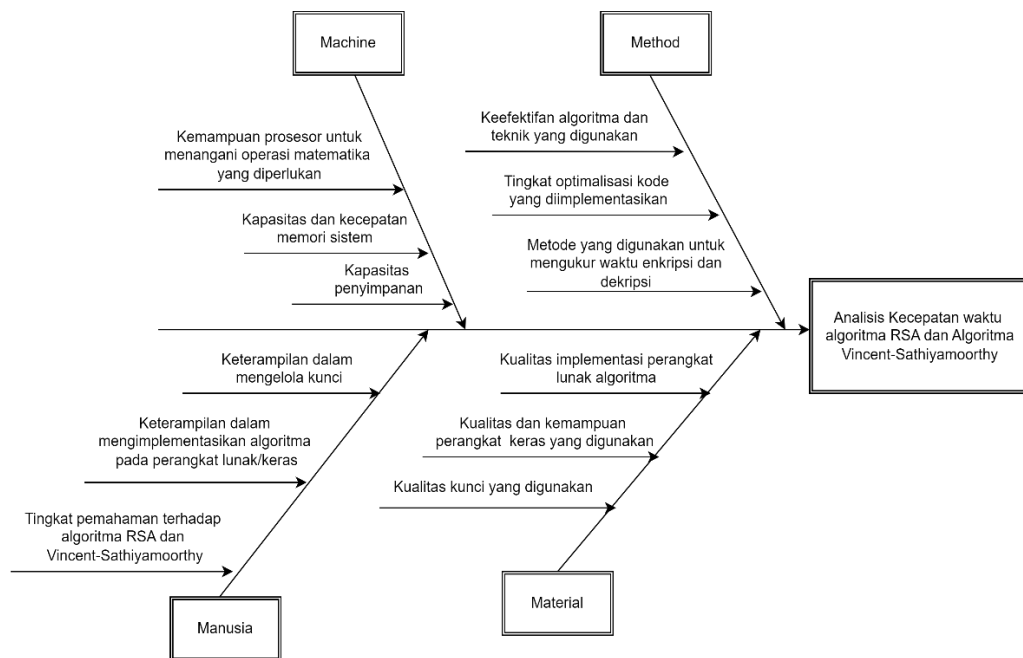
Bab ini membahas mengenai analisis dan perancangan sistem. Analisis sistem dirancang sedemikian rupa dengan menganalisis masalah dan juga kebutuhan dalam penelitian. Selain itu perancangan sistem dilakukan untuk mendapatkan permodelan dan alur perancangan program dalam penelitian.

#### **3.1 Analisis Sistem**

Analisis sistem merupakan proses di mana untuk mendapatkan model, informasi dan spesifikasi terkait dengan software yang diinginkan.

##### *3.1.1 Analisis Masalah*

*Fishbone diagram* menggambarkan akar permasalahan yang memiliki dampak terbesar dalam efisiensi waktu dalam melakukan enkripsi dekripsi pada algoritma RSA dan Vincent-Sathiyamoorthy. Pemecahan kunci merupakan faktor yang paling mempengaruhi dalam analisis ini. Berikut disajikan *fishbone diagram* dalam menganalisis permasalahan ini:



Gambar 3. 1 Fishbone Diagram Enkripsi Dekripsi

### 3.1.2 Analisis Kebutuhan

Ada 2 tipe jenis analisis kebutuhan, yakni :

#### 1. Kebutuhan Fungsional

Kebutuhan fungsional ialah semua keperluan sistem dan tindakan-tindakan yang akan dilakukan oleh sistem tersebut.

- Sistem akan melaksanakan 2 proses, yaitu menerapkan proses pengenkripsian dan pendekripsian pada algoritma RSA dan Vincent-Sathiyamoorthy.
- Dalam tahap pembuatan kunci yaitu kunci publik dan kunci privat, pengguna memasukkan bilangan prima acak yang dibutuhkan masing-masing algoritma.
- Pada tahap enkripsi dekripsi, pengguna memasukkan *plaintext* yang diinginkan dan memperoleh hasil output yang sama dengan yang diinputkan.
- Plaintext* dalam karakter *ASCII* sedangkan *ciphertext* nya direpresentasi dalam bentuk *numerik* (angka).

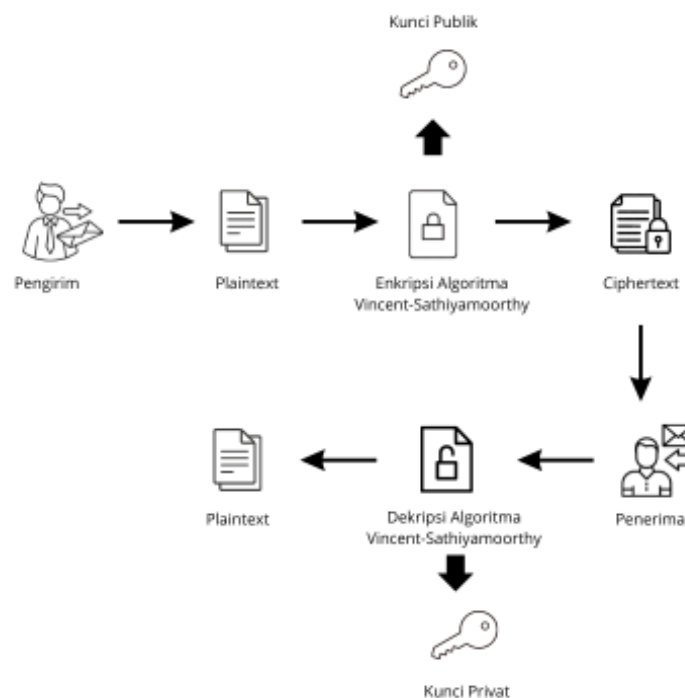
## 2. Kebutuhan Non-Fungsional

Kebutuhan non-fungsional ialah batasan-batasan yang menjadi parameter dalam mengukur tingkat kepuasan terhadap sistem tersebut.

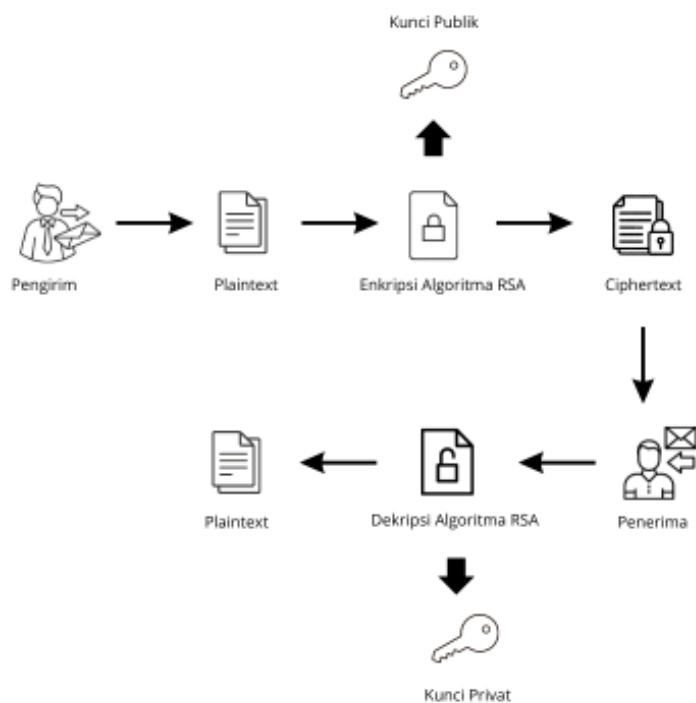
- Sistem mampu melakukan enkripsi dekripsi dalam waktu yang bersamaan.
- Sistem dapat mengeluarkan perintah *error* jika pengguna memasukkan inputan yang salah.
- Sistem memiliki *running time* untuk enkripsi dekripsi pada masing-masing algoritma.

## 3.2 Diagram Umum Sistem

Diagram umum sistem adalah gambaran alur sistem secara menyeluruh. Pertama, pengirim akan mengirimkan pesan berupa *plaintext* yang dimana akan di enkripsi menjadi *ciphertext* dengan menggunakan kunci publik. Setelah itu, pesan dikirim ke penerima dan pesan yang berupa *ciphertext* akan diubah kembali menjadi *plaintext* melewati proses dekripsi menggunakan kunci privat. Setelah pesan telah berubah menjadi *plaintext*, maka pesan tersebut sudah dapat dikirim ke penerima. Adapun diagram umum sistem masing-masing algoritma pada skripsi adalah sebagai berikut:



Gambar 3. 2 Diagram Umum Sistem Algoritma Vincent-Sathiyamoorthy



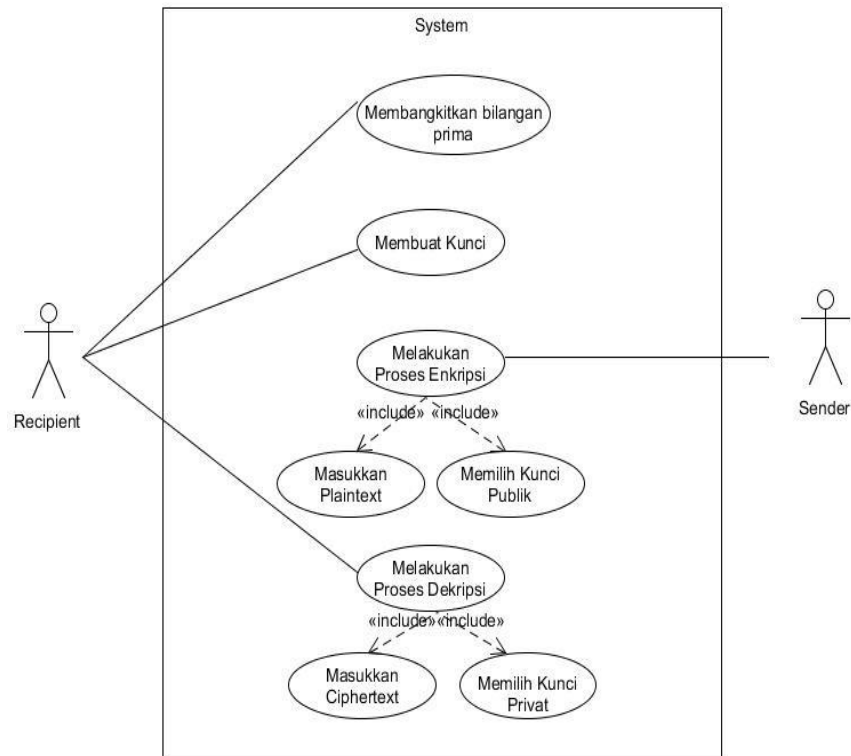
Gambar 3. 3 Diagram Umum Sistem Algoritma RSA

### 3.3 Perancangan Sistem

Perancangan sistem akan menggunakan teknologi *Python* dengan penerapan paradigma pengembangan berorientasi objek. Maka, dalam proses perancangan akan digunakan *activity diagram*, *use case diagram*, serta *sequence diagram*.

#### 3.3.1 Use Case Diagram

*Use case diagram* ialah sebuah diagram yang bertujuan untuk mengilustrasikan interaksi yang terjadi antara sistem dengan pengguna yang akan dikembangkan menggunakan simbol-simbol tertentu guna memperjelas alurnya. Adapun *use case diagram* yang dimanfaatkan dalam pengembangan sistem adalah sebagai berikut:

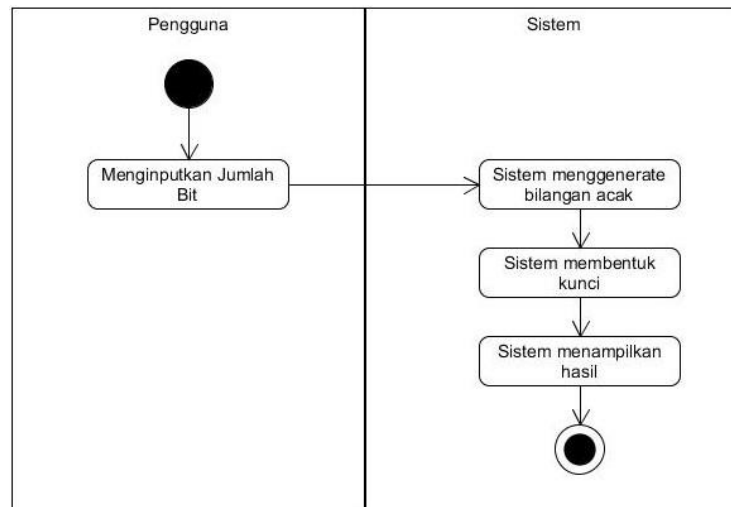


Gambar 3. 4 Use Case Diagram Enkripsi Dekripsi

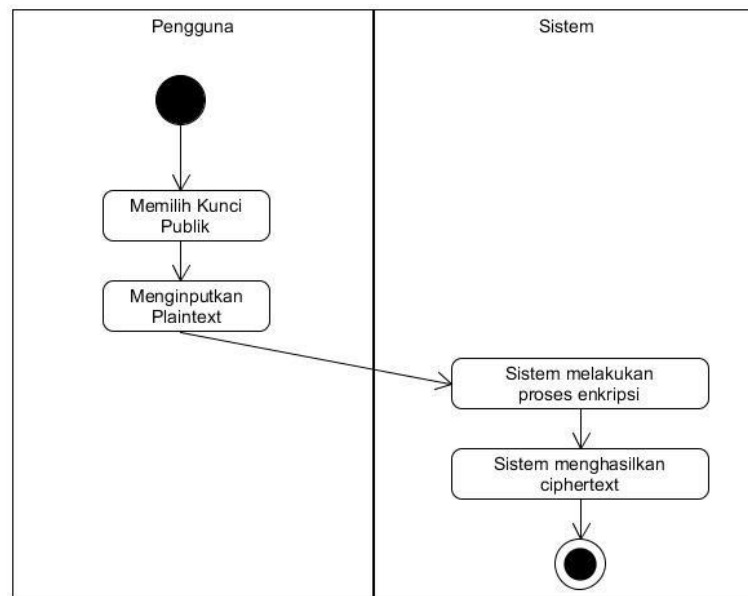
### 3.3.2 Activity Diagram

*Activity diagram* ialah visualisasi dari urutan langkah-langkah di dalam sistem yang akan dilaksanakan. Selain itu, *activity diagram* juga berperan dalam merinci atau mengelompokkan tampilan alur dari sistem tersebut. Pada *activity diagram* terdapat komponen-komponen dengan bentuk khusus yang dihubungkan oleh tanda panah, yang mengilustrasikan rangkaian proses dari awal hingga akhir. Adapun *activity diagram* yang digunakan untuk merancang sistem ialah sebagai berikut:

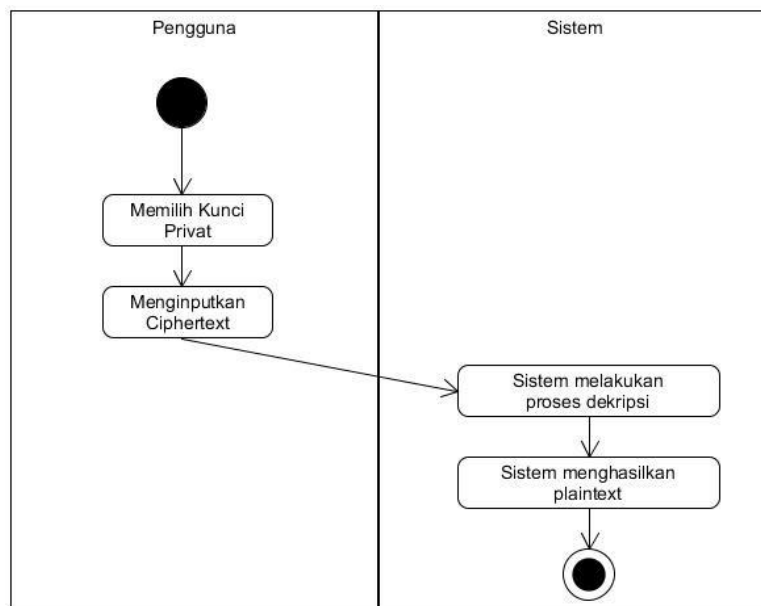




*Gambar 3. 5 Activity Diagram Pembuatan Kunci*



*Gambar 3. 6 Activity Diagram Enkripsi*

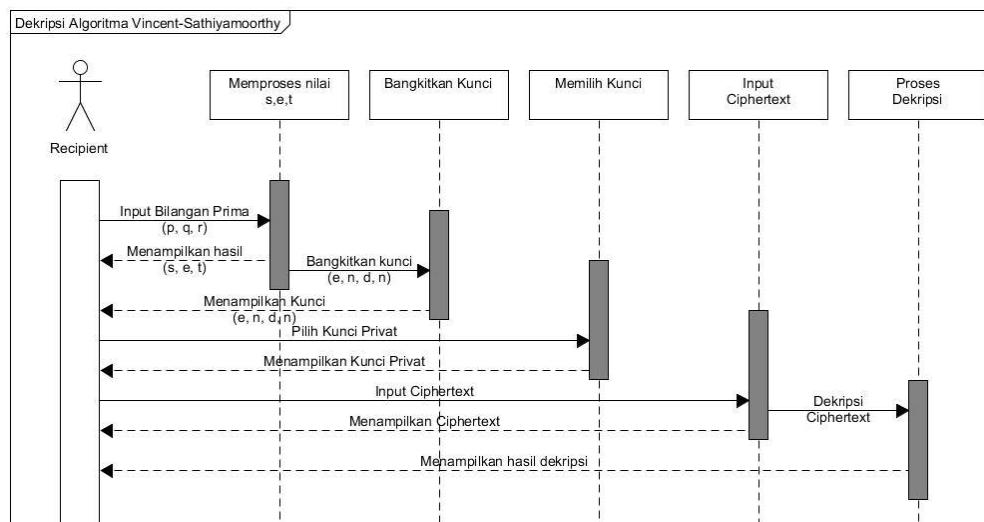


Gambar 3. 7 Activity Diagram Dekripsi

### 3.3.3 Sequence diagram

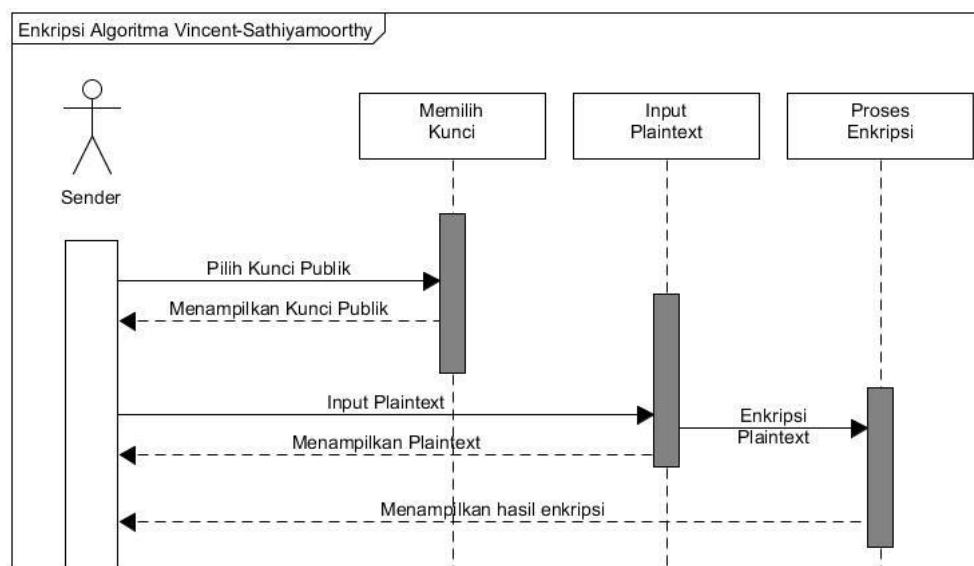
*Sequence diagram* ialah representasi visual yang mengilustrasikan interaksi antara objek dalam suatu kelas. Komponen-komponen dalam *sequence diagram* meliputi: (1) *Aktivations*, yang menjelaskan pelaksanaan fungsi dari suatu objek; (2) *Actor*, yang menggambarkan peran dalam serangkaian aksi dalam proses; (3) *Collaboration boundary*, yang menunjukkan lokasi untuk lingkungan uji coba dan pemantauan objek; (4) *Parallel vertical lines*, yang mewakili alur proses yang mengacu pada sebuah keadaan; (5) *Processes*, yang menggambarkan langkah-langkah yang diambil oleh aktor pada waktu tertentu; (6) *Window*, yang menggambarkan halaman yang sedang ditampilkan selama proses; (7) *Loop*, yang menggambarkan model logika yang memiliki potensi untuk diulang beberapa kali.

a. Algoritma Vincent-Sathiyamoorthy



Gambar 3. 8 Sequence Diagram Dekripsi Algoritma Vincent-Sathiyamoorthy

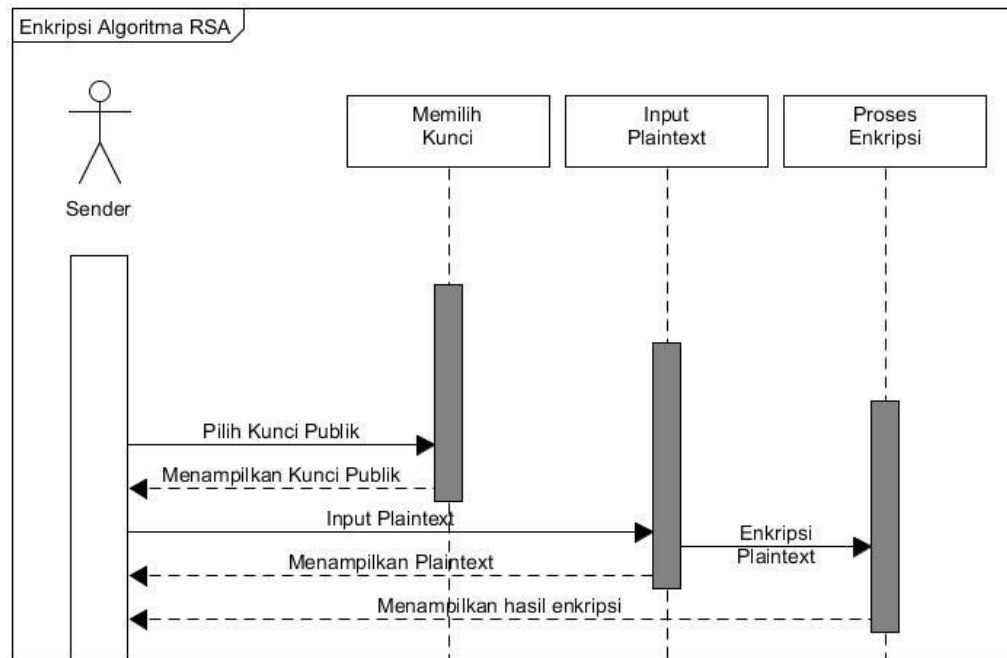
Gambar 3.8 menggambarkan *sequence diagram* interaksi pengguna yang melakukan proses dekripsi algoritma Vincent-Sathiyamoorthy.



Gambar 3. 9 Sequence Diagram Enkripsi Algoritma Vincent-Sathiyamoorthy

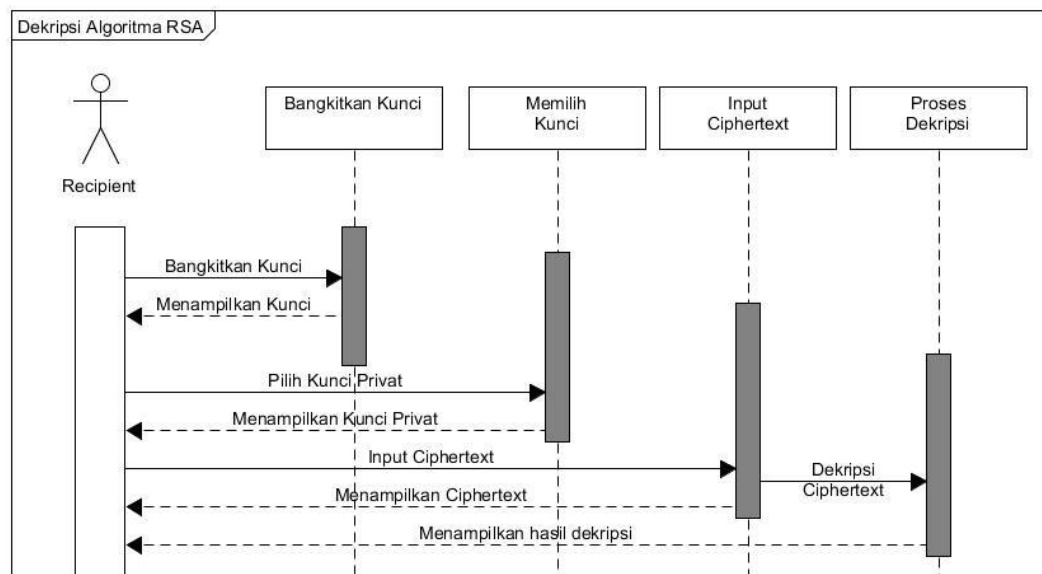
Gambar 3.9 menggambarkan *sequence diagram* interaksi pengguna yang melakukan proses enkripsi algoritma Vincent-Sathiyamoorthy.

b. Algoritma RSA



Gambar 3. 10 Sequence Diagram Dekripsi Algoritma RSA

Gambar 3.10 menggambarkan *sequence diagram* interaksi pengguna yang melakukan proses dekripsi algoritma RSA.

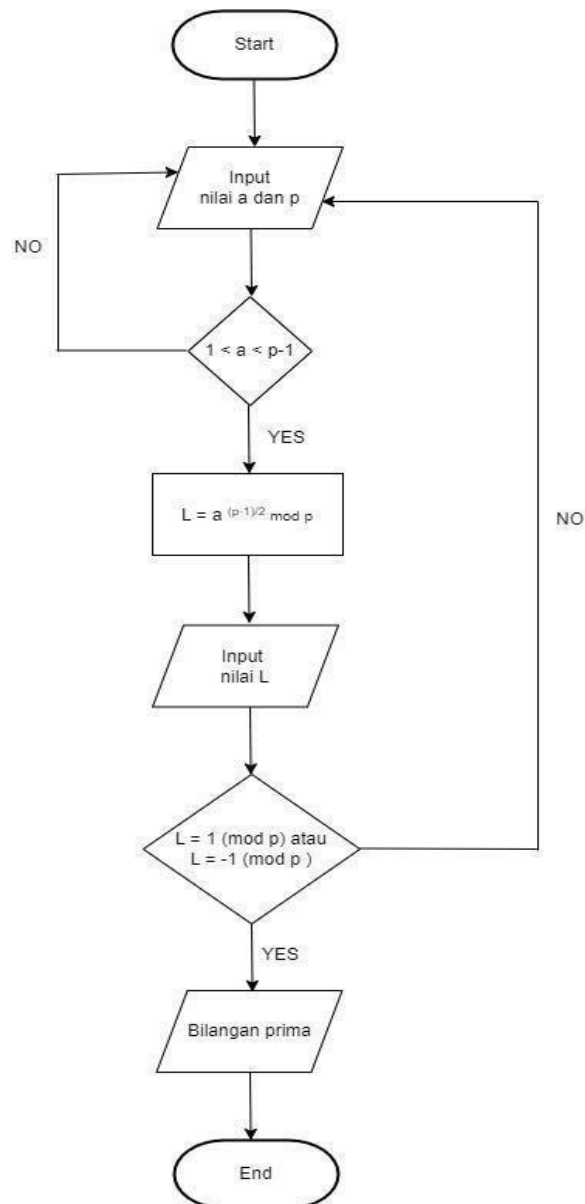


Gambar 3. 11 Sequence Diagram Enkripsi Algoritma RSA

Gambar 3.11 menggambarkan *sequence diagram* interaksi pengguna yang melakukan proses enkripsi algoritma RSA.

### 3.4 Flowchart

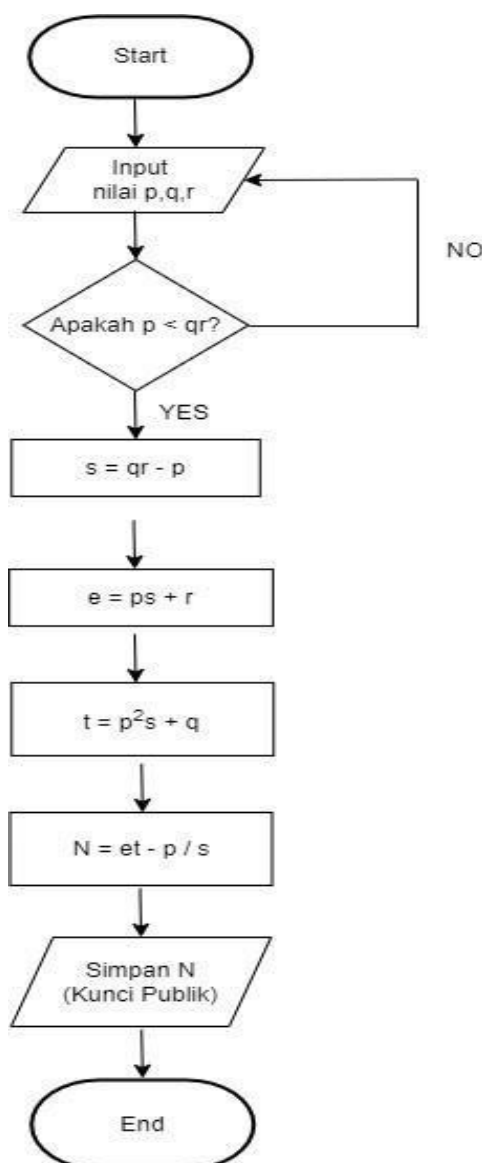
#### 3.4.1 Flowchart Pembangkit Bilangan Prima



Gambar 3. 12 Flowchart Pembangkit Bilangan Prima

Gambar 3.12 menjelaskan proses pembangkitan bilangan prima dengan memasukkan nilai  $a$  dan  $p$ , dimana  $1 < a < p-1$ . Setelah itu, dilakukan perhitungan  $L$  menggunakan rumus  $L = a^{(p-1)/2} \bmod p$ . Jika hasil  $L$  sama dengan  $1$  atau  $-1$ , jadi bilangan tersebut dianggap prima. Namun, jika hasilnya tidak sama dengan  $1$  atau  $-1$ , maka kembali ke langkah awal dengan memasukkan kembali nilai  $a$  dan  $p$ .

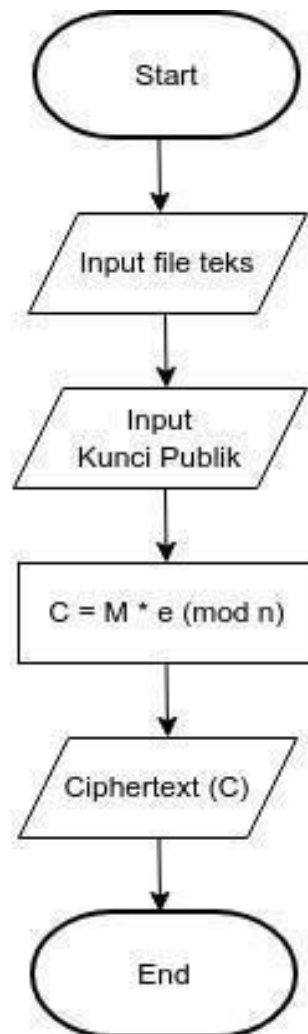
### 3.4.2 Flowchart Pembangkit Kunci Algoritma Vincent- Sathiyamoorthy



Gambar 3. 13 Flowchart Pembangkit Kunci Algoritma Vincent-Sathiyamoorthy

Gambar 3.13 menjelaskan tentang proses pembangkit kunci algoritma Vincent-Sathiyamoorthy yang akan digunakan dalam mengenkripsikan teks. Dalam algoritma Vincent-Sathiyamoorthy, diperlukan tiga bilangan, yang di mana  $p$  merupakan bilangan prima yang harus lebih kecil dari hasil perkalian  $q$  dan  $r$  untuk membangkitkan kunci tersebut. Apabila  $p < qr$  bernilai benar maka perhitungan dapat dilanjutkan dengan mencari nilai  $s$ ,  $e$ ,  $t$  dan  $N$ . Namun, apabila  $p < qr$  bernilai salah maka kembali ke langkah awal yaitu menginputkan nilai  $p, q$  dan  $r$ . Setelah nilai  $n$  didapat, maka simpan nilai tersebut sebagai kunci publik.

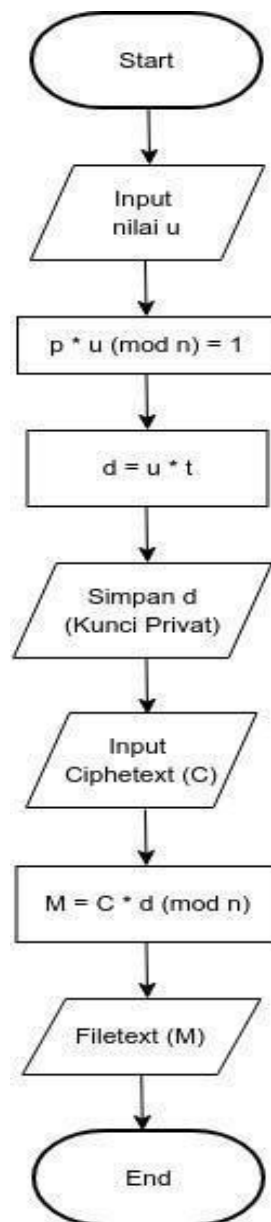
### 3.4.3 Flowchart Enkripsi Algoritma Vincent-Sathiyamoorthy



Gambar 3. 14 Flowchart Enkripsi Algoritma Vincent-Sathiyamoorthy

Gambar 3.14 menjelaskan bahwa proses enkripsi menggunakan kunci publik. Langkah pertama adalah memasukkan file teks dan kunci publik yang sudah disimpan. Kemudian, proses pengenkripsian pada file teks dilakukan dengan memakai rumus  $C = M * e \pmod{n}$ , dan menghasilkan *ciphertext*.

### 3.4.4 Flowchart Dekripsi Algoritma Vincent-Sathiyamoorthy

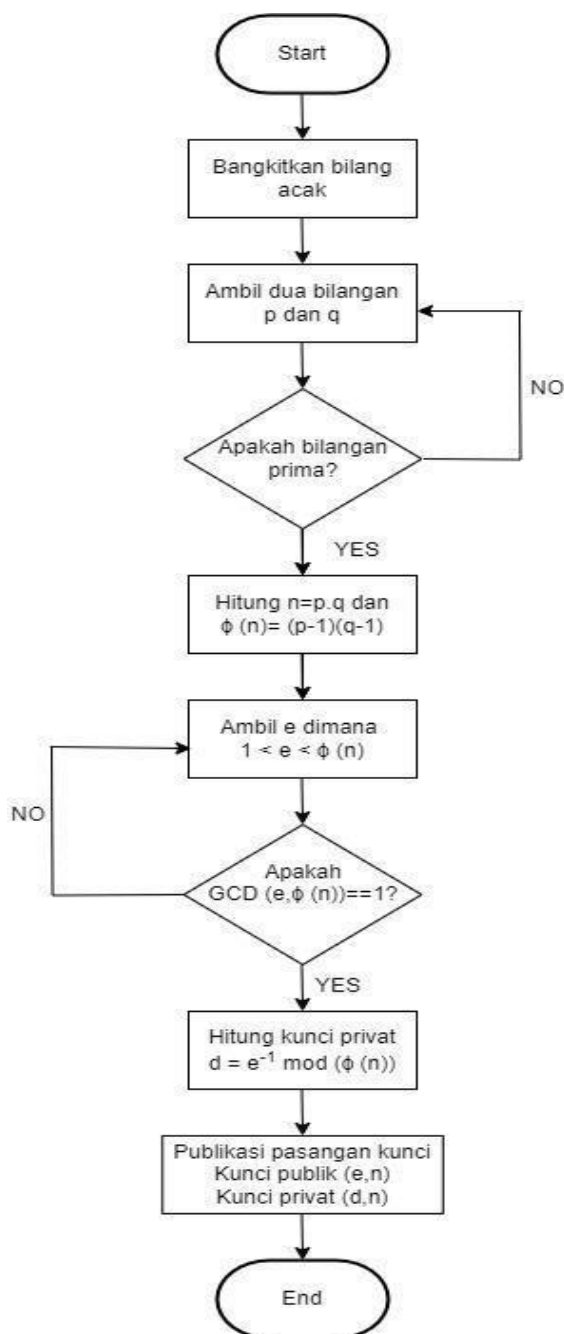


Gambar 3. 15 Flowchart Dekripsi Algoritma Vincent-Sathiyamoorthy

Gambar 3.15 menjelaskan bahwa sebelum *ciphertext* akan di dekripsi maka yang pertama kita lakukan adalah membuat kunci privat nya terlebih dahulu dengan mencari nilai  $u$ . Setelah nilai  $u$  didapat, maka hitung nilai  $d$  dengan rumus  $d = u * t$ . Setelah memperoleh nilai  $d$ , langkah selanjutnya adalah menyimpan nilai tersebut sebagai kunci privat yang digunakan dalam proses dekripsi. Kemudian *input ciphertext* yang akan diubah menjadi *plaintext*. Setelah itu, ubah *ciphertext* menggunakan kunci privat melalui proses dekripsi memakai rumus  $M = C * d \pmod n$  dan akan menghasilkan *plaintext*.



### 3.4.5 Flowchart Pembentukan Kunci Algoritma RSA

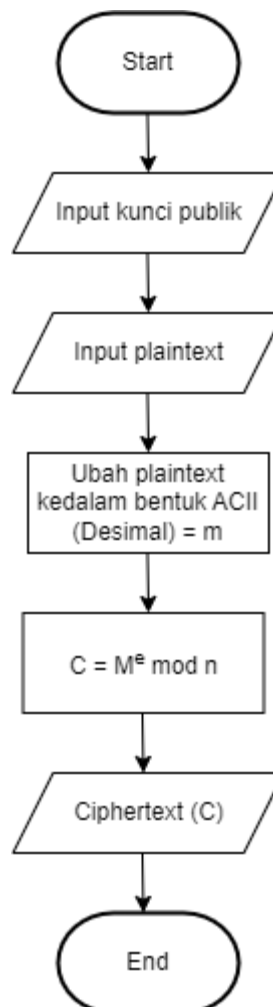


Gambar 3. 16 Flowchart Pembentukan Kunci Algoritma RSA

Gambar 3.16 menjelaskan pembentukan kunci pada algoritma RSA yang digunakan dalam proses pengenkripsian dan pendekripsian. Dalam algoritma RSA, diperlukan 2 bilangan, yaitu p dan q, yang dimana nilai dari kedua bilangan tersebut harus merupakan bilangan prima untuk menghasilkan kunci tersebut. Jika p dan q bukan bilangan prima, proses akan kembali ke langkah awal. Namun, apabila p & q adalah bilangan prima, jadi perhitungan dimulai dengan menentukan nilai

nmenggunakan rumus  $n = p * q$  dan  $\phi(n) = (p-1)(q-1)$ . Selanjutnya, pilihlah nilai  $e$  secara sembarang di mana  $1 < e < \phi(n)$  dan  $\text{GCD}$  antara  $e$  dan  $\phi(n) = 1$ . Setelah itu, hitung nilai  $d$  menggunakan rumus  $d = e^{-1} \bmod \phi(n)$ . Dengan demikian, pasangan kunci publik yaitu  $e$  dan  $n$  dan kunci privat yaitu  $d$  dan  $n$  telah diperoleh.

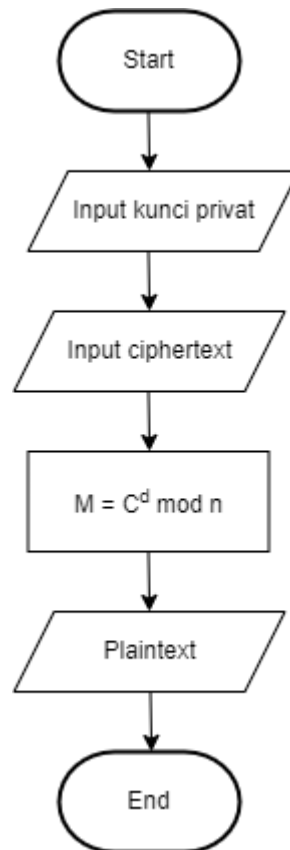
#### 3.4.6 Flowchart Proses Enkripsi Algoritma RSA



Gambar 3. 17 Flowchart Proses Enkripsi Algoritma RSA

Gambar 3.17 menjelaskan bahwa proses *enkripsi* menggunakan kunci publik, di mana langkah pertama yang harus dilakukan adalah menginput file teks kemudian kunci publik yang telah disimpan. Kemudian, ubah *plaintext* kedalam bentuk *ASCII* (desimal). Setelah itu, lakukan pengenkripsian menggunakan rumus  $C = M^e \bmod n$  dan menghasilkan *ciphertext*.

### 3.4.7 Flowchart Proses Dekripsi Algoritma RSA



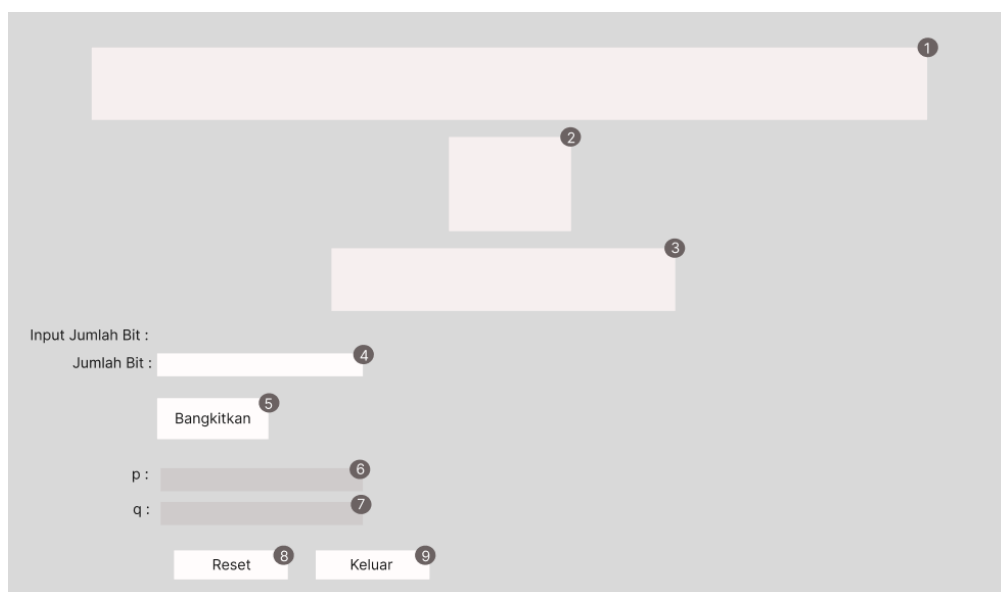
Gambar 3. 18 Flowchart Proses Dekripsi Algoritma RSA

Gambar 3.18 menjelaskan bahwa sebelum *ciphertext* akan di dekripsi maka yang pertama kita lakukan adalah menginput kunci privat yang telah di dapat dan *ciphertext*. Setelah itu, ubah *ciphertext* melalui proses pendekripsian dengan menggunakan rumus  $M = C^d \bmod n$  dan memakai kunci privat dan akan menghasilkan plaintext.

### 3.5 Perancangan User Interface

Rancangan antarmuka pengguna adalah hasil dari perencanaan yang menggambarkan interaksi antara pengguna dengan sistem. Tujuannya adalah untuk merancang tampilan sistem agar sesuai dengan fungsinya dan memberikan kegunaan yang optimal bagi pengguna. Dalam proses ini, berbagai elemen seperti tata letak, warna, jenis huruf, dan ikon dipertimbangkan untuk menciptakan pengalaman pengguna yang intuitif dan efisien. Rancangan antarmuka pengguna yang baik memperhatikan kebutuhan pengguna, prinsip desain *UX/UI*, serta kecocokan dengan tujuan dan identitas merek yang ingin disampaikan oleh sistem tersebut.

#### 3.5.1 Perancangan halaman generate bilangan prima



Gambar 3. 19 Perancangan halaman generate bilangan prima

Pada Gambar 3.19 menjelaskan beberapa komponen yang digunakan dengan keterangan meliputi :

1. *Label* berfungsi untuk mempresentasikan judul penelitian.
2. *Pixture Box* berfungsi untuk mempresentasikan logo universitas.
3. *Label* berfungsi untuk mempresentasikan generate bilangan prima
4. *Textbox* berfungsi untuk mempresentasikan jumlah bit.
5. *Button* berfungsi untuk memproses pembangkit bilangan prima.
6. *Textbox* berfungsi untuk mengeluarkan nilai p.
7. *Textbox* berfungsi untuk mengeluarkan nilai q.

8. *Button* berfungsi untuk mengembalikan keadaan seperti semula.
9. *Button* berfungsi untuk keluar dari sitem.

### 3.5.2 Perancangan halaman algoritma RSA

The diagram shows a web page layout for an RSA algorithm interface. It includes a header area (1), a logo box (2), and a title label (3). The main content area contains input fields for key generation: 'Input Kunci:' with sub-fields for 'p (prima):' (4), 'q (prima):' (5), and 'e (public key):' (6). A 'Bangkitkan' button (7) is positioned between these fields. To the right, there are output fields for 'n (p\*q):' (8), 'φ (n):' (9), and 'd (private key):' (10), along with 'Public Key:' (11) and 'Privat Key:' (12). Below this is the encryption section: 'Input Pesan:' with a 'Masukkan plaintext:' field (13) and a 'Proses' button (14). The bottom section displays results: 'Nilai Bit:' (15), 'Ciphertext:' (16), 'Waktu Enkripsi:' (17), 'Plaintext:' (18), and 'Waktu Dekripsi:' (19). At the very bottom are 'Reset' (20) and 'Keluar' (21) buttons.

Gambar 3. 20 Perancangan halaman algoritma RSA

Pada Gambar 3.20 menjelaskan beberapa komponen yang digunakan dengan keterangan meliputi :

1. *Label* berfungsi untuk mempresentasikan judul penelitian.
2. *Pixture Box* berfungsi untuk mempresentasikan logo universitas.
3. *Label* berfungsi untuk mempresentasikan nama algoritma.
4. *Textbox* berfungsi untuk mempresentasikan nilai p (prima).
5. *Textbox* berfungsi untuk menpresentasikan nilai q (prima).
6. *Textbox* berfungsi untuk mempresentasikan nilai e (*public key*).
7. *Button* berfungsi untuk memproses pembuatan kunci.
8. *Textbox* berfungsi untuk mengeluarkan nilai n (  $p \cdot q$  ).
9. *Textbox* berfungsi untuk mengeluarkan nilai  $\phi(n)$ .
10. *Textbox* berfungsi untuk mengeluarkan nilai d (*private key*).

11. *Textbox* berfungsi untuk mengeluarkan nilai pasangan kunci *public key*.
12. *Textbox* berfungsi untuk mengeluarkan nilai pasangan kunci *private key*.
13. *Textbox* berfungsi untuk menampilkan *plaintext* yang diinputkan oleh user.
14. *Button* berfungsi untuk memproses enkripsi dekripsi.
15. *Textbox* berfungsi untuk mengeluarkan nilai bit.
16. *Textbox* berfungsi untuk mengeluarkan *ciphertext* yang telah diproses.
17. *Textbox* berfungsi untuk mengeluarkan waktu enkripsi.
18. *Textbox* berfungsi untuk mengeluarkan *plaintext* yang telah diproses.
19. *Textbox* berfungsi untuk mengeluarkan waktu dekripsi.
20. *Button* berfungsi untuk mengembalikan keadaan seperti semula.
21. *Button* berfungsi untuk keluar dari sitem.

### 3.5.3 Perancangan halaman algoritma Vincent-Sathiyamoorthy

The image shows a web-based RSA algorithm interface. It features several input fields and buttons. The interface is divided into sections for key generation, message encryption, and results display. Numbered callouts (1-25) identify specific UI elements:

- 1: Title bar
- 2: Subtitle
- 3: Instructional text box
- 4: Input field for p (prima)
- 5: Input field for q (prima)
- 6: Input field for r (prima)
- 7: 'Bangkitkan' button
- 8: Input field for s
- 9: Input field for e
- 10: Input field for t
- 11: 'Proses' button
- 12: Input field for n
- 13: Input field for u
- 14: Input field for d
- 15: Output field for Public Key
- 16: Output field for Privat Key
- 17: Input field for 'Masukkan plaintext'
- 18: 'Proses' button for encryption
- 19: Output field for 'Nilai Bit'
- 20: Output field for 'Ciphertext'
- 21: Output field for 'Waktu Enkripsi'
- 22: Output field for 'Plaintext'
- 23: Output field for 'Waktu Dekripsi'
- 24: 'Reset' button
- 25: 'Keluar' button

Gambar 3. 21 Perancangan halaman algoritma RSA

Pada Gambar 3.21 menjelaskan beberapa komponen yang digunakan dengan keterangan meliputi :

1. *Label* berfungsi untuk mempresentasikan judul penelitian.
2. *Pixture Box* berfungsi untuk mempresentasikan logo universitas.
3. *Label* berfungsi untuk mempresemtasikan nama algoritma.
4. *Textbox* berfungsi untuk mempresentasikan nilai p (prima).
5. *Textbox* berfungsi untuk mempresentasikan nilai q (prima).
6. *Textbox* berfungsi untuk mempresentasikan nilai r (prima).
7. *Button* berfungsi untuk membangkitkan bilangan prima.
8. *Textbox* berfungsi untuk mengeluarkan nilai s.
9. *Textbox* berfungsi untuk mengeluarkan nilai e.
10. *Textbox* berfungsi untuk mengeluarkan nilai t.
11. *Button* berfungsi untuk memproses pembuatan kunci.
12. *Textbox* berfungsi untuk mengeluarkan nilai n.
13. *Textbox* berfungsi untuk mengeluarkan nilai u.
14. *Textbox* berfungsi untuk mengeluarkan nilai d.
15. *Textbox* berfungsi untuk mengeluarkan nilai pasangan kunci *public key*.
16. *Textbox* berfungsi untuk mengeluarkan nilai pasangan kunci *private key*.
17. *Textbox* berfungsi untuk menampilkan *plaintext* yang diinputkan oleh *user*.
18. *Button* berfungsi untuk proses enkripsi dekripsi.
19. *Textbox* berfungsi untuk mengeluarkan nilai bit.
20. *Textbox* berfungsi untuk mengeluarkan *ciphertext* yang telah diproses.
21. *Textbox* berfungsi untuk mengeluarkan waktu enkripsi.
22. *Textbox* berfungsi untuk mengeluarkan *plaintext* yang telah diproses.
23. *Textbox* berfungsi untuk mengeluarkan waktu dekripsi.
24. *Button* berfungsi untuk mengembalikan keadaan seperti semula.
25. *Button* berfungsi untuk keluar dari sitem.

## BAB 4

### IMPLEMENTASI DAN PENGUJIAN

Bab ini menjelaskan langkah-langkah detail yang diambil untuk mengimplementasikan sistem atau model yang telah dirancang sebelumnya, termasuk teknologi dan alat yang digunakan. Selain itu, bab ini juga menguraikan metode pengujian yang diterapkan untuk memastikan bahwa sistem atau model berfungsi sesuai dengan spesifikasi yang diharapkan.

#### 4.1 Implementasi Perhitungan

##### 4.1.1 Algoritma Vincent-Sathiyamoorthy

a. Pembentukan Kunci

Asumsikan Sinta ingin menghasilkan kunci publiknya. Sinta memilih nilai  $p = 3$  dan  $q = 5$ , serta  $r = 7$ . Selanjutnya, Sinta menghitung:

- $s = qr - p$   
 $s = 5(7) - 3$   
 $s = 32$
- $e = ps + r$   
 $e = 3(32) + 7$   
 $e = 103$
- $t = p^2s + q$   
 $t = 3^2(32) + 5$   
 $t = 293$

Setelah mendapat nilai dari  $s, e$ , dan  $t$ , kemudian Sinta menghitung kunci publik dengan rumus :

- $n = \frac{et-p}{s}$
- $n = \frac{103(293)-3}{32}$



- $n = 943$

Sekarang pasangan kunci publik telah didapat yaitu (103, 943).

b. Enkripsi

Setelah Sinta mendapat pasangan kunci publik, Sinta pun ingin mengirim pesan kepada temannya Tini. Namun, pesan bersifat rahasia sehingga dia harus mengenkripsi pesan itu menjadi pesan yang tidak mudah dipahami oleh orang lain. Pesan (*plaintext*) yang ingin dia kirim kepada Tini ialah :

$$m = \text{CHINDY}$$

Sinta mengubah nilai  $m$  menjadi bentuk desimal dengan menggunakan pengkodean ASCII. Kemudian, sistem membagi pesan menjadi beberapa blok yang lebih kecil dengan menjaga agar panjang setiap blok seimbang menjadi 3 digit. Proses ini melibatkan penambahan digit tambahan (biasanya 0) jika diperlukan, karena panjang maksimal digit pada kode ASCII adalah 3 digit.

$$\begin{array}{lll} m_1 = 067 & m_3 = 073 & m_5 = 068 \\ m_2 = 072 & m_4 = 078 & m_6 = 089 \end{array}$$

Sinta mengetahui bahwa pasangan kunci publik adalah  $e = 103$  dan  $n = 943$ . Dengan informasi ini, Sinta dapat melakukan enkripsi pada tiap-tiap blok *plaintext* sebagai berikut:

$$\begin{array}{ll} c_1 = 67 \times 103 \bmod 943 = 300 & c_2 = 72 \times 103 \bmod 943 = 815 \\ c_3 = 73 \times 103 \bmod 943 = 918 & c_4 = 78 \times 103 \bmod 943 = 490 \\ c_5 = 68 \times 103 \bmod 943 = 403 & c_6 = 89 \times 103 \bmod 943 = 680 \end{array}$$

Setelah melewati proses perhitungan di atas, maka *ciphertext* yang di dapat ialah :

$$C = 300\ 815\ 918\ 490\ 403\ 680$$

Kemudian, *ciphertext* di atas akan dikirim Sinta kepada Tini.

c. Dekripsi

Tini selaku temannya Sinta, telah mendapat pesan yang dikirim oleh Sinta. Namun, Tini memerlukan kunci privat untuk mendapatkan pesan yang asli.

Sehingga, langkah awal yang dilakukan Tini ialah membentuk kunci privat. Untuk mendapat kunci privat, Tini harus memecahkan persamaan berikut ini:

$$P * u \pmod{n} \equiv 1$$

$$3 * u \equiv 1 \pmod{n}$$

Dengan menguji nilai-nilai  $u$  mulai dari 1, 2, 3, ..., ditemukan bahwa nilai yang memenuhi adalah 629.

$$u = 1 \quad (3,1) \pmod{943} = 3$$

$$u = 2 \quad (3,2) \pmod{943} = 6$$

$$u = 3 \quad (3,3) \pmod{943} = 9$$

.

.

.

$$u = 629 \quad (3,629) \pmod{943} = 1$$

Setelah mendapat nilai  $u$ , langkah berikutnya adalah mencari nilai  $d$  dengan menggunakan rumus berikut :

$$d = u \times t$$

$$d = 629 \times 293$$

$$d = 184297$$

Setelah mendapat nilai  $u$ , akhirnya pembentukan kunci privat Tini telah selesai dan Tini mendapatkan pasangan kunci privat yaitu 184297, 943. Kemudian, langkah terakhir yang harus dilakukan Tini untuk mendapatkan pesan asli dari *ciphertext* yang dikirim oleh Sinta ialah dengan mengubahnya menjadi *plaintext* (teks biasa). Dengan menggunakan pasangan kunci privat, *ciphertext* yang sudah dibagi menjadi blok-blok *cipher* diubah kembali menjadi *plaintext* : CHINDY.

$$C = 300\ 815\ 918\ 490\ 403\ 680$$

$$m_1 = 300 \times 184297 \pmod{943} = 67$$

$$m_2 = 815 \times 184297 \pmod{943} = 72$$

$$m_3 = 918 \times 184297 \pmod{943} = 73$$

$$m_4 = 490 \times 184297 \pmod{943} = 78$$

$$m_5 = 403 \times 184297 \pmod{943} = 68$$

$$m_6 = 680 \times 184297 \pmod{943} = 89$$

Sehingga *plaintext* yang dihasilkan adalah  $m = \text{CHINDY}$ . Akhirnya, Tini dapat membaca dan mengerti pesan yang dikirim oleh Sinta.

#### 4.1.2 Algoritma RSA

##### a. Pembentukan Kunci

Asumsikan Siti ingin menghasilkan kunci publik dan kunci privatnya. Siti menentukan nilai  $p = 13$  serta  $q = 11$ . Kemudian, Siti mendapatkan :

$$n = 143 \text{ dan } \phi(n) = 120$$

Siti menentukan kunci publik  $e = 7$  dikarenakan  $e$  relatif prima dengan 120. Kemudian, Siti melakukan perhitungan kunci dekripsi  $d$  berdasarkan persamaan berikut :

$$7 \cdot d \equiv 1 \pmod{120}$$

Kemudian coba nilai  $d$  yang mulai dari 1, 2, 3, ..., ditemukan bahwa nilai  $d$  yang memenuhi persamaan (1) ialah 103.

$d = 1$	$(7,1) \bmod 120 = 7$
$d = 2$	$(7,2) \bmod 120 = 14$
$d = 3$	$(7,3) \bmod 120 = 21$
.	.
.	.
.	.
$d = 103$	$(7,103) \bmod 120 = 1$

Kunci privat dipakai untuk melakukan proses dekripsi pesan dan harus bersifat rahasia. Oleh karena itu, mengakibatkan terbentuknya pasangan kunci sebagai berikut :

- Kunci publik yaitu  $e = 7$ ,  $n = 143$
- Kunci privat yaitu  $d = 103$ ,  $n = 143$

##### b. Enkripsi

Misal Tono ingin mengirimkan pesan kepada Siti. Pesan yang akan dikirimkan kepada Siti ialah :

$$m = \text{CHINDY}$$

Tono mengubah nilai  $m$  menjadi desimal menggunakan pengkodean ASCII. Sistem kemudian membagi  $m$  menjadi beberapa blok yang lebih kecil dengan membuat panjang setiap blok sama dengan tiga angka. Proses ini melibatkan penambahan digit tambahan (biasanya 0) jika diperlukan, karena panjang maksimal digit pada kode ASCII adalah 3 digit.

$$\begin{array}{lll} m_1 = 067 & m_3 = 073 & m_5 = 068 \\ m_2 = 072 & m_4 = 078 & m_6 = 089 \end{array}$$

Nilai-nilai ini tetap berada dalam rentang  $[0, 143-1]$  untuk memastikan transformasinya tetap berlangsung secara unik. Tono mengetahui bahwa kunci publik ialah  $e = 7$  dan  $n = 143$ . Dengan informasi ini, Tono dapat melakukan enkripsi pada setiap blok- blok *plaintext* (teks biasa) seperti berikut:

$$\begin{array}{ll} c_1 = 67^7 \bmod 143 = 123 & c_2 = 72^7 \bmod 143 = 56 \\ c_3 = 73^7 \bmod 143 = 116 & c_4 = 78^7 \bmod 143 = 99 \\ c_5 = 68^7 \bmod 143 = 95 & c_6 = 89^7 \bmod 143 = 55 \end{array}$$

Untuk membantu sistem memecah *ciphertext* menjadi beberapa blok yang mewakili setiap karakter, sistem menambahkan digit tambahan ke blok *cipher*. Langkah ini dilakukan sedemikian rupa sehingga setiap balok mempunyai panjang yang seragam, yaitu 3 digit sesuai spesifikasi yang diberikan. Jadi *ciphertext* yang di dapat ialah :

$$c = 123\ 056\ 116\ 099\ 095\ 055$$

#### c. Dekripsi

Menggunakan kunci privat  $d = 103$ , pesan terenkripsi yang sudah dipisahkan menjadi blok-blok *cipher* diubah kembali menjadi teks biasa: CHINDY.

$$\begin{array}{ll} c = 123\ 056\ 116\ 099\ 095\ 055 \\ m_1 = 123^{103} \bmod 143 = 67 & m_2 = 56^{103} \bmod 143 = 72 \\ m_3 = 116^{103} \bmod 143 = 73 & m_4 = 099^{103} \bmod 143 = 78 \\ m_5 = 095^{103} \bmod 143 = 68 & m_6 = 055^{103} \bmod 143 = 89 \end{array}$$

Sehingga *plaintext* yang dihasilkan  $m = \text{CHINDY}$ .

## 4.2 Implementasi sistem

Setelah menganalisis dan merancang sistem, tahap berikutnya adalah mengimplementasikannya dan melaksanakan pengujian memakai bahasa pemrograman *Python* dengan menggunakan perangkat lunak *Visual Studio Code*. Pengaplikasian dilakukan sesuai dengan hasil evaluasi dan perencanaan yang telah dipersiapkan sebelumnya.

### 4.2.1 *Generate Bilangan Prima*



Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi  
Pada Algoritma RSA dan Vincent Sathiyamoorthy

GENERATE BILANGAN PRIMA

Input Jumlah Bit :

Jumlah Bit:

p :


q :

*Gambar 4. 1 Tampilan Sistem Generate Bilangan Prima*

Gambar 4.1 menunjukkan tampilan sistem *generate* bilangan prima secara acak. Pengguna akan memasukkan jumlah bit bilangan yang akan dibangkitkan. Kemudian, pengguna menekan tombol bangkitkan, maka sistem akan menampilkan nilai dari  $p$  dan  $q$  yang dimana keduanya adalah bilangan prima acak.

#### 4.2.2 Algoritma Vincent-Sathiyamoorthy

**Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi  
Pada Algoritma RSA dan Vincent Sathiyamoorthy**



**ALGORITMA VINCENT-SATHIYAMOORTHY**

**Input Kunci :**

p (prima) :  s :  n :  u :  d :

q (prima) :  Bangkitkan e :  Proses Public Key :

r (prima) :  t :  Private Key :

**Input Pesan :**

Masukan Plaintext

Nilai Bit :

Ciphertext :

Waktu Enkripsi :

Plaintext :

Waktu Dekripsi :

*Gambar 4. 2 Tampilan Sistem Algoritma Vincent-Sathiyamoorthy*

Gambar 4.2 menunjukkan tampilan sistem Vincent-Sathiyamoorthy yang dapat melakukan enkripsi dan dekripsi. Namun, sebelum melakukan proses tersebut, pengguna harus terlebih dahulu menginputkan tiga bilangan acak yang menjadi dasar pembentukan kunci publik dan kunci privat. Setelah pengguna menginputkan tiga bilangan acak, langkah selanjutnya adalah pengguna dapat memproses bilangan acak tersebut untuk mendapatkan pasangan kunci publik dan kunci privat. Setelah itu, pengguna memasukkan pesan atau *plaintext* yang ingin diproses lalu menekan tombol proses. Maka secara otomatis sistem melakukan proses enkripsi serta dekripsi sekaligus dan pesan tersebut diubah menjadi *ciphertext*. Selain itu, sistem juga akan menghasilkan waktu enkripsi dan dekripsi dari *plaintext* yang telah diproses oleh sistem.

### 4.2.3 Algoritma RSA

**Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi  
Pada Algoritma RSA dan Vincent Sathiyamoorthy**



**ALGORITMA RSA**

**Input Kunci :**

p (prima) :	<input type="text"/>	n ( $p * q$ ) :	<input type="text"/>	Public Key :	<input type="text"/>
q (prima) :	<input type="text"/>	$\phi(n)$ :	<input type="text"/>	Private Key :	<input type="text"/>
e (public key) :	<input type="text"/>	d (private key) :	<input type="text"/>		

**Input Pesan :**

Masukan Plaintext :

Nilai Bit :

Ciphertext :

Waktu Enkripsi :

Plaintext :

Waktu Dekripsi :

*Gambar 4. 3 Tampilan Sistem Algoritma RSA*

Gambar 4.3 menunjukkan tampilan sistem RSA yang dapat melakukan enkripsi dan dekripsi. Namun, sebelum melakukan proses tersebut, pengguna harus terlebih dahulu menginputkan dua bilangan acak yang menjadi dasar pembentukan kunci publik dan kunci privat. Setelah pengguna menginputkan dua bilangan acak, kemudian langkah selanjutnya adalah pengguna dapat memproses bilangan acak tersebut untuk mendapatkan pasangan kunci publik dan kunci privat. Setelah itu, pengguna memasukkan pesan atau *plaintext* yang ingin dienkripsi lalu menekan tombol proses. Maka secara otomatis sistem melakukan proses enkripsi serta dekripsi sekaligus dan pesan tersebut diubah menjadi *ciphertext*. Selain itu, sistem juga akan menghasilkan waktu enkripsi dan dekripsi dari *plaintext* yang telah diproses oleh sistem.

### 4.3 Pengujian Sistem

Pengujian sistem dilaksanakan untuk memperoleh hasil dari kesesuaian sistem dan spesifikasi perancangan algoritma Vincent-Sathiyamoorthy dan algoritma RSA. Pengujian sistem dilakukan uji coba dengan menggunakan karakter yang akan diubah ke dalam kode *ASCII*. Pada pengujian akan dilakukan uji coba dengan membandingkan kedua algoritma berdasarkan persamaan nilai  $p$  dan  $q$  yang sama dan perbedaan panjang *plaintext*. Dan pada akhirnya, kita mencari nilai rata-rata dari ketiga percobaan tersebut untuk dibandingkan.

#### 4.3.1 Perbandingan Waktu Pembuatan Kunci Berdasarkan Nilai $p$ dan $q$ yang sama

Pada bagian ini, dilakukan uji coba dengan membandingkan waktu pembuatan kunci berdasarkan nilai bit yang berbeda sebanyak 5 kali. Adapun nilai bit yang digunakan adalah sebagai berikut :

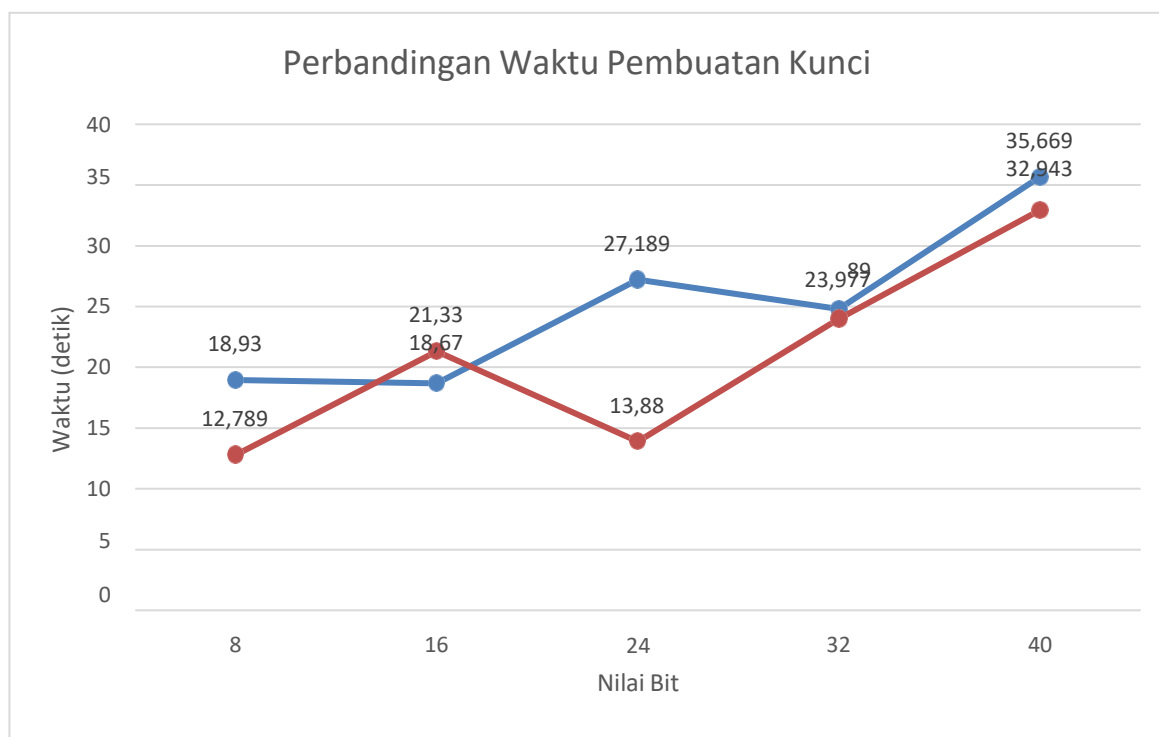
- 8 bit :
  - $p = 193$
  - $q = 181$
- 16 bit :
  - $p = 49681$
  - $q = 30389$
- 24 bit :
  - $p = 3706393$
  - $9825443$
- 32 bit :
  - $p = 522594971$
  - $q = 3367303081$
- 40 bit :
  - $p = 1004833636861$
  - $q = 535204005157$



Tabel 4. 1 Waktu Pembuatan Kunci Algoritma VS dan RSA

Nilai Bit	Waktu VS ( x 10 <sup>-5</sup> )	Waktu RSA ( x 10 <sup>-5</sup> )
8	18.930	12.789
16	18.670	21.330
24	27.189	13.880
32	24.789	23.977
40	35.669	32.943

Pada tabel 4.1, menyajikan informasi mengenai waktu pembuatan kunci algoritma Vincent- Sathiyamoorthy dan RSA dalam satuan detik dengan nilai p dan q yang sama. Namun, untuk mendapatkan perbedaan yang signifikan pada grafik, waktu tersebut diubah menjadi ( x 10<sup>-5</sup> ). Untuk mendapatkan pemahaman yang lebih terperinci, kita bisa mengamati Gambar 4.4.



Gambar 4. 4 Grafik waktu pembuatan kunci algoritma VS dan RSA

Pada Gambar 4.4, kita dapat melihat bahwa garis yang berwarna merah menunjukkan waktu algoritma RSA , sedangkan garis yang berwarna biru menunjukkan waktu algoritma Vincent- Sathiyamoorthy.

#### 4.3.2 Perbandingan Rata-rata Waktu Enkripsi Dekripsi Berdasarkan Persamaan Nilai $p$ dan $q$ yang sama dan Perbedaan Panjang Plaintext

##### a. Percobaan 1

Pada bagian ini, dilakukan uji coba dengan menggunakan nilai  $p = 71366164147$  dan  $q = 827451605447$  pada kedua algoritma. Selain itu, akan dilakukan uji coba sebanyak 25 kali dengan panjang *plaintext* yang berbeda. Adapun isi dari *plaintext* tersebut dapat kita lihat di dokumen *karakter.docx*.

- Algoritma Vincent-Sathiyamoorthy

Tabel 4. 2 Waktu Enkripsi dan Dekripsi Algoritma Vincent-Sathiyamoorthy

Panjang Plaintext	Waktu Enkripsi VS	Waktu Dekripsi VS
200	0,00006160000000225144	0,0001464999999960528
400	0,00009560000000874425	0,0003343999999856351
600	0,0001275999999900629	0,00019720000000233995
800	0,00015550000000530417	0,0002588000000045554
1000	0,00019619999999775928	0,00032929999999709025
1200	0,0002459999999970819	0,0004975999999956571
1400	0,00038760000000337413	0,0005419000000017604
1600	0,0001547999999971239	0,0005158999999963498
1800	0,0005894000000026267	0,0006999000000007527
2000	0,00019809999999864658	0,0006277000000025623
2200	0,00020849999999938973	0,0007074000000031333
2400	0,00022449999999452075	0,0007758999999936123
2600	0,000594899999995846	0,000817200000001844
2800	0,0003888999999972498	0,000886500000000123
3000	0,0002919000000005667	0,000147000000027106
3200	0,00035579999999413303	0,0011216000000047188
3400	0,0003484000000071319	0,0010474999999985357
3600	0,000345999999997908526	0,00111019999999709733
3800	0,00038119999999253196	0,0014644999999973152
4000	0,00033489999999858355	0,0012311999999994327

4200	0,0003589999999746851	0,001610800000008794
4400	0,0003298999999969965	0,0011016999999924337
4600	0,0006155000000376276	0,0013811000000032436
4800	0,00041650000000004184	0,0015536999999952172
5000	0,0004233999999954108	0,0015010999999987007

Pada tabel 4.2, menyajikan informasi mengenai waktu enkripsi *dan* dekripsi algoritma Vincent- Sathiyamoorthy dalam satuan detik dengan nilai p dan q yang sama dan panjang *plaintext* yang berbeda-beda. Adapun waktu enkripsi *dan* dekripsi yang dibutuhkan adalah berkisar kurang dari 1 detik.

- Algoritma RSA

Tabel 4. 3 Waktu Enkripsi dan Dekripsi Algoritma RSA

Panjang Plaintext	Waktu Enkripsi RSA	Waktu Dekripsi RSA
200	0,0023063000000149714	0,003558400000002848
400	0,004544400000000337	0,004485299999998915
600	0,0052626000000017825	0,007567700000002731
800	0,008057899999997176	0,009378900000001522
1000	0,005558499999999356	0,01154300000000319
1200	0,0069452000000040925	0,01342449999999284
1400	0,007590699999994399	0,016019900000003418
1600	0,00834549999999723	0,0180460999999994
1800	0,008607499999996548	0,020174799999992388
2000	0,010771999999974469	0,02288200000000792
2200	0,01076199999999716	0,024490299999911258
2400	0,011543999999958032	0,026904399999921225
2600	0,012703900000001953	0,029168799999979456
2800	0,0132347000000111843	0,031372599999940576
3000	0,014244299999973009	0,03372709999985091
3200	0,015198099999906844	0,03575070000003352
3400	0,0164783000000017043	0,0382548000000231626
3600	0,0170419000000002386	0,040562900000000473

3800	0,017890500000000003	0,042543999999999947
4000	0,0190707000000000033	0,044513500000000076
4200	0,01985920000000006183	0,047097000000000083
4400	0,0171430000000000435	0,040343299999998931
4600	0,022094199999997954	0,062742100000000124
4800	0,0226090000000002766	0,053413200000000138
5000	0,0237373000000000516	0,057665499999998815

Pada tabel 4.3, menyajikan informasi mengenai waktu enkripsi *dan* dekripsi algoritma RSA dalam satuan detik dengan nilai p dan q yang sama dan panjang *plaintext* yang berbeda-beda. Adapun waktu enkripsi dan dekripsi yang dibutuhkan adalah berkisar kurang dari 1 detik.

b. Percobaan 2

Pada bagian ini, dilakukan uji coba dengan menggunakan nilai  $p = 820466790257$  dan  $q = 1062850237531$  pada kedua algoritma. Selain itu, akan dilakukan uji coba sebanyak 25 kali dengan panjang *plaintext* yang berbeda. Adapun isi dari *plaintext* tersebut dapat kita lihat di dokumen *karakter.docs*.

- Algoritma Vincent-Sathiyamoorthy

Tabel 4. 4 Waktu Enkripsi dan Dekripsi Algoritma Vincent-Sathiyamoorthy

Panjang Plaintext	Waktu Enkripsi VS	Waktu Dekripsi VS
200	0,0000793000000101074	0,00018049999999902866
400	0,00010559999998349667	0,000134500000000148066
600	0,00012870000000475557	0,00023249999998142812
800	0,0001522000000022672	0,0002620000000206346
1000	0,00016279999999824213	0,00034180000000010579
1200	0,00015439999992850062	0,00040979999994306127
1400	0,00014709999993556266	0,000460699999962344
1600	0,00018490000002202578	0,0005418000000645407

1800	0,00018999999997504347	0,0005962000000181433
2000	0,00018990000012308883	0,000653199999987919
2200	0,00022439999997914128	0,0007343999998283834
2400	0,00022330000001602457	0,0007966999999098334
2600	0,00023549999991701042	0,0008666999999604741
2800	0,00024089999988063937	0,0009116000001085922
3000	0,00031120000016926497	0,001017000000047119
3200	0,0002953000000616157	0,0010301999998318934
3400	0,00029380000000855944	0,001081099999964863
3600	0,0004928000000745669	0,001158499999974083
3800	0,00035260000004200265	0,0012164999998276471
4000	0,0003706000002239307	0,001280100000258244
4200	0,0003764999996747065	0,0013357000002542918
4400	0,00031950000038705184	0,0013506000000234053
4600	0,00042619999976523104	0,0014550000000781438
4800	0,0008043000002544431	0,001522699999895849
5000	0,00047360000007756753	0,00157870000020921

Pada tabel 4.4, menyajikan informasi mengenai waktu enkripsi dan dekripsi algoritma Vincent- Sathiyamoorthy dalam satuan detik dengan nilai  $p$  dan  $q$  yang sama dan panjang *plaintext* yang berbeda-beda. Adapun waktu enkripsi dan dekripsi yang dibutuhkan adalah berkisar kurang dari 1 detik.

- . Algoritma RSA

Tabel 4. 5 Waktu Enkripsi dan Dekripsi Algoritma RSA

Panjang Plaintext	Waktu Enkripsi RSA	Waktu Dekripsi RSA
200	0,0031673000000012053	0,005772000000000332
400	0,005586999999991349	0,10982299999994893
600	0,006838900000019521	0,011365900000015472
800	0,009211699999980283	0,01027559999999994
1000	0,01088090000007469	0,027492800000004536
1200	0,014839100000003214	0,015521299999996074
1400	0,014879999999998006	0,018099599999999327
1600	0,01780730000000119	0,02752290000000812
1800	0,018357999999992103	0,022599999999997067
2000	0,019493299999993496	0,025256799999993973
2200	0,016768200000001343	0,028158099999998853
2400	0,021933500000002937	0,0316812000000013
2600	0,017226000000000852	0,03251690000000451
2800	0,0168119999999987393	0,03526110000001381
3000	0,016425900000001548	0,0380726999999986526
3200	0,016848799999991115	0,03987250000000131
3400	0,017563100000074883	0,042803199999998028
3600	0,036227899999997167	0,04568249999999807
3800	0,036014999999990665	0,06561820000001717
4000	0,026658200000156285	0,04991790000008223
4200	0,03103940000005423	0,052523899999986917
4400	0,01862270000015087	0,04510529999993196
4600	0,025742199999967852	0,05795880000005127
4800	0,046644999999989857	0,06072960000005878
5000	0,03434360000005654	0,07438660000002528

Pada tabel 4.5, menyajikan informasi mengenai waktu enkripsi dan dekripsi algoritma RSA dalam satuan detik dengan nilai  $p$  dan  $q$  yang sama dan panjang *plaintext* yang berbeda-beda. Adapun waktu enkripsi dan dekripsi yang dibutuhkan adalah berkisar kurang dari 1 detik.

## c. Percobaan 3

Pada bagian ini, dilakukan uji coba dengan menggunakan nilai  $p = 260131322447$  dan  $q = 155403985739$  pada kedua algoritma. Selain itu, akan dilakukan uji coba sebanyak 25 kali dengan panjang *plaintext* yang berbeda. Adapun isi dari *plaintext* tersebut dapat kita lihat di dokumen *karakter.docs*.

- Algoritma Vincent-Sathiyamoorthy

Tabel 4. 6 Waktu Enkripsi dan Dekripsi Algoritma Vincent-Sathiyamoorthy

Panjang Plaintext	Waktu Enkripsi VS	Waktu Dekripsi VS
200	0,00000583000003470559	0,90001362999999514636
400	0,00009699999998247222	0,00015880000000834116
600	0,00013479999995524485	0,00019450000002052548
800	0,00015989999997145787	0,000389000000004105095
1000	0,00017079999997804407	0,00038970000002791494
1200	0,00015929999995023536	0,00039470000001529115
1400	0,00014659999999366846	0,0004590000000916916
1600	0,0001450999982692534	0,000513199998866377
1800	0,00029390000008788775	0,0005848999999216176
2000	0,00019100000008620555	0,0006614000001263776
2200	0,00020649999987654155	0,000720500000170432
2400	0,0002214999995235703	0,0007635999998001353
2600	0,0002300000005561742	0,000835699999697681
2800	0,00028840000049967784	0,000894899999663165
3000	0,0002834999995684484	0,0010044000000561937
3200	0,00030609999976149993	0,0009958000000551692
3400	0,00031630000012228265	0,0010505000000193832
3600	0,00031799999942450086	0,001120699999773933
3800	0,00048179999976127874	0,0011713000003510388
4000	0,0003483000000414904	0,0012248000002728077
4200	0,00036870000076305587	0,001680200000097905

4400	0,0003326999994897051	0,0011279999998805579
4600	0,00039470000004700385	0,0014242999995985883
4800	0,00041110000074695563	0,0014956999993955833
5000	0,00042209999992337544	0,001534199999696284

Pada tabel 4.6, menyajikan informasi mengenai waktu enkripsi dan dekripsi algoritma Vincent- Sathiyamoorthy dalam satuan detik dengan nilai  $p$  dan  $q$  yang sama dan panjang *plaintext* yang berbeda-beda. Adapun waktu enkripsi dan dekripsi yang dibutuhkan adalah berkisar kurang dari 1 detik.

- Algoritma RSA

Tabel 4. 7 Waktu Enkripsi dan Dekripsi Algoritma RSA

Panjang Plaintext	Waktu <i>Enkripsi</i> RSA	Waktu <i>Dekripsi</i> RSA
200	0,003045399999977555	0,00469249999997324
400	0,004539599999986876	0,009146500000014157
600	0,006724399999995967	0,011758100000008653
800	0,008223600000007991	0,015785600000015165
1000	0,00995770000002949	0,01960619999998353
1200	0,01138180000000947	0,020415600000092127
1400	0,01449929999997106	0,016293400000002123
1600	0,015396700000003705	0,022368000000000166
1800	0,016411500000003798	0,01753880000001118
2000	0,0094351000000187922	0,035702800000024126
2200	0,010282899999992878	0,038685400000019854
2400	0,010798700000009376	0,024115199999982906
2600	0,023720400000001973	0,02524930000004133
2800	0,014441400000009708	0,027132899999969595
3000	0,0175659000000223433	0,0296387000000777
3200	0,014373999999861553	0,03110030000016195
3400	0,024727499999926295	0,033106100000168226
3600	0,020838800000092306	0,046345599999951
3800	0,017058199999837598	0,03738219999991088
4000	0,01796199999989767	0,03939970000010362



4200	0,018780399999968722	0,04094370000007075
4400	0,016215900000133843	0,05066820000001826
4600	0,022192199999835793	0,07522999999991953
4800	0,021474099999977625	0,4666080000015427
5000	0,024912400000175694	0,06284760000016831

Pada tabel 4.7, menyajikan informasi mengenai waktu enkripsi dan dekripsi algoritma RSA dalam satuan detik dengan nilai p dan q yang sama dan panjang *plaintext* yang berbeda-beda. Adapun waktu enkripsi dan dekripsi yang dibutuhkan adalah berkisar kurang dari 1 detik.

d. Perbandingan Rata-Rata

Pada bagian ini, dilakukan perbandingan waktu enkripsi dan dekripsi kedua algoritma dengan menggunakan rata-rata waktu dari tiga kali percobaan di atas pada kedua algoritma tersebut.

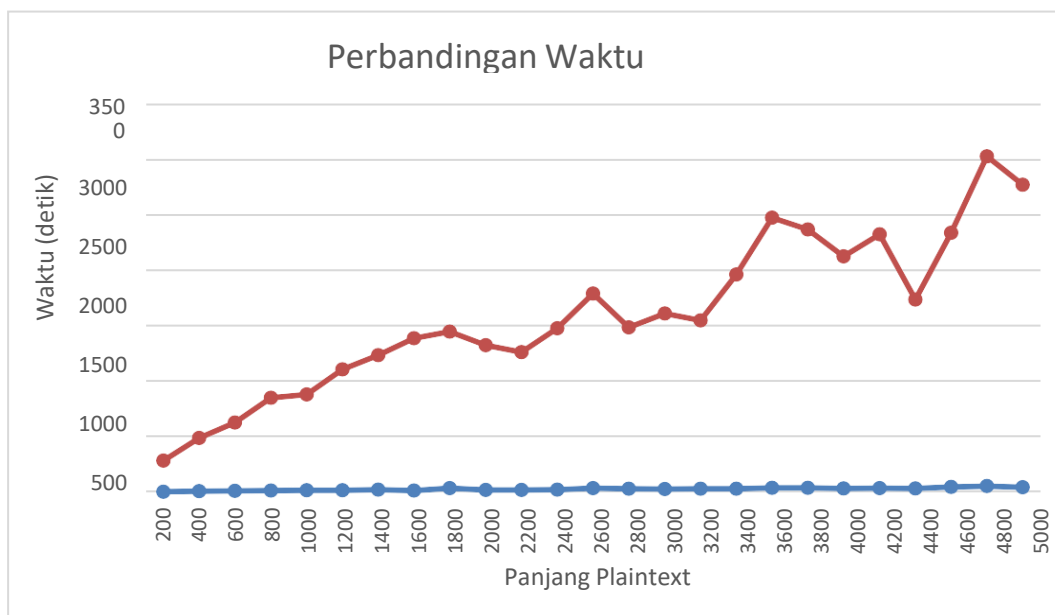
- Enkripsi

Tabel 4. 8 Waktu Enkripsi Algoritma Vincent-Sathiyamoorthy dan RSA

Panjang Plaintext	Waktu <i>Enkripsi</i> VS ( $\times 10^{-5}$ )	Waktu <i>Enkripsi</i> RSA ( $\times 10^{-5}$ )
200	4,8910000015688146	283,9666666671311
400	9,939999999157105	489,0333333326187
600	13,036666666726895	627,5300000005757
800	15,58666666596764	849,7733333328483
1000	17,659999999134848	879,9033333367845
1200	18,656666662527263	1105,5366666700669
1400	22,709999997753508	1232,3333333329837
1600	16,15999994294677	1384,9833333334042
1800	35,77666666885193	1445,8999999997483
2000	19,30000000652534	1323,3466666662252
2200	21,313333328502417	1260,4366666664646
2400	22,30999998447052	1475,8733333323448
2600	35,34666668230102	1788,3433333340786
2800	30,60666667925223	1482,9366666702981
3000	29,55333332460934	1607,8699999998964
3200	31,906666660574956	1547,3633333325317

3400	31,950000004599133	1958,9633333339407
3600	38,5599999826051	2470,2866666965705
3800	40,51999999319378	2365,4566666581427
4000	35,12666667546682	2123,0300000018094
4200	36,80666668041492	2322,6333333343046
4400	32,73666666245845	1732,7200000096354
4600	47,87999999499542	2334,2866666600532
4800	54,39666670004802	3024,2699999959655
5000	43,96999999878456	2766,4433333410916

Pada tabel 4.8, menyajikan informasi mengenai perbandingan waktu enkripsi algoritma Vincent- Sathiyamoorthy dan RSA dalam satuan detik dengan mencari rata-rata waktu kedua algoritma tersebut dari 3 kali percobaan. Adapun waktu enkripsi yang dibutuhkan adalah berkisar kurang dari 1 detik. Namun, untuk mendapatkan perbedaan yang signifikan pada grafik, waktu tersebut diubah menjadi ( $\times 10^{-5}$ ). Untuk mendapatkan pemahaman yang lebih terperinci, kita bisa mengamati Gambar 4.5.



Gambar 4. 5 Grafik Perbandingan Waktu Enkripsi Algoritma VS dan RSA

Pada Gambar 4.5, kita dapat melihat bahwa garis yang berwarna merah menunjukkan waktu algoritma RSA, sedangkan garis yang berwarna biru menunjukkan waktu algoritma Vincent- Sathiyamoorthy.

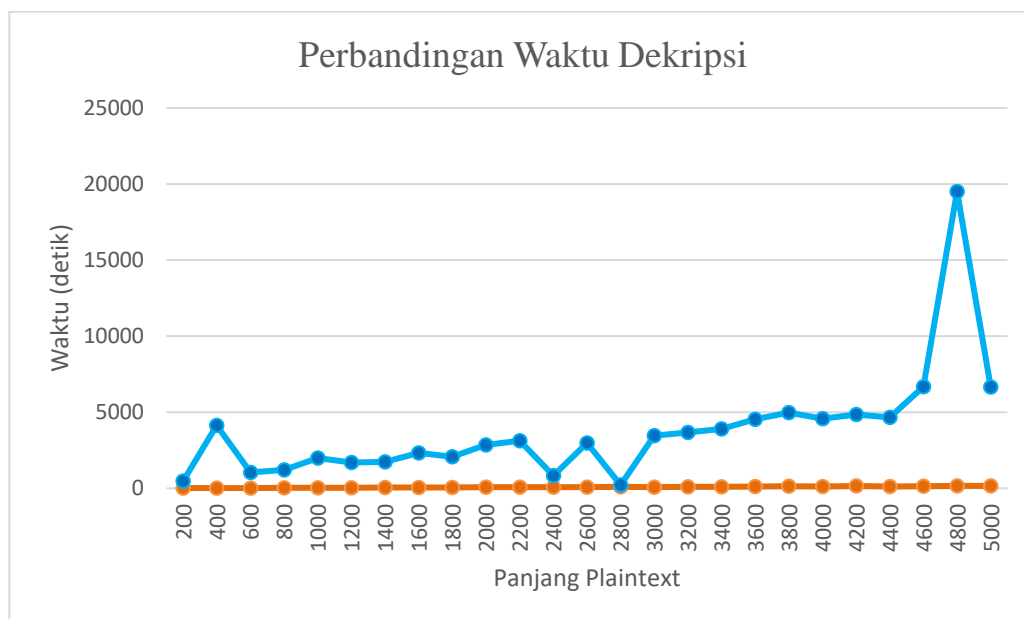
- Dekripsi

Tabel 4. 9 Waktu Dekripsi Algoritma Vincent-Sathiyamoorthy dan RSA

Panjang Plaintext	Waktu <i>Dekripsi</i> VS ( $\times 10^{-5}$ )	Waktu <i>Dekripsi</i> RSA ( $\times 10^{-5}$ )
200	15,443333331669918	467,4300000000168
400	20,923333333181895	4115,159999998733
600	20,806666666809784	1023,056666667562
800	30,326666668874697	1181,336666667221
1000	35,36000000086877	1954,733333333536
1200	43,403333331800315	1645,3800000029162
1400	48,72000000185987	1680,430000000162
1600	52,36333329757558	2264,566666666923
1800	62,69999999801712	2010,4533333333546
2000	64,74333333722863	2794,7200000032434
2200	72,07666666673163	3044,4599999973215
2400	77,8733333234527	756,6933333301808
2600	83,98666665527799	2897,8333333341766
2800	89,76666665905898	125,5533333307994
3000	72,28000000434729	3381,2833333305044
3200	104,91999999639272	3557,4500000166154
3400	105,96999999942607	3805,470000012671
3600	112,97999999063297	4419,69999999993
3800	128,4100000058667	4851,479999997584
4000	124,53666668434948	4461,036666672887
4200	154,22333334242921	4685,486666664692
4400	119,3433333298799	4537,226666664651
4600	142,01333332266586	6531,029999999068
4800	152,40333330955498	19358,36000005343
5000	153,7999999968065	6496,656666672725

Pada tabel 4.9, menyajikan informasi mengenai perbandingan waktu dekripsi algoritma Vincent- Sathiyamoorthy dan RSA dalam satuan detik dengan mencari rata-rata waktu kedua algoritma tersebut dari 3 kali percobaan. Adapun waktu dekripsi yang dibutuhkan adalah berkisar kurang dari 1 detik. Namun, untuk

mendapatkan perbedaan yang signifikan pada grafik, waktu tersebut diubah menjadi ( $\times 10^{-5}$ ). Untuk mendapatkan pemahaman yang lebih terperinci, kita bisa mengamati Gambar 4.6.



Gambar 4. 6 Grafik Perbandingan Waktu Dekripsi Algoritma VS dan RSA

Pada Gambar 4.6, kita dapat melihat bahwa garis yang berwarna biru menunjukkan waktu algoritma RSA, sedangkan garis yang berwarna *orange* menunjukkan waktu algoritma Vincent- Sathiyamoorthy.

## 4.4 Perhitungan Kompleksitas Algoritma

### 4.4.1 Algoritma Vincent-Sathiyamoorthy

#### a. Generate Key

Tabel 4. 10 Kompleksitas Waktu Proses Pembuatan Kunci Algoritma Vincent-Sathiyamoorthy

Statement	s/e	frequency	Total steps
def generate_keypair(self, p, q):			
s = int(self.txtS.text())	C1	1	C1
e = int(self.txtE.text())	C2	1	C2
t = int(self.txtT.text())	C3	1	C3
n = (e * t - p) // s	C4	1	C4
u = mod_inverse(p, n)	C5	log n	C5 log n
d = u * t	C6	1	C6
self.txtN.setText(f"{n}")	C7	1	C7
self.txtU.setText(f"{u}")	C7	1	C7
self.txtD.setText(f"{d}")	C7	1	C7
public_key = (e, n)	C8	1	C8
private_key = (d, n)	C9	1	C9
return public_key, private_key	C10	1	C10
Total			$T(n) = \Theta(\log n)$

Pada tabel 4.10, menyajikan informasi mengenai perhitungan kompleksitas waktu pembuatan kunci algoritma Vincent-Sathiyamoorthy yang hasilnya adalah  $T(n) = \Theta(\log n)$ .

## b. Proses Enkripsi

Tabel 4. 11 Kompleksitas Waktu Proses Enkripsi Algoritma Vincent-Sathiyamoorthy

Statement	s/e	frequency	Total Steps
def encrypt(message, public_key):			
e, n = public_key	C1	1	C1
start_time = time.perf_counter()	C2	1	C2
cipher_text = [ord(char) * e % n for char in message]	C3	n	C3 n
encryption_time = time.perf_counter() – start_time	C4	1	C4
return cipher_text, encryption_time	C5	1	C5
Total			$T(n) = \Theta(n)$

Pada tabel 4.11, menyajikan informasi mengenai perhitungan kompleksitas waktu proses enkripsi algoritma Vincent-Sathiyamoorthy, yang hasilnya adalah  $T(n) = \Theta(n)$ .

## c. Proses Dekripsi

Tabel 4. 12 Kompleksitas Waktu Proses Dekripsi Algoritma Vincent-Sathiyamoorthy

Statement	s/e	frequency	Total Steps
def decrypt(cipher_text, private_key):			
d, n = private_key	C1	1	C1
start_time = time.perf_counter()	C2	1	C2
plain_text = [chr(char * d % n) for char in cipher_text]	C3	n	C3 n
decryption_time = time.perf_counter() – start_time	C4	1	C4
return ".join(plain_text), decryption_time	C5	1	C5
Total			$T(n) = \Theta(n)$

Pada tabel 4.12, menyajikan informasi mengenai perhitungan kompleksitas waktu proses dekripsi algoritma Vincent-Sathiyamoorthy yang hasilnya adalah  $T(n) = \Theta(n)$ .

#### 4.4.2 Algoritma RSA

##### a. Generate Key

Tabel 4. 13 Kompleksitas Waktu Pembuatan Kunci Algoritma RSA

Statement	s/e	Frequency	Total Steps
def generate_keypair(self, p, q):			
n = p * q	C1	1	C1
self.txtN.setText(f"{n}")	C2	1	C2
phi = (p - 1) * (q - 1)	C3	1	C3
self.txtTotient.setText(f"{phi}")	C2	1	C2
e = int(self.txtE.text())	C4	1	C4
if gcd(e, phi) != 1:	C5	log n	C5 log n
self.warning("Nilai e tidak relatif prima dengan phi.")	C6	1	C6
return None	C7	1	C7
d = mod_inverse(e, phi)	C8	log n	C8 log n
self.txtD.setText(f"{d}")	C2	1	C2
public_key = (e, n)	C9	1	C9
private_key = (d, n)	C10	1	C10
return public_key, private_key	C11	1	C11
Total			$T(n) = \Theta(\log n)$

Pada tabel 4.13, menyajikan informasi mengenai perhitungan kompleksitas waktu pembuatan kunci algoritma RSA, yang hasilnya adalah  $T(n) = \Theta(\log n)$ .

## b. Proses Enkripsi

Tabel 4. 14 Kompleksitas Waktu Proses Enkripsi Algoritma RSA

Statement	s/e	frequency	Total Steps
def encrypt(message, public_key):			
e, n = public_key	C1	1	C1
start_time = time.perf_counter()	C2	1	C2
cipher_text = [pow(ord(char), e, n) for char in message]	C3	n	C3 n
encryption_time = time.perf_counter() - start_time	C4	1	C4
return cipher_text, encryption_time	C5	1	C5
Total			$T(n) = \Theta(n)$

Pada tabel 4.14, menyajikan informasi mengenai perhitungan kompleksitas waktu proses enkripsi algoritma RSA, yang hasilnya adalah  $T(n) = \Theta(n)$ .

## c. Proses Dekripsi

Tabel 4. 15 Kompleksitas Waktu Proses Dekripsi Algoritma RSA

Statement	s/e	frequency	Total Steps
def decrypt(cipher_text, private_key):			
d, n = private_key	C1	1	C1
start_time = time.perf_counter()	C2	1	C2
plain_text = [chr(pow(char, d, n)) for char in cipher_text]	C3	n	C3 n
decryption_time = time.perf_counter() - start_time	C4	1	C4
return "".join(plain_text), decryption_time	C5	1	C5
Total			$T(n) = \Theta(n)$

Pada tabel 4.15, menyajikan informasi mengenai perhitungan kompleksitas waktu proses dekripsi algoritma RSA, yang hasilnya adalah  $T(n) = \Theta(n)$ .



## **BAB 5**

### **KESIMPULAN DAN SARAN**

Bab ini berisi kesimpulan dan saran dalam penelitian. Adapun kesimpulan didapatkan berdasarkan hasil analisis, rancangan, dan pengujian sistem dalam penelitian. Selain itu saran dituliskan sebagai catatan khusus dalam mengembangkan penelitian yang relevan.

#### **5.1 Kesimpulan**

Berdasarkan hasil analisis rancangan dan pengujian yang sudah dilaksanakan pada penelitian “Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi Pada Algoritma RSA dan Vincent- Sathiyamoorthy” maka kesimpulan yang didapat ialah algoritma Vincent- Sathiyamoorthy lebih cepat melakukan proses pengenkripsian dan pendekripsian dibanding dengan algoritma RSA dengan menggunakan nilai  $p$  dan  $q$  yang sama dan panjang *plaintext* yang berbeda. Selain itu, pada perhitungan pembuatan kunci, algoritma Vincent-Sathiyamoorthy lebih sulit dibanding dengan algoritma RSA sehingga algoritma RSA lebih cepat dibandingkan dengan algoritma Vincent-Sathiyamoorthy. Nilai  $N$  dan panjang *plaintext* mempengaruhi kecepatan enkripsi dekripsi pada kedua algoritma.

#### **5.2 Saran**

Berdasarkan analisis, perancangan dan pengujian sudah dilaksanakan pada penelitian “Analisis Perbandingan Kecepatan Waktu Enkripsi Dekripsi Pada Algoritma RSA dan Vincent- Sathiyamoorthy”, maka saran yang dapat diperhatikan untuk mengembangkan penelitian yang akan dilakukan selanjutnya, ialah:

1. Merancang sistem yang dapat melakukan proses enkripsi dan dekripsi dengan tidak memecah pesan menjadi blok-blok tertentu.
2. Merancang sistem yang dapat mengenkripsi dan mendekripsi file yang lain seperti docs, pdf, jpg.

3. Membandingkan Algoritma Vincent-Sathiyamoorthy dengan algoritma lainnya seperti ElGamal, LUC, ECC (Elliptic Curve Cryptography), Schmidt-Samoa.
4. Mengembangkan *user interface* menjadi lebih baik dan mudah dimengerti oleh pengguna.

## DAFTAR PUSTAKA

- Ahmad, K., Doja, M. N., Udzir, N. I., Singh, M. P. (Eds). (2019). *Emerging Security Algorithms and Techniques*. New York: CRC Press, Taylor & FrancisGroup, LLC
- Alemami, Yahia et all (2019). Research on Various Cryptography Techniques. *International Journal of Recent Technology and Engineering (IJRTE)*. DOI : 10.35940/ijrte.B1069.0782S319
- Anwar, Md. N. B., Hasan, M., Hasan, Md. M., Lorem, J. Z., Hossain, S. M. T. (2019). Comparative Study Of Cryptography Algoritms and It's Applications. *International Journal Of Computer Networks and Communications Security*, 7(5), 96-103
- Anshori, Y., Dodu, A. Y. E., & Wedananta, D. M. P. (2019). Implementasi Algoritma Kriptografi Rivest Shamir Adleman (RSA) pada Tanda Tangan Digital. *Techno. Com*, 18(2), 110–121.
- A V, Gahan., Devanagavi, G. D. (2019). A Empirical Study of Security Issues In Encryption Techniques. *International Journal of Applied Engineering Research*, 14 (5), 1049-1061
- Akbar, F. Y. (2021). *Teori Bilangan dalam Ilmu Kriptografi*. Institut Teknologi Bandung.
- Arianti, T., Fai'zi, A., Adam, S., Wulandari, M. (2022). PERANCANGAN SISTEM INFORMASI PERPUSTAKAAN MENGGUNAKAN DIAGRAM UML (UNIFIED MODELLING LANGUAGE). *Jurnal Ilmiah Komputer Terapan dan Informasi*, 1(1), 19-25
- Aulia, B. W., Rizki, M., Prindiyana, P., & Surgana, S. (2023). Peran Krusial Jaringan Komputer dan Basis Data dalam Era Digital. *JUSTINFO/ Jurnal Sistem Informasi Dan Teknologi Informasi*, 1(1), 9–20.
- Bufalo, M., Bufalo, D., Orlando, G. (2021). A Note on the Computation of the Modular Inverse for Cryptography. DOI: <https://doi.org/10.3390/axioms10020116>
- Chandra, S., Paira, S., Alam, Sk. S., Sanyal, G. (2014). A comparative survey of symmetric and asymmetric key cryptography. *International Conference on*

Electronics, Communication and Computational Engineering (ICECCE)

- Hafiz, A. (2019). Steganografi Berbasis Citra Digital Untuk Menyembunyikan Data Menggunakan Metode Least Significant Bit (LSB). *Jurnal Cendikia*, 17(1 April), 194–198.
- Hidayat, A. W., Arifudin. R., Akhlis, I. (2020). Implementation of RSA and RSA- CRT Algorithms for Comparison of Encryption and Decryption Time in Android-based Instant Message Applications. *Journal of Advances In Information Systems and Technology*, 2(2), 1-10
- Hermawati, F. D., Tahir, M., Syaifurrohman, M., Hikmah, M., Amroin, J. A., Bahrudin, M., & Irsyad, I. (2023). Keamanan E-Voting Di Indonesia Melalui Pemanfaatan Kriptografi Pada Sistem AES (Advance Encryption Standard). *Jurnal Teknik Mesin, Industri, Elektro Dan Informatika*, 2(2), 45–56.
- Kusumaningtyas, J. A. (2022). ANALISA PENGAPLIKASIAN SOAP HEADER LOGIN TOKEN UNTUK MENGAMANKAN WEB SERVICES SIKASA UKSW. *Jurnal Teknologi Informasi dan Komunikasi*, 12(1), 1-9
- Li, Shu., Tian, Jianwei., Zhu, Hongyu., Tian, Zheng., Qiao, Hong., Liu, Jie. (2019). Research In Fast Modular Exponentiation Algorithm Based On Fqga. *International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*. DOI : 10.1109/ICMTMA.2019.00024
- Lee, S. Jho, N., Chung, D., Kang, Y., Kim, M. (2021). Rcryptect: Real-time detection of cryptographic function in the user-space file system. <https://doi.org/10.1016/j.cose.2021.102512>
- Muchlis, B. S., Budiman, M. A., Rachmawati, Dian. (2017). Teknik Pemecahan Kunci Algoritma Rivest Shamir Adleman (RSA) dengan Metode Kraitichik. *Jurnal & Penelitian Teknik Informatika*, 2(2), 49-64
- Mallouli, F., Hellal, A., Saeed, N. S., Alzahrani, F. A. (2019). A Survey on Cryptography: comparative study between RSA vs ECC Algorithms And RSA vs El-Gamal Algorithms. DOI :10.1109/CSCloud/EdgeCom.2019.00022
- Munir, Rinaldi. 2019. Kriptografi (Edisi Kedua). Informatika Bandung
- Marqas, R. B., Almufti, S. M., Ihsan, R. R. (2020). Comparing Symmetric and Asymmetric cryptography in message encryption and decryption by using

- AES and RSA algorithms. *Journal of Xi'an University of Architecture & Technology*, 12 (3), 3110 – 3116
- Matahelumual, T. J., Fitri, A., & Asmara, A. S. (2020). Analisis Kemampuan Pemahaman Konsep KPK dan FPB Pada Siswa Kelas IV Sekolah Dasar. *Indonesian Journal of Primary School Education*, 1(1), 54–68.
- M, Siddharta., Rodriques, J., Chandavarkar, B. R. (2020). Greatest common divisor and its applications in security : Case study. *International Conference on Interdisciplinary Cyber Physical Systems*.
- Merliana, N. P. E. (2020). Pemanfaatan Teknologi Kriptografi dalam mengatasi Kejahatan Cyber. *Satya Dharma: Jurnal Ilmu Hukum*, 3(2), 23–40.
- Novantara, P., Apriani. A. (2021). Implementasi Algoritma Euclides Pada Model Pembelajaran Latihan FPB dan KPK Berbasis Android. *Jurnal Teknologi dan Manajemen Informatika*, 6(2), 43-53
- Ometov, A., Zeman, K., Masek, P., Balazeic, L., Komarov, M. (2021). A Comprehensive and Reproducible Comparison of Cryptographic Primitives Execution on Android Devices. 10.1109/ACCESS.2021.3069627
- Osamor, V.C., Edosomwan, I.B. (2021). Employing scrambled alpha-numeric randomization and RSA algorithm to ensure enhanced encryption in electronic medical records. *Informatics in Medicine Unlocked*. DOI : <https://doi.org/10.1016/j.imu.2021.100672>
- Putri, Y. D., Rosihan, R., & Lutfi, S. (2019). Penerapan Kriptografi Caesar Cipher Pada Fitur Chatting Sistem Informasi Freelance. *JIKO (Jurnal Informatika Dan Komputer)*, 2(2), 87–94.
- Purba, B., Gulo, F. A., Utami, N. I., & Sihotang, Y. A. (2020). Pengamanan File Teks Menggunakan Algoritma RC4. *Seminar Nasional Teknologi Komputer & Sains (SAINTEKS)*, 1(1), 420–425.
- Pamungkas, A. A., & Irawan, A. S. Y. (2023). Analisis Penerapan Algoritma Kriptografi Rivest-Shamir-Adleman (RSA) dan Zero-Knowledge Proof Pada Aplikasi Whatsapp Mod. *Jurnal Ilmiah Wahana Pendidikan*, 9(13), 81–95.
- Rhomdani, R. W., & Ningtyas, Y. D. W. K. (2021). Aplikasi modulo berpangkat ab mod n menggunakan pola barisan dan teorema Euler berbasis web. *JEMS: Edukasi Matematika Dan Sains*, 9(2), 499–506.
- Suroso, A. (2018). Studi Perbandingan Kriptografi Menggunakan Metode DES, Triple

- DES dan RSA. *Jurnal SIGMA*, 8(1), 17–25.
- Suhandinata, S., Rizal, R. A., Wijaya, D. O., Warren, P., Srinjiwi. (2019). Analisis Performa Kriptografi Hybrid Algoritma Blowfish Dan Algoritma RSA. *JURTEKSI (Jurnal Teknologi dan Sistem Informasi)*, 6(1), 1-10
- Sulistiyorini., Prihanto, A. (2019). Perbandingan Efisiensi Algoritma RSA dan RSA-CRT Dengan Data Teks Berukuran Besar. *Journal Of Informatics and Computer Science*, 2(1), 84-90
- Saputro, T. H., Hidayati, N. H., & Ujianto, E. I. H. (2020). Survei Tentang Algoritma Kriptografi Asimetris. *Jurnal Informatika Polinema*, 6(2), 67–72.
- Siddhartha, M., Rodriques, J., & Chandavarkar, B. R. (2020). Greatest common divisor and its applications in security: Case study. 2020 International Conference on Interdisciplinary Cyber Physical Systems (ICPS), 51–57.
- Suharya, Y., Widia, H. (2020). Implementasi Digital Signature Menggunakan Algoritma Kriptografi RSA untuk Pengamanan Data di SMK Wirakarya 1 Ciparay. *Jurnal Informatika-Computing*, 07 (01), 20-29
- Thabab, S. D., Somowal, M., Ahmed, R. U., Saha, P. (2019). Fast and Area Efficient Implementation of RSA Algorithm. International Conference on Recent Trends In Advanced Computing (ICRTAC). DOI : 10.1016/j.procs.2020.01.024
- Vincent, P.M Durai Raj., Sathiyamoorthy, E. (2016). A novel and efficient public key encryption algorithm. *Int. J. Information and Communication Technology*, 19 (2), 199-211
- Vahdati, Z., Yasin, S. MD., Ghasempour, Ali., Saleh, M. (2019). COMPARISON OF ECC AND RSA ALGORITHMS IN IOT DEVICES. *Journal of Theoretical and Applied Information Technology*, 97(16), 4293-4308
- Yel, M. B., & Nasution, M. K. M. (2022). Keamanan informasi data pribadi pada media sosial. *Jurnal Informatika Kaputama (JIK)*, 6(1), 92–101.
- Yousif, S. F. (2023). Performance Comparison between RSA and El-Gamal Algorithms for Speech Data Encryption and Decryption. *Diyala Journal of Engineering Sciences*. 16(1), 123-137
- Zhu, Xiaomeng. (2022). The Role of Number Theory to Cryptography. *International Journal of Mathematics and Statistic Invention (IJMSI)*, 10(6), 07-13