

**PERBANDINGAN METODE *KRAITCHIK* DAN METODE PEMFAKTORAN  
*FERMAT* DALAM KRIPTANALISIS KUNCI PUBLIK ALGORITMA  
*SCHMIDT-SAMOA***

**SKRIPSI**

**REYNOLD GIDEON MUNTHER  
20140160**



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

**PERBANDINGAN METODE *KRAITCHIK* DAN METODE PEMFAKTORAN  
*FERMAT* DALAM KRIPTANALISIS KUNCI PUBLIK ALGORITMA  
*SCHMIDT-SAMOA***

**SKRIPSI**

Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah  
Sarjana Ilmu Komputer

**REYNOLD GIDEON MUNTHE**

**20140160**



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

**PERSETUJUAN**

Judul : PERBANDINGAN METODE KRAITCHIK DAN  
METODE PEMFAKTORAN FERMAT DALAM  
KRIPTANALISIS KUNCI PUBLIK ALGORITMA  
*SCHMIDT-SAMOA*

Kategori : SKRIPSI

Nama : REYNOLD GIDEON MUNTHE

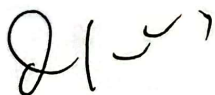
Nomor Induk Mahasiswa : 201401060

Program Studi : SARJANA (S1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI

Komisi Pembimbing :

Pembimbing 2



Dian Rachmawati S.Si., M.Kom.

NIP. 198307232009122004

Pembimbing 1



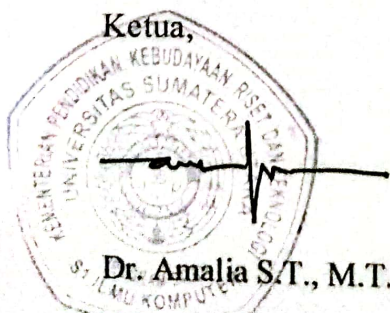
Amer Sharif S.Si., M.Kom.

NIP. 196910212021011001

Diketahui/disetujui oleh

Program Sarjana S1 Ilmu Komputer

Ketua,



Dr. Amalia S.T., M.T.

NIP. 197812212014042001

**PERNYATAAN**

**PERBANDINGAN METODE *KRAITCHIK* DAN METODE PEMFAKTORAN  
*FERMAT* DALAM KRIPTANALISIS KUNCI PUBLIK ALGORITMA  
*SCHMIDT-SAMOA***

**SKRIPSI**

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

Medan,



Reynold Gideon Munthe

201401060

## PENGHARGAAN

Segala puji dan syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas segala berkah dan anugerah-Nya, penulis bisa menyelesaikan laporan tugas akhir ini dengan baik.

Dalam proses penyusunan tugas akhir ini, penulis menerima banyak bantuan, dukungan, dan doa dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan rasa terima kasih yang mendalam kepada:

1. Bapak Prof. Dr. Muryanto Amin, S.Sos., M.Si. selaku Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lydia B.Sc., M.Sc. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Ibu Dr. Amalia, ST., M.T. selaku Ketua Program Studi S-1 Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Bapak Amer Sharif S.Si., M.Kom. selaku Dosen Pembimbing 1 yang selalu memberikan bimbingan, saran, masukan, serta motivasi kepada penulis selama penulisan skripsi ini.
5. Ibu Dian Rachmawati S.Si., M.Kom. selaku Dosen Pembimbing 2 yang selalu memberikan bimbingan, saran, masukan, serta motivasi kepada penulis selama penulisan skripsi ini.
6. Bapak / Ibu dosen di lingkungan Program Studi S-1 Ilmu komputer yang telah banyak memberikan ilmu pengetahuan yang bermakna bagi penulis.
7. Kedua orang tua tercinta, Ayahanda Monalon Munte S.Pd dan Ibunda Laonma Silaban S.Pd yang telah memberikan restu, dukungan, motivasi, kasih sayang, dan doa yang tulus serta pengorbanan yang tidak ternilai harganya.
8. Saudara penulis Twinda Yohana Munthe, Twinsis Yohani Munthe, dan Raymond Giovanni Munthe yang selalu memberi doa, motivasi, dukungan kepada penulis.
9. Keluarga Bapak R. Hutagalung / R. Silaban dan Keluarga C. M. Ambarita / S. Silaban yang telah banyak memberikan dukungan motivasi selama penulis menjalani studi.

10. Keluarga dan kerabat dekat dan jauh yang membantu penulis dalam menjalani studinya dengan baik.
11. Teman-teman seperjuangan mahasiswa S1 Ilmu Komputer stambuk 2020 khususnya kom B yang telah memberikan pengalaman belajar bersama yang berharga kepada penulis.
12. Seluruh anggota Ikatan Mahasiswa Humbang Hasundutan USU (IMHU) yang telah memberikan pengalaman dalam berorganisasi.
13. Anak Humbang @Mandolin56 yang telah memberikan semangat dan motivasi kepada penulis.
14. Semua pihak yang tidak dapat disebutkan satu per satu, yang telah membantu dalam bentuk apapun selama proses penyusunan skripsi ini.

Penulis sadar bahwa skripsi ini masih belum sempurna, oleh sebab itu penulis sangat mengharapkan kritik dan juga saran yang membangun untuk dapat menjadi perbaikan di masa mendatang.

Medan, Juli 2024

Penulis,



Reynold Gideon Munthe

**PERBANDINGAN METODE *KRAITCHIK* DAN METODE  
PEMFAKTORAN *FERMAT* DALAM KRIPTANALISIS KUNCI PUBLIK  
ALGORITMA *SCHMIDT-SAMOA***

**ABSTRAK**

Di era modern ini, keamanan data menjadi fokus utama dalam setiap aspek komunikasi dan pertukaran informasi. Oleh karena itu, algoritma kriptografi, khususnya yang menggunakan kunci publik, sangat penting untuk menjamin kerahasiaan, integritas, dan keaslian data. Salah satu contoh algoritma yang menggunakan kunci publik yang banyak digunakan adalah Algoritma *Schmidt-Samoa*. Namun, seiring dengan kemajuan pesat dalam teknologi dan perkembangan metode kriptanalisis, penting mengevaluasi keamanan algoritma kriptografi yang ada secara terus menerus. Salah satu teknik kriptanalisis yang dapat digunakan untuk mengevaluasi keamanan Algoritma *Schmidt-Samoa* adalah metode *Kraitichik* dan metode pemfaktoran *Fermat*. Melakukan perbandingan terhadap efisiensi kedua metode dalam kriptanalisis sangat penting karena masing-masing metode memiliki pendekatan dan kekuatan yang berbeda dalam memecahkan masalah faktorisasi. Metode pemfaktoran *Fermat* memiliki kecepatan pemfaktoran yang lebih cepat dibanding metode *Kraitichik*. Ketika nilai  $p$  lebih besar dibanding  $q$ , maka waktu pemfaktorannya lebih lama dibandingkan ketika nilai  $p$  lebih kecil dari  $q$ .

**Kata Kunci** : Kriptografi, Kriptanalisis, *Schmidt-Samoa*, Metode *Kraitichik*, Metode Pemfaktoran *Fermat*.

## **COMPARISON OF KRAITCHIK METHOD AND FERMAT'S DIFFERENCE OF SQUARES IN PUBLIC KEY CRYPTANALYSIS OF SCHMIDT-SAMOA ALGORITHM**

### **ABSTRACT**

In the modern era, data security has become a primary focus in all aspects of communication and information exchange. Cryptographic algorithms, particularly those using public keys, play a crucial role in ensuring the confidentiality, integrity, and authenticity of data. One widely used public-key algorithm is the Schmidt-Samoa Algorithm. However, with rapid advancements in technology and the development of cryptanalysis methods, it is essential to continuously evaluate the security of existing cryptographic algorithms. One cryptanalysis technique that can be used to evaluate the security of the Schmidt-Samoa Algorithm is the Kraitichik method and the Fermat factorization method. Comparing the efficiency of these two methods in cryptanalysis is vital as each method has different approaches and strengths in solving factorization problems. The Fermat factorization method is generally faster compared to the Kraitichik method. When the value of  $p$  is larger than  $q$ , the factorization time is longer compared to when the value of  $p$  is smaller than  $q$ .

**Keywords :** Cryptography, Cryptanalysis, Schmidt-Samoa Algorithm, Kraitichik Method, Fermat Factoring Method.



## DAFTAR ISI

PERSETUJUAN .....	ii
PERNYATAAN .....	iii
PENGHARGAAN .....	iv
ABSTRAK .....	vi
ABSTRACT .....	vii
DAFTAR ISI .....	viii
DAFTAR TABEL .....	x
DAFTAR GAMBAR .....	xi
BAB I PENDAHULUAN .....	1
1.1. Latar Belakang .....	1
1.2. Rumusan Masalah .....	4
1.3. Batasan Penelitian .....	4
1.4. Tujuan Penelitian .....	5
1.5. Manfaat Penelitian .....	5
1.6. Metode Penelitian .....	5
1.7. Sistematika Penelitian .....	6
BAB II LANDASAN TEORI .....	8
2.1. Kriptografi .....	8
2.2. Algoritma <i>Schmidt-Samoa</i> .....	10
2.3. Algoritma Tes primalitas <i>Lehman</i> .....	13
2.4. Kriptanalisis .....	14
2.5. Pemfaktoran Metode <i>Kraitchik</i> .....	15
2.6. Metode Pemfaktoran <i>Fermat</i> .....	16
2.7. Penelitian Relevan .....	18
BAB III ANALISIS DAN PERANCANGAN .....	20
3.1. Analisis .....	20
3.1.1. Analisis Masalah .....	20
3.1.2. Analisis Kebutuhan .....	21
3.2. Perancangan Sistem .....	23

3.2.1. Diagram Umum .....	23
3.2.2. <i>Use Case Diagram</i> .....	24
3.2.3. <i>Activity Diagram</i> .....	25
3.2.4. <i>Sequence Diagram</i> .....	29
3.2.5. Diagram Alir ( <i>flowchart</i> ) .....	31
3.2.6. Perancangan <i>User Interface</i> .....	41
<b>BAB IV IMPLEMENTASI DAN PENGUJIAN</b> .....	<b>45</b>
4.1. Implementasi Sistem .....	45
4.1.1. Laman Beranda .....	45
4.1.2. Laman Pembangkitan Kunci Publik .....	46
4.1.3. Laman Pemfaktoran Kunci Publik .....	46
4.2. Pengujian Sistem.....	47
4.2.1. Pengujian Pembangkitan Kunci Publik.....	47
4.2.2. Pengujian Pemfaktoran Kunci Publik .....	48
4.3. Kompleksitas Waktu .....	53
<b>BAB V PENUTUP</b> .....	<b>56</b>
5.1. Kesimpulan .....	56
5.2. Saran .....	57
<b>DAFTAR PUSTAKA</b> .....	<b>58</b>

## DAFTAR TABEL

Tabel 4. 1 Tabel Perbandingan Waktu Eksekusi .....	49
Tabel 4. 2 Perbandingan Waktu Metode Kraitchik.....	51
Tabel 4. 3 Perbandingan Waktu Metode Pemfaktoran Fermat.....	52

## DAFTAR GAMBAR

Gambar 2. 1 Algoritma Simetris .....	9
Gambar 2. 2 Algoritma Asimetris .....	9
Gambar 3. 1 Diagram Ishikawa Masalah Penelitian .....	21
Gambar 3. 2 Diagram Umum Kriptanalisis .....	23
Gambar 3. 3 Diagram Umum Pembangkitan Kunci .....	24
Gambar 3. 4 Use Case Diagram .....	25
Gambar 3. 5 Activity Diagram Pembangkitan Kunci .....	27
Gambar 3. 6 Activity Diagram Kriptanalisis Kunci Publik .....	29
Gambar 3. 7 Sequence Diagram Pembangkitan Kunci .....	30
Gambar 3. 8 Sequence Diagram Kriptanalisis Kunci Publik .....	31
Gambar 3. 9 Diagram Alir Pembangkitan Kunci .....	33
Gambar 3. 10 Diagram Alir Algoritma Tes Primalitas Lehman .....	35
Gambar 3. 11 Diagram Alir Kriptanalisis .....	36
Gambar 3. 12 Diagram Alir Metode Kraitchik .....	38
Gambar 3. 13 Diagram Alir Metode Pemfaktoran Fermat .....	40
Gambar 3. 14 Rancangan Menu Beranda .....	41
Gambar 3. 15 Rancangan Menu Pembangkitan Kunci .....	42
Gambar 3. 16 Rancangan Menu Pemfaktoran Kunci Publik .....	43
Gambar 4. 1 Laman Beranda .....	45
Gambar 4. 2 Laman Pembangkitan Kunci Publik .....	46
Gambar 4. 3 Laman Pemfaktoran Kunci Publik .....	47
Gambar 4. 4 Pengujian Pembangkitan Kunci Publik .....	48
Gambar 4. 5 Pengujian Pemfaktoran Kunci Publik .....	48
Gambar 4. 6 Grafik Perbandingan Waktu Pemfaktoran .....	50
Gambar 4. 7 Grafik Perbandingan Waktu Metode Kraitchik .....	52
Gambar 4. 8 Grafik Perbandingan Waktu Metode Fermat .....	53
Gambar 4. 9 Pseudocode dan Kompleksitas Waktu metode Kraitchik .....	54
Gambar 4. 10 Pseudocode dan Kompleksitas Waktu metode pemfaktoran Fermat ....	55

# BAB I

## PENDAHULUAN

### 1.1. Latar Belakang

Pada era modern saat ini yang didominasi oleh pertukaran informasi digital, perlindungan terhadap data sensitif menjadi sangat penting. Keamanan data menjadi fokus utama dalam setiap aspek komunikasi dan pertukaran informasi, baik dalam lingkup pribadi, seperti pertukaran pesan antara individu, penyimpanan data pribadi, atau pembayaran online, maupun dalam lingkup umum, seperti pertukaran informasi di media sosial, transfer file melalui internet, atau akses ke platform pelayanan publik. Ancaman terhadap keamanan data bisa berupa peretasan, pencurian identitas, atau bahkan penyadapan komunikasi. Oleh karena itu, algoritma kriptografi, khususnya yang menggunakan kunci publik, memainkan peran utama dalam memastikan kerahasiaan, integritas, dan keaslian data dalam berbagai konteks penggunaan. Salah satu contoh algoritma yang menggunakan kunci publik yang banyak digunakan adalah Algoritma *Schmidt-Samoa*.

Algoritma *Schmidt-Samoa* adalah sebuah algoritma kriptografi kunci publik yang dikembangkan oleh Katja Schmidt-Samoa. Algoritma ini adalah sebuah usulan baru yang terinspirasi dari algoritma kriptografi RSA dan algoritma kriptografi *Rabin*. Sama halnya dengan algoritma RSA dan Rabin, *Schmidt-Samoa Cryptosystem* menggunakan masalah faktorisasi sebagai dasar sebagai proses pembangkitan kunci, enkripsi, dan dekripsi. Algoritma ini didasarkan pada prinsip – prinsip matematika yang kompleks, yang menjadikannya sebagai salah satu algoritma yang dianggap aman dalam dunia kriptografi.

Angka prima mempunyai peran penting dalam algoritma kriptografi kunci publik karena keunikannya yang hanya memiliki dua faktor, yaitu 1 dan dirinya

sendiri. Sifat dasar ini membuat bilangan prima sangat penting dalam pembangkitan kunci kriptografi yang aman. Penggunaan bilangan prima yang besar dan sulit diprediksi menjadikan algoritma kriptografi lebih tangguh terhadap serangan faktorisasi. Untuk memastikan bahwa bilangan yang digunakan adalah bilangan prima, digunakan algoritma tes primalitas. Salah satu algoritma yang efektif untuk tujuan ini adalah algoritma tes primalitas *Lehman*. Algoritma tes primalitas *Lehman* bekerja dengan cara menguji suatu bilangan terhadap sejumlah basis acak dan menggunakan sifat-sifat bilangan prima untuk mengidentifikasi bilangan komposit. Dengan menambahkan algoritma tes primalitas *Lehman*, Algoritma *Schmidt-Samoa* menjadi lebih kuat dalam memilih bilangan prima yang digunakan dalam proses pembangkitan kunci, sehingga meningkatkan keamanan keseluruhan dari sistem tersebut. Penggunaan bilangan prima yang terjamin keasliannya adalah salah satu faktor utama dalam menjaga keamanan algoritma kriptografi kunci publik.

Namun, seiring dengan kemajuan pesat dalam teknologi dan perkembangan metode kriptanalisis, penting mengevaluasi keamanan algoritma kriptografi yang ada secara terus menerus, termasuk Algoritma *Schmidt-Samoa*. Tantangan baru yang muncul memerlukan respons yang cepat dan efektif untuk memastikan bahwa sistem-sistem kriptografi tetap tangguh dan dapat diandalkan. Salah satu teknik kriptanalisis yang telah digunakan untuk mengevaluasi keamanan Algoritma *Schmidt-Samoa* adalah metode *Kraitichik* dan metode pemfaktoran *Fermat*.

Metode *Kraitichik* adalah salah satu teknik dalam kriptanalisis yang telah terbukti efektif dalam mengungkap kerentanan pada berbagai algoritma kriptografi. Metode ini berfokus pada penemuan faktor-faktor yang mungkin dari bilangan-bilangan yang terlibat dalam algoritma kriptografi, sehingga memungkinkan identifikasi potensi kelemahan dalam keamanan algoritma. Dengan menerapkan metode ini, para peneliti dapat mengidentifikasi kerentanan atau kelemahan potensial dalam keamanan algoritma tersebut, sehingga

memungkinkan pengembangan langkah-langkah perlindungan yang lebih kuat dan efektif.

Selain metode *Kraitichik*, metode kriptanalisis lainnya yang relevan untuk mengevaluasi keamanan Algoritma *Schmidt-Samoa* adalah metode faktorisasi *Fermat*. Metode faktorisasi *Fermat* adalah teknik yang digunakan untuk memecahkan bilangan bulat menjadi faktor-faktor primanya. Metode ini didasarkan pada prinsip bahwa setiap bilangan ganjil dapat dinyatakan sebagai selisih dua kuadrat. Dalam konteks kriptografi, metode ini dapat diaplikasikan untuk mencari faktor-faktor dari bilangan besar yang digunakan dalam algoritma kunci publik, seperti algoritma *Schmidt-Samoa*. Jika bilangan yang akan difaktorkan memiliki faktor yang cukup dekat satu sama lain, metode faktorisasi *Fermat* dapat sangat efektif dan cepat. Dengan demikian, mengevaluasi Algoritma *Schmidt-Samoa* menggunakan metode ini memungkinkan para peneliti untuk mengidentifikasi potensi kerentanan dalam keamanan algoritma yang disebabkan oleh kelemahan dalam proses pemfaktoran bilangan besar.

Melakukan perbandingan terhadap efisiensi metode *Kraitichik* dan metode faktorisasi *Fermat* dalam kriptanalisis sangat penting karena masing-masing metode memiliki pendekatan dan kekuatan yang berbeda dalam memecahkan masalah faktorisasi. Perbandingan ini tidak hanya akan memberikan gambaran yang lebih jelas tentang kelemahan spesifik Algoritma *Schmidt-Samoa*, tetapi juga memungkinkan penentuan metode mana yang lebih efektif dalam mengidentifikasi kerentanan. Pemahaman mendalam tentang efisiensi kedua metode ini akan membantu dalam mengembangkan algoritma kriptografi yang lebih aman dan tangguh terhadap berbagai teknik serangan kriptanalisis, sehingga meningkatkan perlindungan terhadap data sensitif dalam komunikasi digital.

## 1.2. Rumusan Masalah

Algoritma *Schmidt-Samoa* memiliki keunggulan pada prinsip matematika yang kompleks pada kuncinya dan menggunakan bilangan prima yang telah diuji menggunakan algoritma tes primalitas seperti algoritma *Lehman* untuk memastikan keamanan kunci publiknya. Namun, algoritma ini masih bisa diserang menggunakan beberapa metode kriptanalisis seperti metode *Kraitchik* dan metode pemfaktoran *Fermat*. Oleh sebab itu, pada penelitian ini akan dilakukan kriptanalisis terhadap kunci publik algoritma *Schmidt-Samoa* menggunakan metode *Kraitchik* dan metode pemfaktoran *Fermat* untuk mengetahui bagaimana perbandingan efisiensi kedua metode pemfaktoran tersebut dalam memfaktorkan kunci publik algoritma *Schmidt-Samoa*.

## 1.3. Batasan Penelitian

1. Algoritma kriptografi yang diuji adalah algoritma *Schmidt-Samoa*
2. Proses pada sistem ada 2 yaitu :
  - Membangkitkan bilangan  $p$  dan  $q$  yang adalah bilangan prima dengan memakai algoritma tes keprimaan *Lehman*, dan membangkitkan kunci publik ( $n$ ) dari bilangan yang dibangkitkan.
  - Memfaktorkan kunci publik  $n$  untuk mendapatkan nilai  $p$  dan  $q$  menggunakan dua metode yaitu metode *Kraitchik* dan metode pemfaktoran *Fermat*
3. Sistem yang dibangun tidak melakukan proses enkripsi dan dekripsi pesan.
4. Parameter efisiensi pemecahan kunci diukur menggunakan dari lamanya *running time* program dalam satuan *milliseconds (ms)*
5. Program dirancang menggunakan bahasa pemrograman *Python*
6. GUI dirancang menggunakan framework *streamlit*.



#### 1.4. Tujuan Penelitian

Penelitian ini bertujuan untuk membangun sebuah sistem yang melakukan kriptanalisis terhadap kunci publik algoritma *Schmidt-Samoa* menggunakan metode *Kraitichik* dan metode pemfaktoran *Fermat*, serta memastikan bilangan prima yang digunakan dalam kunci publik melalui algoritma tes primalitas *Lehman*. Selain itu, penelitian ini juga memiliki tujuan untuk membandingkan efisiensi kedua metode kriptanalisis tersebut dalam memfaktorkan kunci publik algoritma *Schmidt-Samoa*.

#### 1.5. Manfaat Penelitian

Mengetahui perbandingan efisiensi metode *Kraitichik* dan metode pemfaktoran *Fermat* dalam memecahkan kunci publik algoritma *Schmidt-Samoa* yang dapat memberikan wawasan sebagai tumpuan untuk memajukan metode kriptanalisis yang lebih efisien di masa mendatang dan dapat diangkat sebagai rujukan bagi para peneliti yang tertarik dengan algoritma *Schmidt-Samoa*, metode *Kraitichik* dan metode pemfaktoran *Fermat*.

#### 1.6. Metode Penelitian

Langkah - langkah penelitian yang akan dilakukan pada proses penyusunan tugas akhir ini adalah:

##### 1. Studi Pustaka

Pada fase ini, penelitian akan diawali dengan mengumpulkan referensi yang didapatkan melalui sumber tertulis, seperti jurnal, buku, serta artikel ilmiah. Pencarian referensi dilakukan untuk mendapatkan informasi yang terkait dengan algoritma *Schmidt-Samoa*, algoritma tes primalitas *Lehman*, serta teknik kriptanalisis seperti metode *Kraitichik* dan metode pemfaktoran *Fermat*.

## 2. Analisis dan Perancangan

Pada fase ini, analisis akan dilaksanakan terhadap algoritma tes primalitas *Lehman*, algoritma *Schmidt-Samoa* yang akan dikriptanalisis, metode *Kraitchik* dan metode pemfaktoran *Fermat* yang akan memecahkan kunci publik algoritma *Schmidt-Samoa* dengan melakukan perancangan diagram alir (*flowchart*), *sequence* diagram, *activity* diagram, *usecase* diagram, *user interface* dan *ishikawa* diagram.

## 3. Implementasi Sistem

Dalam fase ini, proses pengkodean (*coding*) akan dilakukan dengan menggunakan bahasa *python*.

## 4. Pengujian Sistem

Dalam fase ini dilakukan proses uji coba kepada sistem yang dibangun.

## 5. Dokumentasi

Dalam fase ini, akan dilaksanakan proses penyusunan laporan yang melibatkan tahap analisis sampai pengujian dalam format skripsi.

### 1.7. Sistematika Penelitian

Dalam struktur penyusunan tugas akhir ini, terbagi menjadi lima bab, dengan setiap bab dijelaskan sebagai berikut :

#### **BAB 1            PENDAHULUAN**

Pada bagian ini kita akan memahami penjelasan di balik pemilihan judul, pembahasan masalah, termasuk sasaran penelitian, pemanfaatan hasil ujian, strategi penelitian, dan ikhtisar artikel.

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

Algoritma *Schmidt-Samoa*, metode *Kraitchik*, metode pemfaktoran *Fermat*, dan algoritma tes primalitas *Lehman* merupakan subjek tinjauan teoritis dalam bab ini.

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

Analisis masalah dan desain perangkat lunak dijelaskan dalam bab ini.

**BAB 4            IMPLEMENTASI DAN PENGUJIAN**

Implementasi perangkat lunak hasil analisis dan perancangan yang telah dilakukan, dibahas pada bab ini.

**BAB 5            PENUTUP**

Pada bagian ini, akan dibahas rangkuman dari penjelasan bab-bab sebelumnya. Dengan berdasarkan kesimpulan tersebut, penulis akan memberikan saran yang dapat memberikan kontribusi positif untuk melengkapi serta meningkatkan pengembangan sistem yang telah disusun.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1. Kriptografi**

Kriptografi ialah bidang yang mendalami proses-proses matematika yang berkaitan dengan keamanan informasi, termasuk aspek aspek seperti kerahasiaan, integritas data, serta otentikasi (Menez, 1996). Tujuan utama kriptografi adalah untuk melakukan perlindungan terhadap informasi rahasia dari pihak - pihak yang tidak berhak. Informasi yang dilindungi dapat berbentuk pesan, file, gambar, teks, audio, video, dan sejenisnya.

Proses kriptografi melibatkan dua langkah utama, yaitu :

1. Enkripsi

Enkripsi ialah suatu proses mengganti data asli (*plaintext*) menjadi bentuk yang tidak bisa dimengerti (*ciphertext*) menggunakan algoritma kriptografi dan kunci enkripsi yang sesuai.

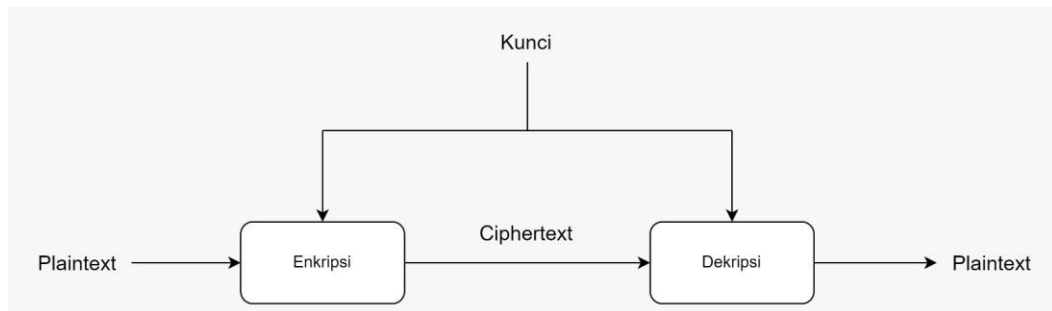
2. Dekripsi

Dekripsi adalah proses mengembalikan data yang dienkripsi kembali ke bentuk aslinya (*plaintext*) dengan memakai kunci dekripsi yang sesuai.

Jenis kriptografi dibedakan berdasarkan kunci yang digunakan. Secara umum, ada 2 macam algoritma kriptografi berdasarkan kuncinya, yaitu :

1. Algoritma Simetris

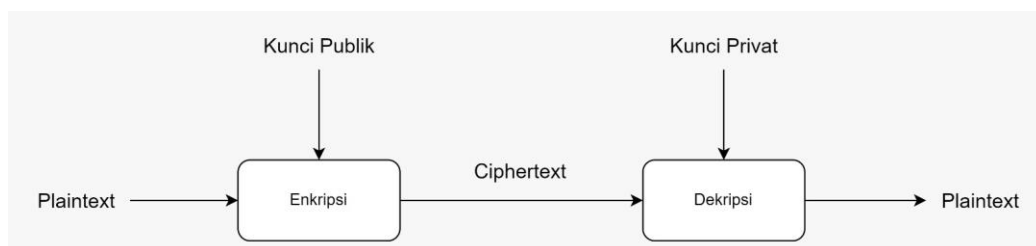
Algoritma simetris, atau yang juga dikenal sebagai algoritma konvensional, adalah jenis algoritma kriptografi yang mana kunci yang dipakai oleh pengirim untuk mengenkripsi adalah sama dengan kunci yang dipakai oleh penerima untuk mendekripsi. Contoh algoritma kriptografi simetri adalah *Caesar Cipher*, *Hill Cipher*, *Affine Cipher*.



**Gambar 2. 1 Algoritma Simetris**

## 2. Algoritma Asimetris

Algoritma asimetris merupakan jenis algoritma kriptografi yang memakai dua kunci yang tidak sama untuk melakukan proses enkripsi dan dekripsi. Dalam algoritma ini, terdapat sepasang kunci berbeda yakni kunci publik (*public key*) dan kunci privat (*private key*). Kunci publik dipakai untuk melakukan enkripsi pesan yang dikirim nantinya dan dapat dibagikan secara terbuka kepada siapa saja. Sebaliknya, kunci privat dipakai untuk mendekripsi terhadap pesan yang diterima dan wajib disimpan secara rahasia. Contoh algoritma kriptografi asimetris adalah *Schmidt-Samoa Cryptosystem*, *RSA(Rivest Shamir Adleman)*, *Rabin Cryptosystem*, *Diffie-Hellman*, *ElGamal*.



**Gambar 2. 2 Algoritma Asimetris**

## 2.2. Algoritma *Schmidt-Samoa*

Algoritma *Schmidt-Samoa* ialah salah satu algoritma kriptografi asimetris yang dikembangkan oleh Katja Schmidt-Samoa. Algoritma *Schmidt-Samoa* dikembangkan berdasarkan prinsip-prinsip matematika yang kompleks, sehingga membuatnya dianggap sebagai algoritma yang aman. Algoritma ini terinspirasi dari *Rabin Cryptosystem* dan algoritma RSA. Terinspirasi dari *Rabin Cryptosystem* dan algoritma RSA, algoritma *Schmidt-Samoa* menggunakan persoalan pemfaktoran untuk proses dekripsi, enkripsi, dan proses pembangkitan kunci. Namun *Schmidt-Samoa Cryptosystem* menggunakan pemfaktoran bilangan  $n = p^2 \cdot q$ , berbeda dengan *Rabin Cryptosystem* dan algoritma RSA yang menggunakan pemfaktoran bilangan  $n = p \cdot q$ .

Algoritma *Schmidt-Samoa* memiliki beberapa variabel, yaitu :

- Bilangan prima  $p$  dan  $q$  (rahasia)
- $n = p^2 \cdot q$  (tidak rahasia)
- $d = n^{-1} \bmod \text{lcm}(p-1, q-1)$  (rahasia)
- $m$  (*plaintext*) (rahasia)
- $c$  (*ciphertext*) (tidak rahasia)

Algoritma *Schmidt-Samoa* memiliki beberapa tahapan, yaitu :

### 1. Pembangkitan Kunci

Proses pembangkitan kunci algoritma *Schmidt-Samoa* adalah sebagai berikut :

a. Ambil dua bilangan sembarang berbeda  $p$  dan  $q$  yang merupakan bilangan prima.

b. Hitung nilai kunci publik ( $n$ ) dengan rumus

$$n = p^2 \cdot q$$

c. Hitung nilai  $d$  yang merupakan kunci privat dengan rumus

$$d = n^{-1} \bmod \text{lcm}(p-1, q-1)$$

d. Publikasikan nilai  $n$  sebagai kunci publik, sedangkan  $p$ ,  $q$  dan  $d$  merupakan kunci privat.

**Contoh :** Misalkan si A ingin membangkitkan kunci publik dan kunci privat miliknya. Lalu si A memutuskan dua bilangan, yaitu  $p = 31$  dan  $q = 43$ . Kemudian, si A melakukan perhitungan sebagai berikut untuk mendapatkan kunci publik miliknya :

$$n = p^2 \cdot q = 31^2 \cdot 43 = 41323$$

Lalu dia menghitung kunci privat miliknya :

$$d = n^{-1} \bmod \text{lcm}(p - 1, q - 1)$$

$$d = 41323^{-1} \bmod \text{lcm}(30, 42)$$

$$d = 41323^{-1} \bmod 210$$

$$d = 67$$

## 2. Proses Enkripsi

Proses enkripsi pesan menggunakan Algoritma *Schmidt-Samoa* adalah sebagai berikut :

- a. Ambil kunci publik dari si penerima pesan (A), yaitu  $n$ .
- b. Tentukan pesan yang akan dikirim ( $m$ ) dan konversi ke menggunakan pengkodean ASCII.
- c. Enkripsi pesan (*plaintext*) menjadi *ciphertext* ( $c$ ) dengan menggunakan rumus :

$$c = m^n \bmod n$$

**Contoh :** Misalkan si B ingin mengirimkan pesan terhadap si A. Pesan (*plaintext*) yang ingin dikirim terhadap si A adalah

$$m = \text{REYNOLD}$$

si B mengonversikan  $m$  menggunakan pengkodean ASCII:

$$m_1 = R = 82$$

$$m_2 = E = 69$$

$$m_3 = Y = 89$$

$$m_4 = N = 78$$

$$m_5 = O = 79$$

$$m_6 = L = 76$$

$$m_7 = D = 68$$

Lalu B mengubah semua *plaintext* ( $m$ ) menjadi *chipertext* ( $c$ ) :

$$c_1 = 82^{41323} \bmod 41323 = 41178$$

$$c_2 = 69^{41323} \bmod 41323 = 41094$$

$$c_3 = 89^{41323} \bmod 41323 = 30378$$

$$c_4 = 78^{41323} \bmod 41323 = 9180$$

$$c_5 = 79^{41323} \bmod 41323 = 1584$$

$$c_6 = 76^{41323} \bmod 41323 = 26285$$

$$c_7 = 68^{41323} \bmod 41323 = 19660$$

Setelah seluruh isi pesan (*plaintext*) selesai diubah menjadi *chipertext*, lalu B mengirimkan pesannya kepada A.

### 3. Proses Dekripsi

Proses dekripsi pesan menggunakan Algoritma *Schmidt-Samoa* adalah sebagai berikut :

- a. Ambil kunci privat  $d$ ,  $p$  dan  $q$
- b. Dekripsi ciphertext ( $c$ ) menjadi *plaintext* dengan menggunakan rumus :

$$m = c^d \bmod (p.q)$$

**Contoh :** A menerima ciphertext dari B. Lalu dia mendekripsi *ciphertext* menjadi *plaintext* agar dapat dibaca.

$$m_1 = 41178^{67} \bmod 1333 = 82 = R$$

$$m_2 = 41094^{67} \bmod 1333 = 69 = E$$

$$m_3 = 30378^{67} \bmod 1333 = 89 = Y$$

$$m_4 = 9180^{67} \bmod 1333 = 78 = N$$



$$m_5 = 1584^{67} \bmod 1333 = 79 = O$$

$$m_6 = 26285^{67} \bmod 1333 = 76 = L$$

$$m_7 = 19660^{67} \bmod 1333 = 68 = D$$

Akhirnya, *plaintext* yang didapatkan adalah  $m = \text{REYNOLD}$

### 2.3. Algoritma Tes primalitas *Lehman*

Bilangan prima tidak menunjukkan pola yang jelas, sehingga menentukan apakah suatu bilangan adalah prima bukanlah pekerjaan yang gampang. Suatu bilangan  $n$  dikatakan prima jika tidak memiliki pembagi  $n$  berada antara 2 dan  $\sqrt{n}$ . Pengujian keprimaan bilangan kecil relatif gampang, namun kesulitannya meningkat seiring bertambahnya nilai  $n$ .

Algoritma *Lehman*:

1. Pilih bilangan acak  $a$  dimana  $1 < a < p$  ( $p$  merupakan bilangan yang diuji keprimaannya).
2. Cari hasil dari  $a^{(p-1)/2} \bmod p$ .
3. Apabila hasil dari  $a^{(p-1)/2} \equiv \pm 1 \pmod{p}$ , maka  $p$  bukan bilangan prima
4. Apabila hasil dari  $a^{(p-1)/2} \equiv \pm 1 \pmod{p}$ , maka kesempatan  $p$  merupakan bilangan prima naik sebesar 50%.
5. Lakukan perulangan terhadap pengujian di atas sejumlah  $t$  kali (dimana  $t = 2 * \text{banyak digit } p$ ) dengan nilai  $a$  yang berbeda dan untuk persentase kebenarannya akan bertambah tinggi.

**Contoh :** Periksalah keprimaan bilangan  $p = 11$  menggunakan algoritma *Lehman*.

Hitung  $a^{(11-1)/2} \bmod 11$  dengan memakai nilai-nilai  $a$  dimana  $1 < a < 11$

$$a : 2 \quad 2^{(11-1)/2} \bmod 11 \equiv 10 \pmod{11}$$

$$a : 3 \quad 3^{(11-1)/2} \bmod 11 \equiv 1 \pmod{11}$$

$$a : 4 \quad 4^{(11-1)/2} \bmod 11 \equiv 1 \pmod{11}$$

$$a : 5 \quad 5^{(11-1)/2} \bmod 11 \equiv 1 \pmod{11}$$

Sehingga, angka 11 merupakan **Bilangan Prima** karena telah mencukupi syarat  $a^{(p-1)/2} \equiv \pm 1 \pmod{p}$  sejumlah  $t$  kali, yang dimana nilai  $t = 2 * 2 = 4$ . Dan untuk persentase kebenarannya mencapai  $50\% + 25\% + 12,5\% + 6,25\% = 93,75\%$

## 2.4. Kriptanalisis

Kriptanalisis adalah cabang dari ilmu kriptografi yang berfokus pada analisis sistem-sistem kriptografi dengan tujuan untuk mengidentifikasi kelemahan atau celah dalam algoritma atau kunci kriptografi yang digunakan. Tujuan utama kriptanalisis adalah untuk memecahkan enkripsi, mendekripsi data yang dienkripsi tanpa memiliki kunci yang benar, atau menemukan kunci yang digunakan dalam proses enkripsi. Kriptanalisis memiliki peran penting dalam pengembangan dan peningkatan keamanan sistem kriptografi, karena dengan mengetahui kelemahan yang mungkin ada, kita dapat mengambil langkah-langkah untuk memperbaikinya dan meningkatkan keamanan sistem secara keseluruhan. Kriptanalisis dapat dilakukan dengan berbagai metode, seperti memfaktorkan kunci publik, komputasi, statistik, dan serangan *brute-force*.

Teknik pemfaktoran kunci publik merupakan satu diantara beberapa metode kriptanalisis yang khusus dipakai untuk mendapatkan faktor-faktor prima dari bilangan-bilangan yang dipakai dalam algoritma kriptografi kunci publik. Metode ini sering digunakan dalam konteks algoritma kriptografi asimetris, dimana kunci publik dan juga kunci privat dipasangkan dengan menggunakan bilangan prima yang sangat besar sebagai dasar pembentukannya. Penemuan faktor-faktor prima dari kunci publik dapat memungkinkan penyerang untuk mendekripsi pesan atau bahkan menghitung kunci privat yang sesuai. Ada beberapa jenis teknik pemfaktoran kunci publik yang sering digunakan dalam kriptanalisis, seperti metode *Pollard's Rho*, metode pemfaktoran *Fermat*, metode *Kraitchik*, metode *Quadratic Sieve*, metode *Elliptic Curve Factorization* (ECM).

## 2.5. Pemfaktoran Metode *Kraitichik*

Metode *Kraitichik* adalah salah satu teknik dalam kriptanalisis yang telah terbukti efektif dalam mengungkap kerentanan pada berbagai algoritma kriptografi. Metode ini berfokus pada pemfaktoran nilai  $n$  berubah menjadi dua faktor primanya. Pada tahun 1945, Maurice Kraitichik mengembangkan metode ini berdasarkan metode dasar *Fermat*. Metode *Kraitichik* mendekomposisi  $n$  menjadi  $x$  dan  $y$  sehingga  $x^2 \equiv y^2 \pmod{n}$ . Kraitichik mengamati bahwa  $x^2 - y^2$  adalah kelipatan dari  $n$ . Oleh karena itu, untuk mencari faktor dari  $n$ , Kraitichik menggunakan persamaan :

$$kn = x^2 - y^2$$

dengan  $k$  adalah bilangan bulat positif.

**Contoh :** Cari faktor dari 41323 dengan memakai metode *Kraitichik* dengan  $k$  merupakan sebuah bilangan ganjil yang prima.

1. Dapatkan nilai  $x_0$  dengan melakukan pengakaran  $n = 41323$

$$x_0 = \sqrt{n} = \sqrt{41323} = 203,28... = 203$$

2. Tetapkan  $k = 3$ , kemudian cari nilai  $y^2$  dengan  $(m = 1, 2, 3, \dots)$ :

$$x^2 - k \cdot n = y^2 \rightarrow (x_0 + m)^2 - k \cdot n = y^2$$

$$m = 1 \rightarrow (203 + 1)^2 - 3 \cdot 41323 = -82353$$

Karena didapatkan hasil yang negatif, maka angka tersebut tidak dapat di akarkan. Oleh sebab itu, nilai  $y^2$  dihitung dengan menaikkan nilai dari  $m$  hingga mendapatkan nilai yang positif dan merupakan akar sempurna:

$$m = 338 \rightarrow (203 + 338)^2 - 3 \cdot 41323 = 168712$$

$$m = 339 \rightarrow (203 + 339)^2 - 3 \cdot 41323 = 169795$$

$$m = 340 \rightarrow (203 + 340)^2 - 3 \cdot 41323 = 170880$$

$$m = 341 \rightarrow (203 + 341)^2 - 3 \cdot 41323 = 171967$$

$$m = 342 \rightarrow (203 + 342)^2 - 3 \cdot 41323 = 173059 = 416^2$$

Sehingga  $3 \cdot 41323 = 545^2 - 416^2$

$$= (545 + 416)(545 - 416)$$

$$\begin{aligned} 41323 &= 961 * 129/3 \\ &= 961 * 43 \end{aligned}$$

Jadi,  $n = 41323 = 961 * 43$   
 $= 31^2 * 43$

## 2.6. Metode Pemfaktoran *Fermat*

Metode faktorisasi *Fermat* adalah teknik yang digunakan untuk memecahkan bilangan bulat menjadi faktor-faktor primanya. Metode ini ditemukan oleh Pierre de Fermat. Metode ini didasarkan pada prinsip bahwa setiap bilangan ganjil dapat dinyatakan sebagai selisih dua kuadrat.

Metode ini beroperasi dengan cara sebagai berikut :

1. Misalkan  $n$  adalah bilangan bulat yang akan difaktorkan.
2. Hitung  $r = \lceil \sqrt{n} \rceil$
3. Hitung  $s = r^2 - n$
4. Jika  $s$  bukan kuadrat sempurna, maka lakukan langkah 4a, 4b, 4c :
  - a. Tambah  $r$  dengan 1
  - b. Hitung  $s = r^2 - n$
  - c. Jika  $s$  adalah kuadrat sempurna, teruskan ke langkah 5, namun apabila tidak, lanjut ke langkah 4.
5. Didapatkan  $r - \sqrt{s}$  sebagai faktor dari  $n$ .

**Contoh :** Faktorkan 41323

1.  $n = 41323$
2. Hitung  $r = \lceil \sqrt{n} \rceil$   
 $= \lceil \sqrt{41323} \rceil$

$$= \lceil 203,23 \rceil = 204$$

3. Hitung  $s = r^2 - n$

$$= 204^2 - 41323$$

$$= 293$$

4. Karena  $s$  bukan akar sempurna, maka dilakukan :

a.  $r = r + 1$

$$= 204 + 1 = 205$$

b.  $s = r^2 - n \rightarrow s = 205^2 - 41323$

$$= 702$$

c.  $s$  masih bukan akar sempurna, maka tambah  $r$  dengan 1 hingga  $s$  menjadi akar sempurna.

a.  $r = r + 298$

$$= 204 + 298 = 502$$

b.  $s = r^2 - n \rightarrow s = 502^2 - 41323$

$$= 210681$$

c.  $s$  adalah akar sempurna

5. Dapatkan  $r - \sqrt{s} = 502 - \sqrt{210681}$

$$= 502 - 459$$

$$= 43$$

Sehingga didapatkan **43** sebagai salah satu faktor dari 41323. Dan faktor lainnya adalah  $41323 / 43 = \mathbf{961}$ .

## 2.7. Penelitian Relevan

1. Menurut penelitian yang dilakukan oleh Singly Purba dengan judul “KRIPTANALISIS KUNCI PUBLIK ALGORITMA RABIN MENGGUNAKAN METODE KRAITCHIK” mendapatkan kesimpulan bahwa peningkatan nilai dan panjang kunci publik  $n$  tidak selalu berbanding lurus dengan waktu pemfaktoran yang semakin lama. Peneliti juga memberikan saran untuk melakukan pemecahan kunci publik algoritma lain menggunakan metode yang lainnya selain menggunakan metode *Kraitchik*.
2. Menurut penelitian yang dilakukan oleh Adhitya Ramadhanus, dengan judul “Perbandingan Kriptanalisis RSA dan Schmidt Samoa menggunakan metode faktorisasi elliptic curve dan quadratic sieve” mendapatkan kesimpulan bahwa secara umum, metode *quadratic sieve* mempunyai durasi pengerjaan yang lebih cepat daripada dengan metode *elliptic curve* selain untuk kunci *Schmidt-Samoa* yang memiliki ukuran 96 bit.
3. Berdasarkan penelitian yang dilakukan oleh Qasem Abu Al-Haija dkk, dengan judul “COST-EFFECTIVE FPGA IMPLEMENTATION OF PARALLEL SCHMIDT-SAMOA CRYPTOSYSTEM (SSC)” mereka menggunakan implementasi teknologi FPGA untuk *Schmidt-Samoa Cryptosystem* (SSC) dan mendapatkan hasil menarik bagi perancang FPGA algoritma kriptografi dengan skalabilitas area desain yang luas, proses enkripsi/dekripsi yang cepat, dan konsumsi daya yang rendah.
4. Berdasarkan penelitian yang dilakukan oleh Qasem Abu Al-Haija dkk, dengan judul “Implementing a lightweight Schmidt-Samoa cryptosystem (SSC) for sensory communications” mereka menggunakan *Schmidt-Samoa Cryptosystem* (SSC) 128-bit untuk mengamankan komunikasi data jaringan sensor nirkabel (*wireless sensor network* (WSN)) dari serangan eksternal. Hasil eksperimen menunjukkan bahwa Kriptoprosesor SSC yang diusulkan mencatat hasil yang menarik untuk aplikasi berdaya rendah seperti keamanan jaringan sensor nirkabel.

5. Berdasarkan penelitian yang dilakukan oleh Maya Silvi Lydia dkk, dengan judul “FACTORIZATION OF SMALL RPRIME RSA MODULUS USING FERMAT’S DIFFERENCE OF SQUARES AND KRAITCHIK’S ALGORITHMS IN PYTHON” menunjukkan bahwa algoritma *Fermat's difference of squares* dan algoritma *Kraitchik* dapat digunakan secara efektif sebagai metode untuk memfaktorkan modulus kecil dari RPrime RSA. Waktu yang dibutuhkan kedua algoritma untuk memfaktorkan sebagian besar berbanding lurus dengan ukuran  $n$ . Namun, algoritma *Fermat* jauh lebih cepat daripada algoritma *Kraitchik* ketika memfaktorkan enam digit hingga sebelas digit  $n$ , *Fermat* sekitar 24 hingga 550 kali lebih cepat daripada *Kraitchik*.

## **BAB III**

### **ANALISIS DAN PERANCANGAN**

#### **3.1. Analisis**

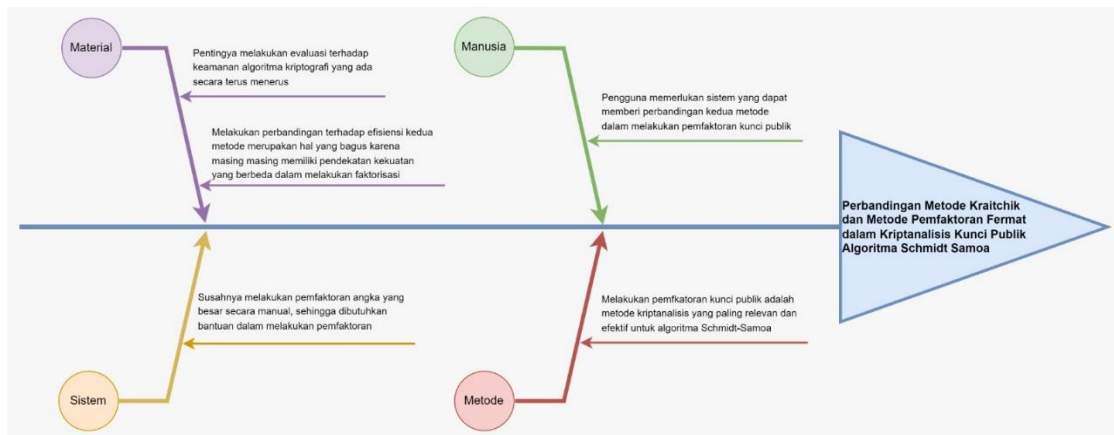
Analisis dan perancangan merupakan sebuah langkah-langkah yang dilakukan dalam pengembangan sistem. Tujuan analisis adalah untuk membuat sebuah gambaran secara umum terhadap perangkat lunak. Hasil analisis sendiri berfungsi untuk menjadi pedoman penting untuk proses desain atau perancangan perangkat lunak untuk memenuhi kebutuhan sistem yang akan dikembangkan.

##### **3.1.1. Analisis Masalah**

Algoritma kriptografi sangat sering digunakan saat ini untuk mengamankan data terutama di era saat ini yang didominasi oleh pertukaran informasi digital. Namun, algoritma kriptografi juga perlu dievaluasi terhadap ketahanan dari serangan kriptografi dari pihak yang tidak bertanggung jawab. Sebab itu, pada penelitian ini akan dilakukan pengujian terhadap sebuah algoritma kriptografi yaitu algoritma *Schmidt-Samoa* menggunakan metode pemecahan kunci *Kraitchik* dan metode pemfaktoran *Fermat*, kemudian membuat perbandingan mengenai efisiensi kedua metode tersebut dalam memecahkan kunci publik algoritma *Schmidt-Samoa*.

Pada Gambar 3. 1 dibawah ini didapatkan bahwa masalah utama yang akan diselesaikan pada penelitian ini terlihat pada bagian sebelah kanan (kepala ikan) yaitu melakukan kriptanalisis terhadap kunci sebuah publik algoritma kriptografi asimetris. Terdapat empat bagian penyebab dari masalah yang mengarah pada tulang utama berbentuk horizontal yaitu terdiri dari manusia, sistem, metode, dan material. Setiap detail dari penyebab masalah digambarkan menggunakan tanda panah yang mengarah ke masing-masing faktor.





**Gambar 3. 1 Diagram Ishikawa Masalah Penelitian**

### 3.1.2. Analisis Kebutuhan

Analisis ini terdiri dari dua komponen utama, yaitu analisis kebutuhan dan analisis kebutuhan non-fungsional. Terdapat dua komponen utama dalam analisis kebutuhan, yaitu analisis kebutuhan fungsional dan analisis kebutuhan non-fungsional. Analisis kebutuhan fungsional menjelaskan fitur-fitur yang berkaitan dengan hasil akhir yang ingin dicapai dari sistem yang akan dikembangkan pada penelitian ini. Analisis kebutuhan non-fungsional memaparkan fitur tambahan seperti kinerja, kemudahan penggunaan, dan fitur lainnya yang dimaksudkan untuk mendukung sistem yang akan dibangun pada penelitian ini.

#### 3.1.2.1 Analisis Kebutuhan Fungsional

Kebutuhan fungsional menentukan tugas sistem dan tindakannya. Kebutuhan fungsional mencakup penjelasan tentang kegiatan yang dapat dilakukan sistem yang akan dibangun pada penelitian ini.

Kebutuhan fungsional pada sistem yang akan dibangun ialah:

1. Melakukan pembangkitan bilangan prima

Sistem mampu melakukan pembangkitan bilangan prima  $p$  dan  $q$ , yang bakal digunakan untuk membangkitkan kunci publik nantinya.

2. Menerima kunci publik

Sistem mampu menerima kunci publik berupa angka yang diberi oleh pengguna.

3. Melakukan kriptanalisis

Sistem mampu melakukan faktorisasi terhadap kunci publik yang diberikan oleh pengguna.

4. Memberi hasil faktorisasi

Sistem dapat menghasilkan 2 buah angka yang merupakan hasil faktorisasi dari kunci publik yang diberikan oleh pengguna.

### 3.1.2.2 Analisis Kebutuhan Non-Fungsional

Kebutuhan non-fungsional adalah kebutuhan yang berguna sebagai pelengkap atau fitur-fitur tambahan pada suatu sistem yang mempengaruhi kinerja sistem yang akan dibangun.

Berikut adalah aspek-aspek non-fungsional yang perlu diperhatikan pada sistem yang akan dikembangkan:

1. Performa

Sistem mampu memecahkan kunci publik algoritma *Schmidt-Samoa* dan mendapatkan kunci privatnya.

2. Efisiensi

Sistem yang dirancang nantinya akan mudah dipahami dan tidak membingungkan pengguna .

3. Perangkat Keras

Pengujian yang dilaksanakan pada penelitian ini dilakukan dengan spesifikasi sebagai berikut :

- Processor : Intel I® Core(TM) i3-8145U
- RAM : 8 GB
- GPU : Intel® UHD Graphics 620

### 3.2. Perancangan Sistem

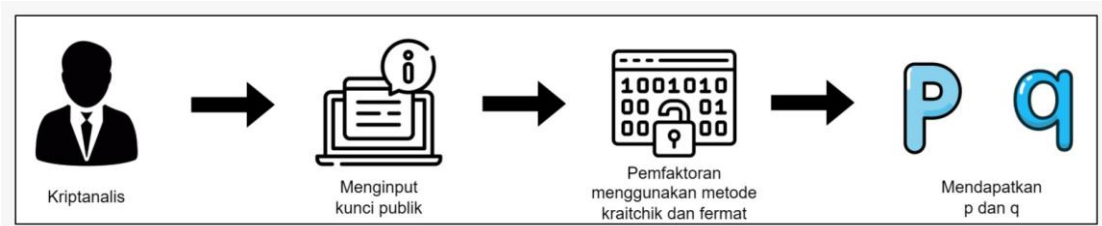
Perancangan sistem ialah proses yang dilakukan untuk merancang sistem dengan tujuan membuat sistem tersebut beroperasi dengan fokus pada meningkatkan tingkat efektivitas dan efisiensi. Dalam konteks penelitian ini, perancangan sistem akan memanfaatkan berbagai jenis diagram, termasuk diagram umum, diagram urutan, diagram aktivitas, diagram use case, dan diagram alir (*flowchart*).

#### 3.2.1. Diagram Umum

Diagram umum ialah gambaran umum yang mendefinisikan seperti apa sistem yang dikembangkan nantinya. Diagram umum dalam penelitian adalah diagram umum kriptanalisis.

##### 3.2.1.1. Diagram Umum Kriptanalisis

Pada Gambar 3.2 menunjukkan diagram umum untuk tahap kriptanalisis.



**Gambar 3. 2 Diagram Umum Kriptanalisis**

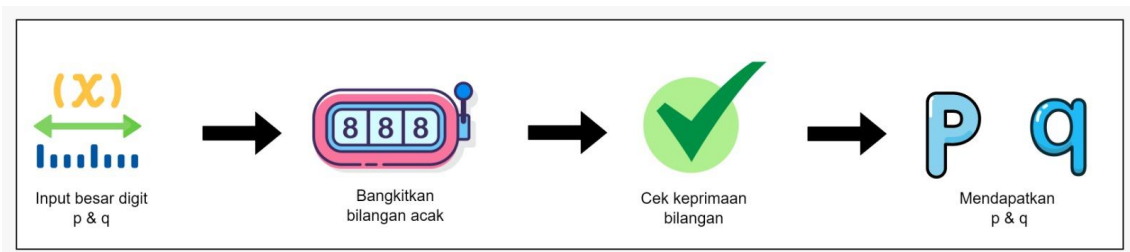
Pada Gambar 3.2 menunjukkan bagaimana proses kriptanalisis akan dilakukan pada sistem yang akan dikembangkan. Berikut merupakan langkah-langkah yang ditunjukkan pada Gambar :

1. Kriptanalisis akan mencari informasi berupa kunci publik, lalu memasukkannya ke dalam sistem.
2. Sistem akan melakukan pemfaktoran terhadap kunci publik dengan memakai metode *Kraitichik* dan metode pemfaktoran *Fermat* untuk memperoleh nilai  $p$  dan  $q$ .

$q$  yang berfungsi untuk membangkitkan kunci privat dari algoritma kriptografi tersebut.

### 3.2.1.1. Diagram Umum Pembangkitan Kunci

Pada Gambar 3.3 menunjukkan diagram umum tahap pembangkitan bilangan  $p$  dan  $q$  yang nantinya akan menjadi kunci.



**Gambar 3.3 Diagram Umum Pembangkitan Kunci**

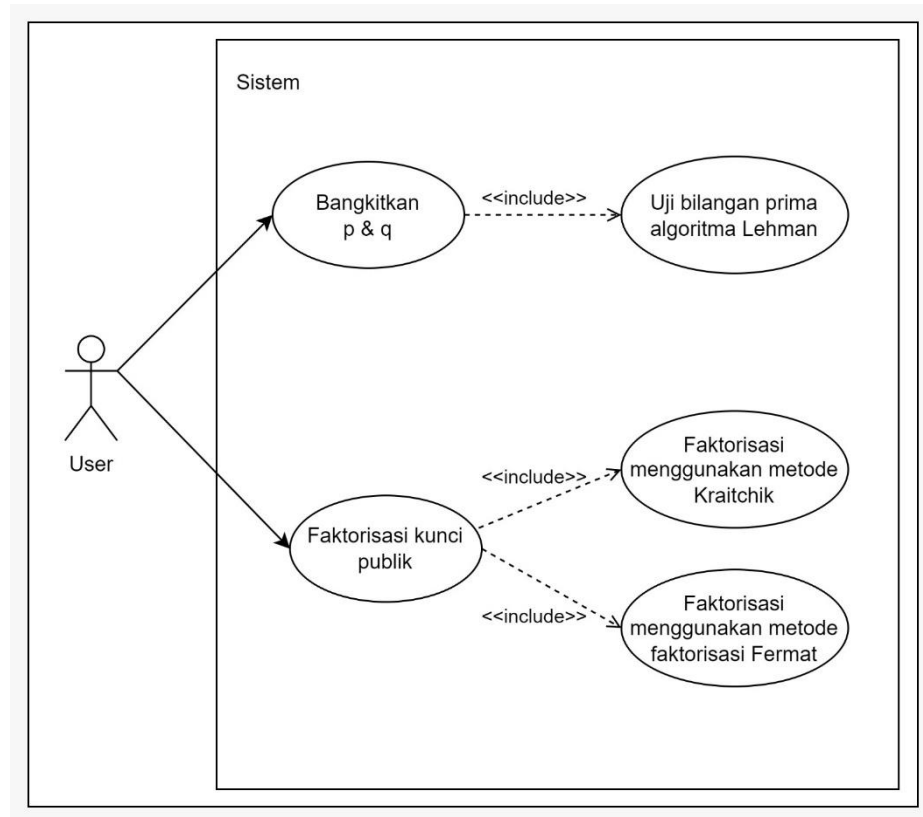
Pada Gambar menunjukkan bagaimana proses pembangkitan bilangan  $p$  dan  $q$  yang akan dilakukan pada sistem yang akan dikembangkan. Berikut merupakan langkah-langkah yang ditunjukkan pada Gambar :

1. User akan menginput besar digit bilangan  $p$  dan  $q$  yang diinginkan.
2. Sistem akan membangkitkan bilangan secara acak, dengan besar digit sesuai yang diinginkan oleh user.
3. Sistem melakukan tes keprimaan terhadap bilangan yang dibangkitkan.
4. Jika bilangan sudah merupakan bilangan prima, maka sistem mengeluarkan  $p$  dan  $q$ .

### 3.2.2. Use Case Diagram

*Use case diagram* merupakan jenis graf pada pemodelan perangkat lunak yang dipakai untuk memvisualisasikan hubungan antara sistem perangkat lunak dan berbagai aktor (pengguna atau entitas eksternal lainnya) yang terlibat dalam sistem. Use case diagram membantu mengidentifikasi, mendeskripsikan, serta memahami bagaimana sistem yang

akan dibangun beroperasi dalam konteks situasi dunia nyata. *Use case diagram* pada sistem yang akan dikembangkan pada penelitian ini digambarkan pada Gambar 3. 4.



**Gambar 3. 4 Use Case Diagram**

Pada Gambar 3.4 terdapat seorang aktor yang dapat menjalankan sistem yang akan dirancang nantinya. User dapat melakukan proses pembangkitan bilangan  $p$  dan  $q$  yang akan digunakan nantinya untuk membangkitkan kunci publik. User juga dapat melakukan proses kriptanalisis pada sistem dengan memasukkan kunci publik yang nantinya akan di faktorkan menggunakan metode *Kraitchik* dan metode pemfaktoran *Fermat*. Setelah memfaktorkan kunci publik, user dapat mendapatkan nilai  $p$  dan  $q$  dan juga waktu faktorisasi dari kedua metode faktorisasi tersebut.

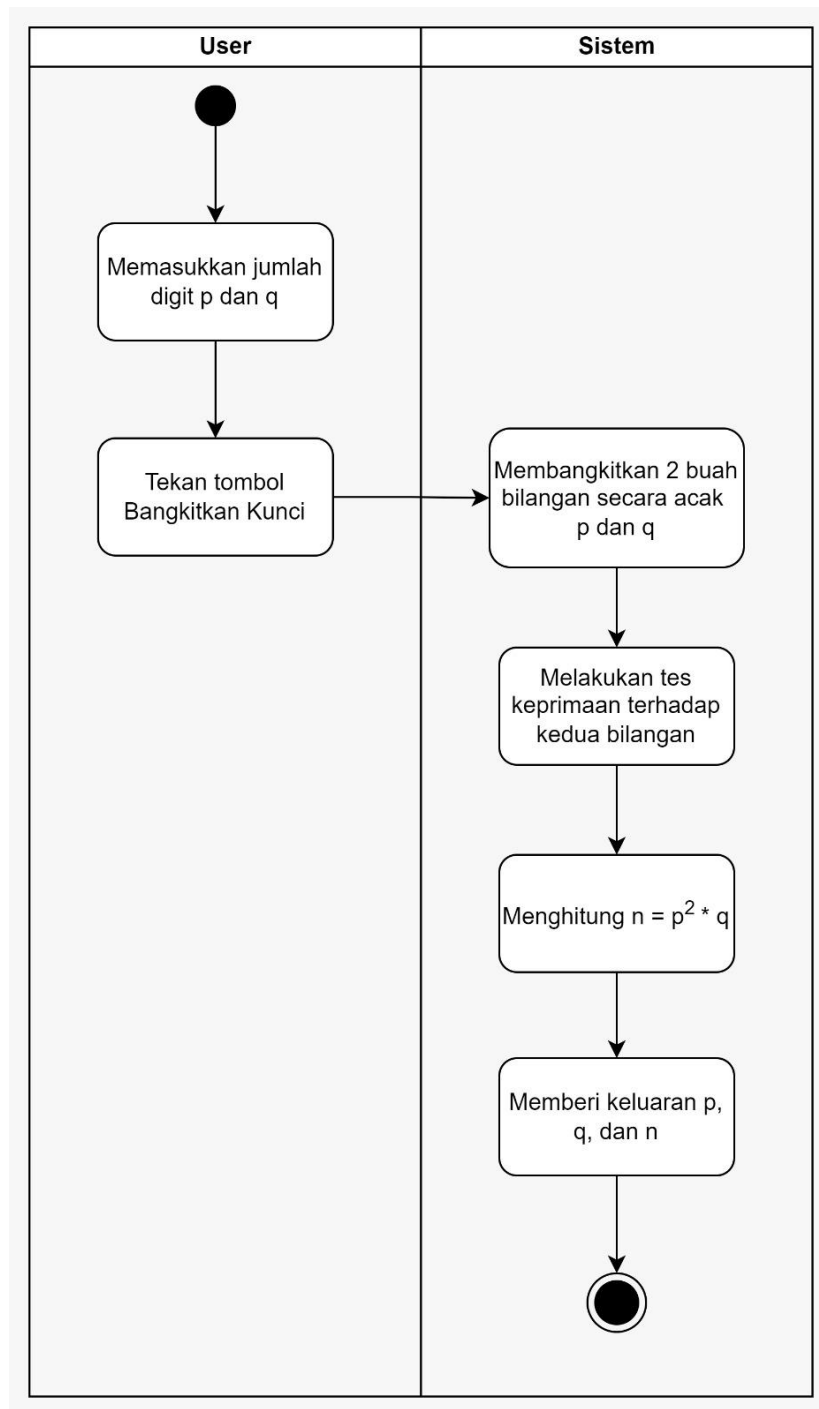
### 3.2.3. Activity Diagram

Activity diagram ialah graf pada representasi perangkat lunak yang digunakan untuk mengilustrasikan aliran aktivitas dalam suatu sistem atau tahapan. Diagram ini

memberikan bantuan dalam memvisualisasikan urutan langkah-langkah atau aktivitas yang dikerjakan dalam suatu proses. Pada sistem yang dikembangkan pada penelitian ini, terdapat dua buah *activity diagram*, yakni *activity diagram* pada tahap membangkitkan kunci publik, dan *activity diagram* pada tahap kriptanalisis kunci publik.

### 3.2.3.1 Activity Diagram Pembangkitan Kunci Publik

Pada Gambar 3.5 menampilkan graf aktivitas pada fase pembangkitan kunci publik yang digunakan pada sistem yang dikembangkan pada penelitian ini. Activity diagram di fase ini memiliki dua buah kotak, yakni sebelah kiri menunjukkan aktivitas yang dilakukan oleh user pada sistem dan kotak sebelah kanan menunjukkan respon apa yang dilakukan oleh sistem. User diminta untuk menentukan jumlah digit untuk bilangan  $p$  dan  $q$  yang akan dibangkitkan. Setelah itu sistem akan merespon dengan melakukan pembangkitan dua bilangan secara acak. Lalu setelah bilangan dibangkitkan, akan dilakukan pengecekan terhadap kedua bilangan tersebut untuk memastikan kedua bilangan tersebut adalah bilangan prima. Kemudian sistem menghitung nilai kunci publik  $n = p^2 * q$ , dan memberikan nilai  $p$ ,  $q$ , dan  $n$  kepada user.

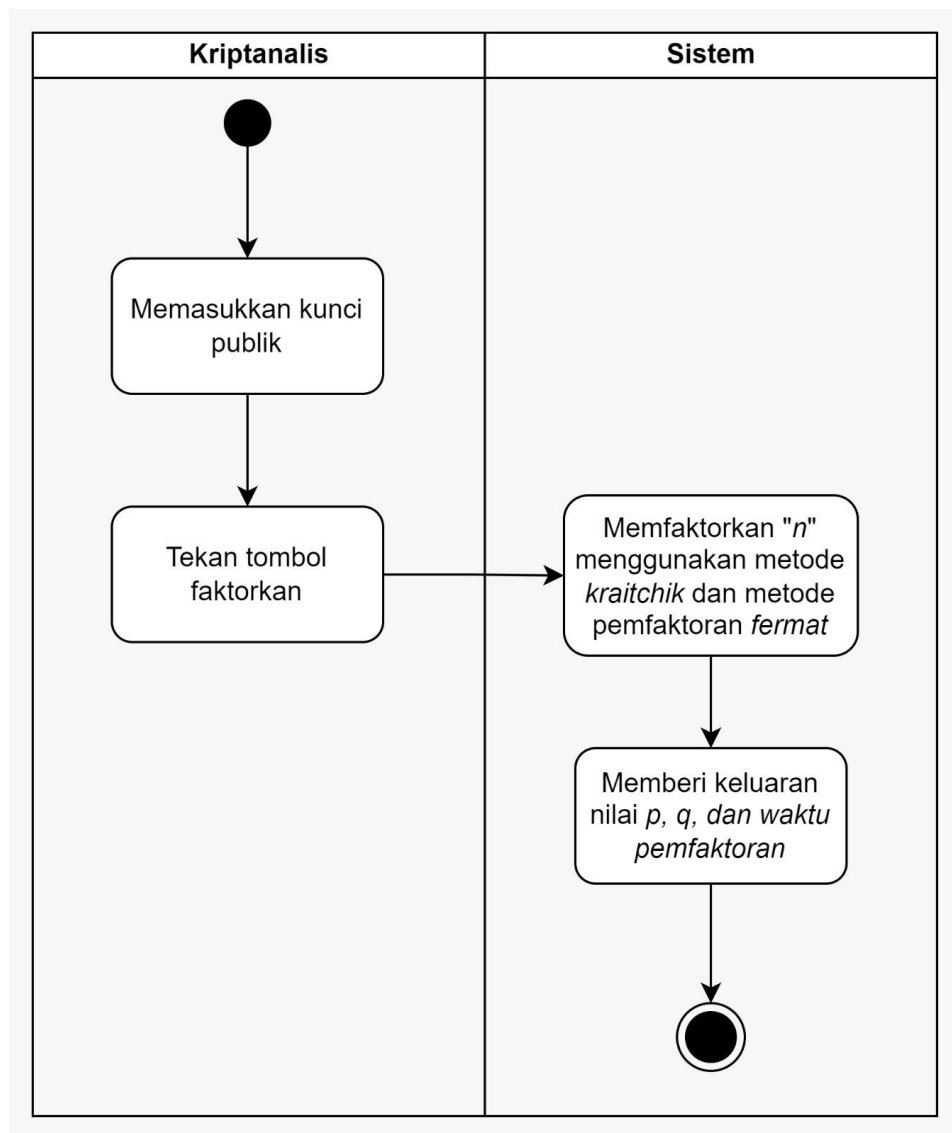


**Gambar 3. 5 Activity Diagram Pembangkitan Kunci**

### 3.2.3.2 Activity Diagram Kriptanalisis Kunci Publik

Pada Gambar 3.6 menunjukkan graf aktivitas pada fase kriptanalisis kunci publik yang digunakan pada sistem yang dikembangkan pada penelitian ini. Activity diagram di fase ini memiliki dua buah kotak, yakni sebelah kiri menunjukkan aktivitas yang dilakukan oleh kriptanalisis pada sistem dan kotak sebelah kanan menunjukkan respon apa yang dilakukan oleh sistem. Kriptanalisis harus memasukkan kunci publik yang nantinya akan di kriptanalisis. Setelah itu sistem akan merespon dengan melakukan proses pemfaktoran terhadap kunci publik dimasukkan oleh kriptanalisis menggunakan metode *Kraitchik* dan metode pemfaktoran *Fermat*. Setelah proses pemfaktoran selesai, sistem akan memberikan keluaran berupa dua buah bilangan yaitu  $p$  dan  $q$ , serta lama waktu proses pemfaktoran kedua metode.





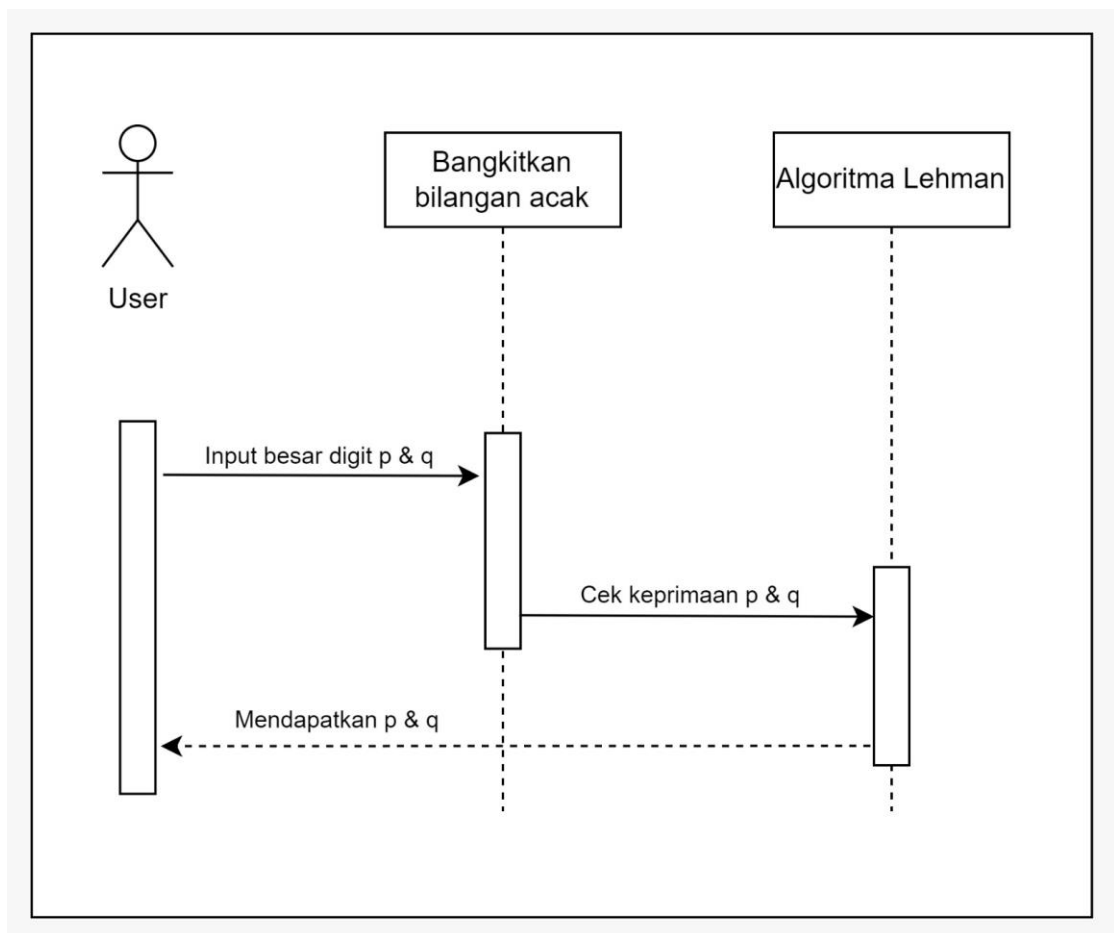
**Gambar 3. 6 Activity Diagram Kriptanalisis Kunci Publik**

### 3.2.4. Sequence Diagram

Diagram urutan ialah graf pemodelan perangkat lunak yang digunakan untuk memvisualisasikan hubungan antar objek dalam suatu sistem dalam urutan waktu tertentu. Diagram ini memvisualisasikan bagaimana objek berhubungan satu dengan lainnya dan berinteraksi dalam konteks use case. Dalam penelitian ini, sistem yang akan dibuat memiliki sebuah diagram urutan yakni diagram urutan kriptanalisis.

#### 3.2.4.1 Sequence Diagram Pembangkitan Kunci

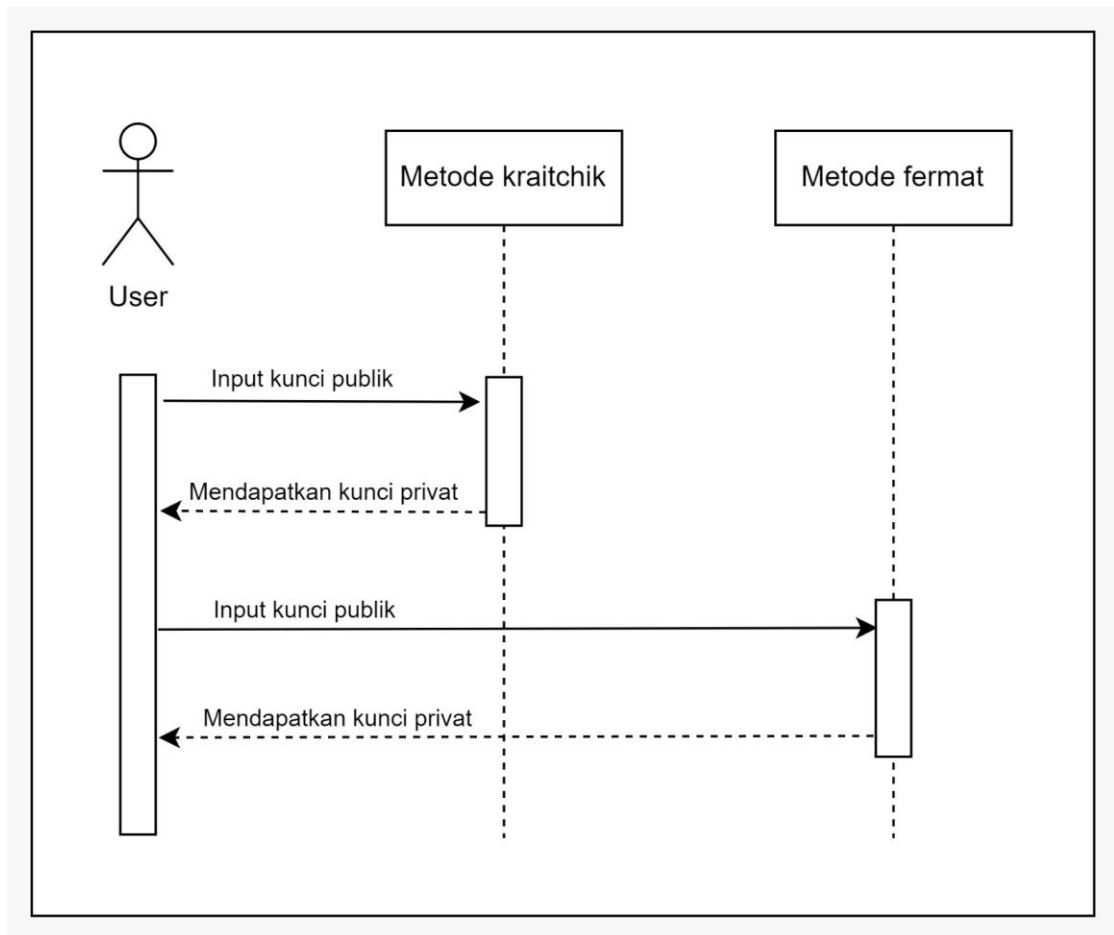
Pada Gambar 3.7 menunjukkan bagaimana sequence diagram untuk interaksi antara user dan sistem pada fase pembangkitan bilangan  $p$  dan  $q$ .



**Gambar 3. 7 Sequence Diagram Pembangkitan Kunci**

#### 3.2.4.2 Sequence Diagram Kriptanalisis Kunci Publik

Pada Gambar 3.8 menunjukkan bagaimana sequence diagram untuk interaksi antara user dan sistem pada fase kriptanalisis kunci publik.



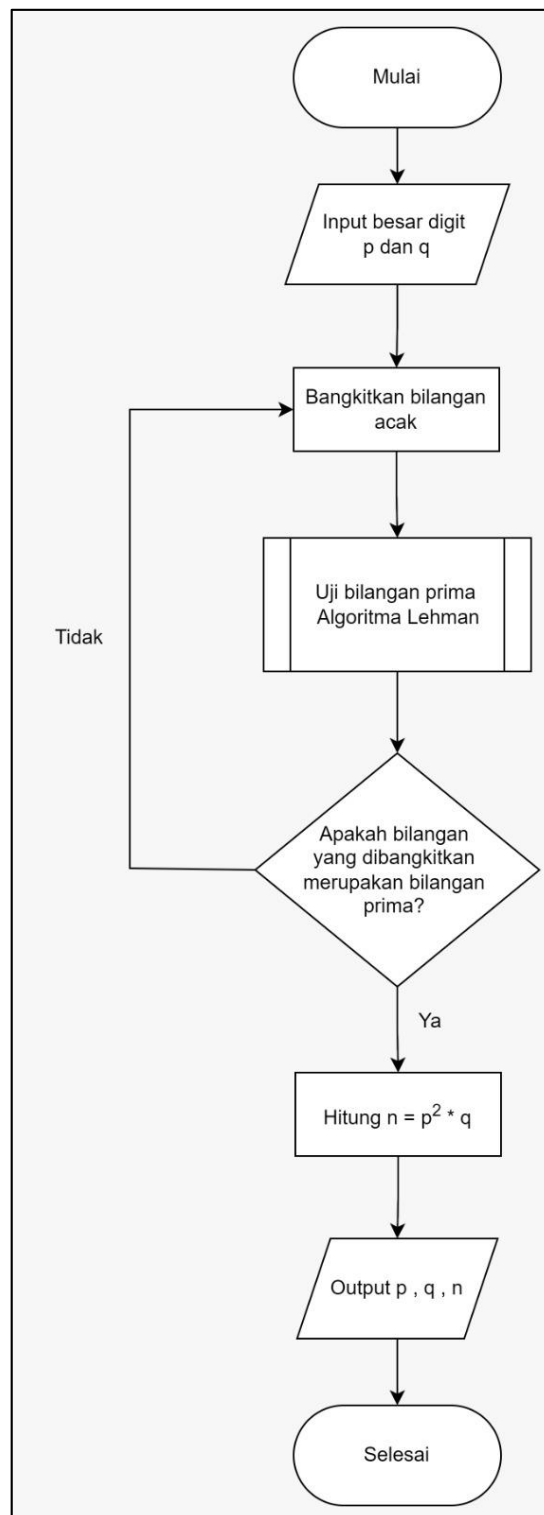
**Gambar 3. 8 Sequence Diagram Kriptanalisis Kunci Publik**

### 3.2.5. Diagram Alir (*flowchart*)

*Flowchart* adalah representasi visual dari proses yang digunakan dalam suatu proses atau sistem. Diagram alir digunakan untuk memodelkan, merancang, dan mendokumentasikan alur kerja, algoritma, atau proses dengan cara yang mudah dimengerti bagi pihak yang terlibat dalam proses tersebut. Diagram alir pada penelitian ini terdiri dari diagram alir kriptanalisis, diagram alir metode *Kraitichik*, dan diagram alir metode pemfaktoran *Fermat*.

### 3.2.5.1 Diagram Alir Pembangkitan Kunci Publik

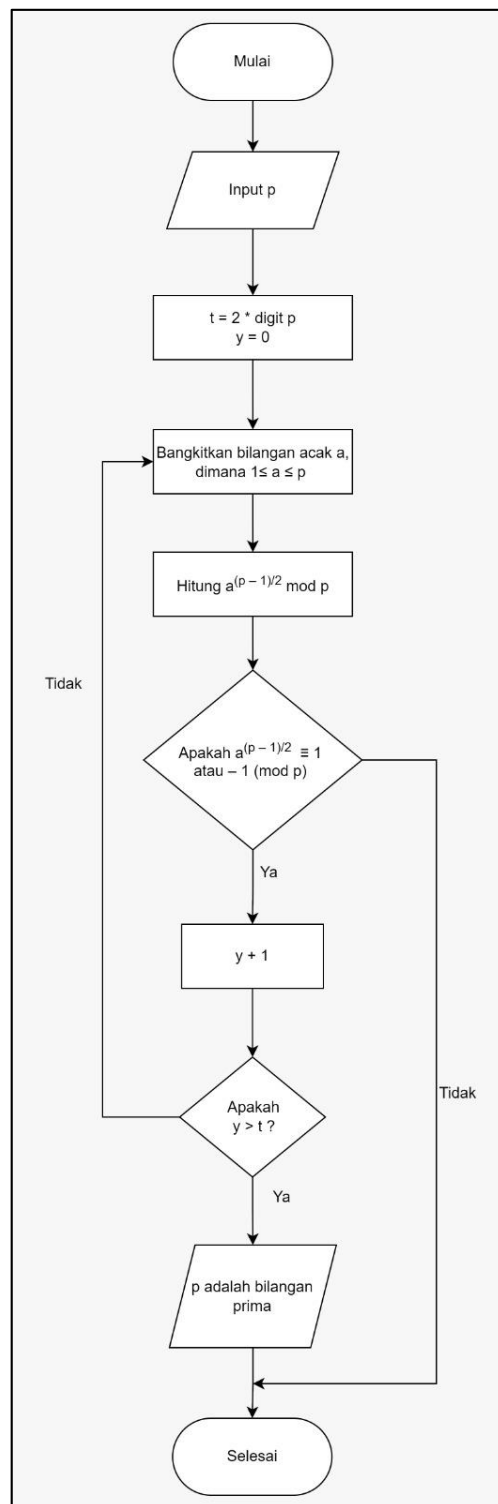
Pada Gambar 3.9 menunjukkan diagram alir untuk fase atau proses pembangkitan kunci publik. Fase pembangkitan kunci publik diawali dengan meminta user untuk menentukan besar digit bilangan  $p$  dan  $q$  yang akan di bangkitkan. Lalu sistem akan membangkitkan dua buah bilangan acak. Setelah kedua bilangan dibangkitkan, maka akan dilakukan tes keprimaan terhadap kedua bilangan tersebut. Setelah dipastikan bahwa kedua bilangan tersebut merupakan bilangan prima, maka akan dilakukan perhitungan terhadap kunci publik ( $n$ ). Setelah itu sistem akan memberi keluaran nilai  $p$ ,  $q$ , dan  $n$ .



**Gambar 3. 9 Diagram Alir Pembangkitan Kunci**

### 3.2.5.2 Diagram Alir Algoritma Tes Primalitas *Lehman*

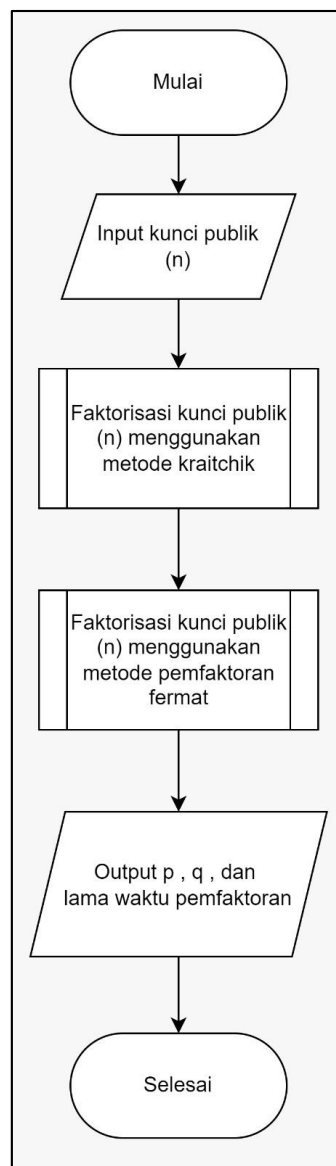
Flowchart untuk sub program dari diagram alir pembangkitan kunci publik digambarkan pada Gambar 3.10, yakni Algoritma Tes Primalitas *Lehman*. Diagram ini menjelaskan bagaimana proses algoritma *Lehman* bekerja nantinya untuk melakukan tes keprimaan terhadap bilangan acak yang dibangkitkan. Perhitungan dimulai setelah bilangan dibangkitkan (misalnya  $p$ ), lalu akan dihitung nilai  $t = \frac{1}{2} p$  dan  $y = 0$  yang akan digunakan sebagai parameter untuk perulangan yang akan dilakukan pada langkah selanjutnya. Kemudian bangkitkan nilai  $a$  dimana  $1 \leq a \leq p$ , yang akan digunakan untuk menghitung  $a^{(p-1)/2} \bmod p$ . Apabila  $a^{(p-1)/2} \bmod p \equiv \pm 1$ , maka nilai  $y$  bertambah 1, namun jika tidak, maka perhitungan selesai karena bilangan yang diperiksa bukan bilangan prima, melainkan bilangan komposit. Perhitungan tersebut dilakukan hingga nilai  $y$  lebih besar dari  $t$ . Jika nilai  $y$  sudah lebih besar dari  $t$ , maka bilangan tersebut dapat dipastikan sebagai bilangan prima.



**Gambar 3. 10 Diagram Alir Algoritma Tes Primalitas *Lehman***

### 3.2.5.3 Diagram Alir Kriptanalisis

Diagram alir untuk proses kriptanalisis digambarkan pada Gambar 3.11. Diagram alir ini menjelaskan bagaimana nantinya proses kriptanalisis berjalan pada sistem yang dikembangkan. Masukan yang dibutuhkan di proses ini adalah kunci publik yang nantinya akan difaktorkan. Lalu akan dilanjutkan pada sub program pemfaktoran menggunakan metode *Kraitchik* dan metode pemfaktoran *Fermat*. Hingga keluaran yang diberikan adalah  $p$ ,  $q$ , dan lama waktu pemfaktoran.

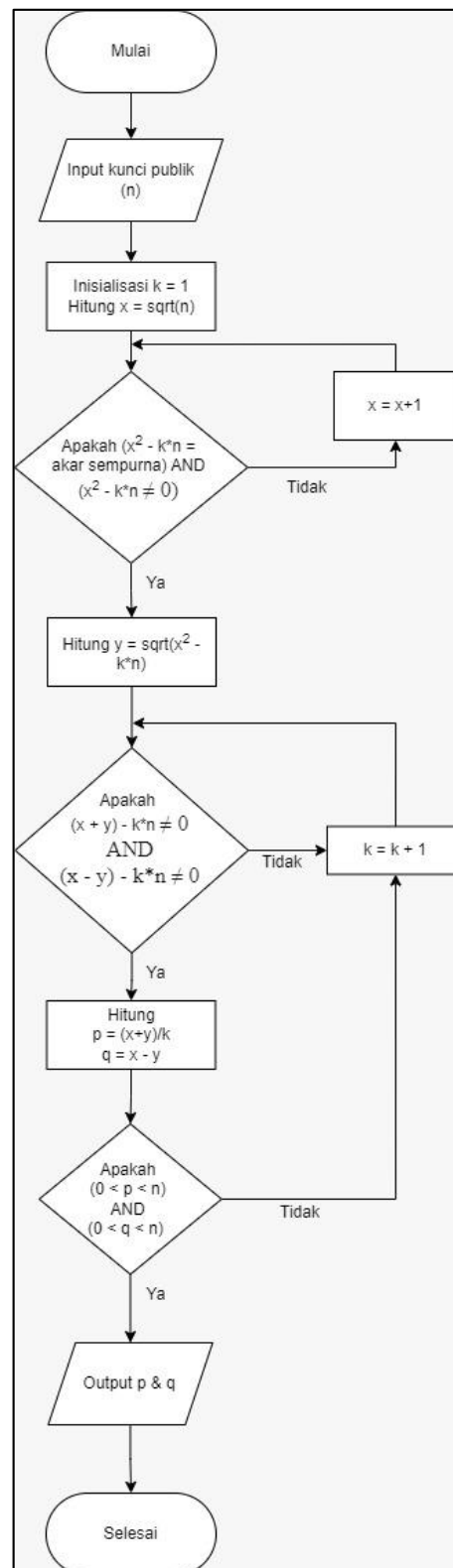


**Gambar 3. 11 Diagram Alir Kriptanalisis**



#### 3.2.5.4 Diagram Alir Metode *Kraitchik*

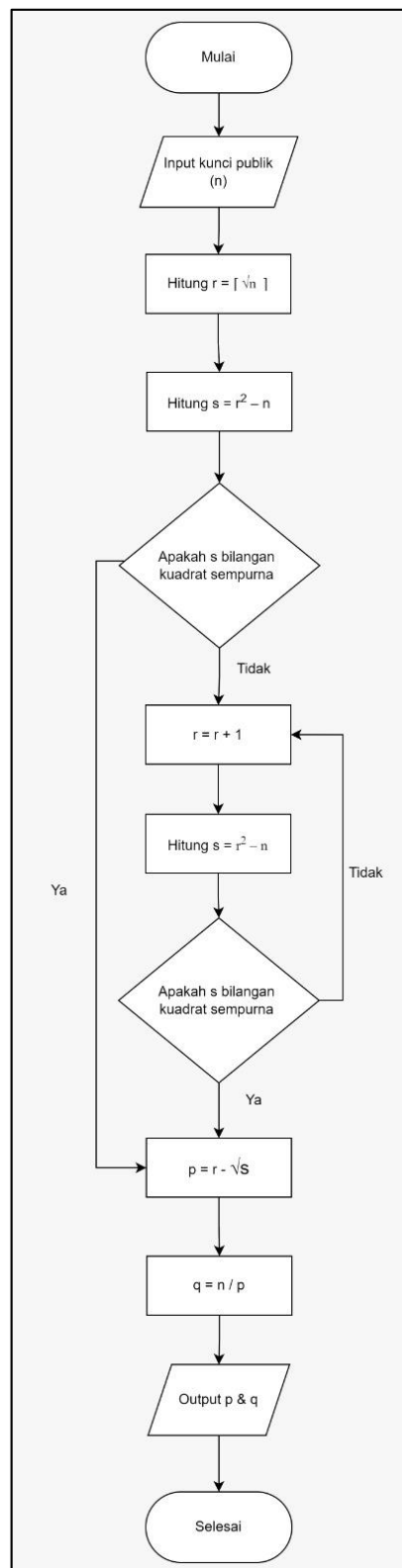
Gambar 3.12 merupakan diagram alir untuk sub program dari flowchart kriptanalisis, yaitu metode *Kraitchik*. Diagram ini menjelaskan bagaimana metode *Kraitchik* bekerja nantinya pada sistem yang akan dibuat. Masukan yang dibutuhkan di tahapan ini adalah kunci publik ( $n$ ). Lalu dilanjutkan dengan menginisialisasi nilai  $k = 1$  dan menghitung nilai  $x = \sqrt{n}$ . Kemudian menghitung nilai  $y = \sqrt{x^2 - k*n}$ , dan dilakukan pengecekan apakah  $y$  adalah akar sempurna dan  $y \neq 0$ . Jika ternyata  $y$  bukan akar sempurna atau  $y = 0$ , maka nilai  $x$  ditambah dengan 1. Namun jika  $y$  adalah akar sempurna dan  $y \neq 0$ , maka akan dilanjutkan ke langkah berikutnya. Langkah berikutnya adalah melakukan pengecekan benarkah  $(x + y) - k*n \neq 0$  DAN  $(x - y) - k*n \neq 0$ . Jika hasilnya tidak sesuai, maka nilai  $k$  ditambah dengan 1, namun jika hasilnya sesuai akan dilakukan perhitungan  $p = (x + y) / k$  dan  $q = x - y$ . Lalu dilakukan pemeriksaan apakah nilai  $p$  dan  $q$  lebih kecil dari  $n$  dan lebih besar dari 0. Jika hasilnya tidak sesuai, maka nilai  $k$  ditambah dengan 1. Namun jika hasilnya sesuai, maka nilai  $p$  dan  $q$  yang diinginkan telah didapatkan.



**Gambar 3. 12 Diagram Alir Metode *Kraitchik***

### 3.2.5.5 Diagram Alir Metode Pemfaktoran *Fermat*

Gambar 3.13 adalah diagram alir untuk salah satu sub program dari flowchart kriptanalisis, yaitu metode pemfaktoran *Fermat*. Diagram ini menjelaskan bagaimana metode pemfaktoran *Fermat* bekerja nantinya pada sistem yang akan dibuat. Masukan yang dibutuhkan di tahapan ini adalah kunci publik ( $n$ ). Lalu dilanjutkan dengan menghitung nilai  $r = \sqrt{n}$  dan nilai  $r$  dibulatkan ke atas. Kemudian hitung nilai  $s = r^2 - n$ . Lalu dilakukan pengecekan apakah  $s$  adalah bilangan kuadrat sempurna. Jika ternyata  $s$  adalah bilangan kuadrat sempurna maka nilai  $p$  dan  $q$  bisa langsung dihitung. Namun jika tidak, nilai  $r$  ditambah dengan 1 lalu dilakukan perhitungan kembali nilai  $s = r^2 - n$  hingga didapatkan nilai  $s$  yang merupakan kuadrat sempurna. Setelah didapatkan nilai  $s$  yang merupakan bilangan kuadrat sempurna, maka dilakukan perhitungan terhadap nilai  $p = r - \sqrt{s}$  dan  $q = n / p$ .



**Gambar 3. 13 Diagram Alir Metode Pemfaktoran *Fermat***

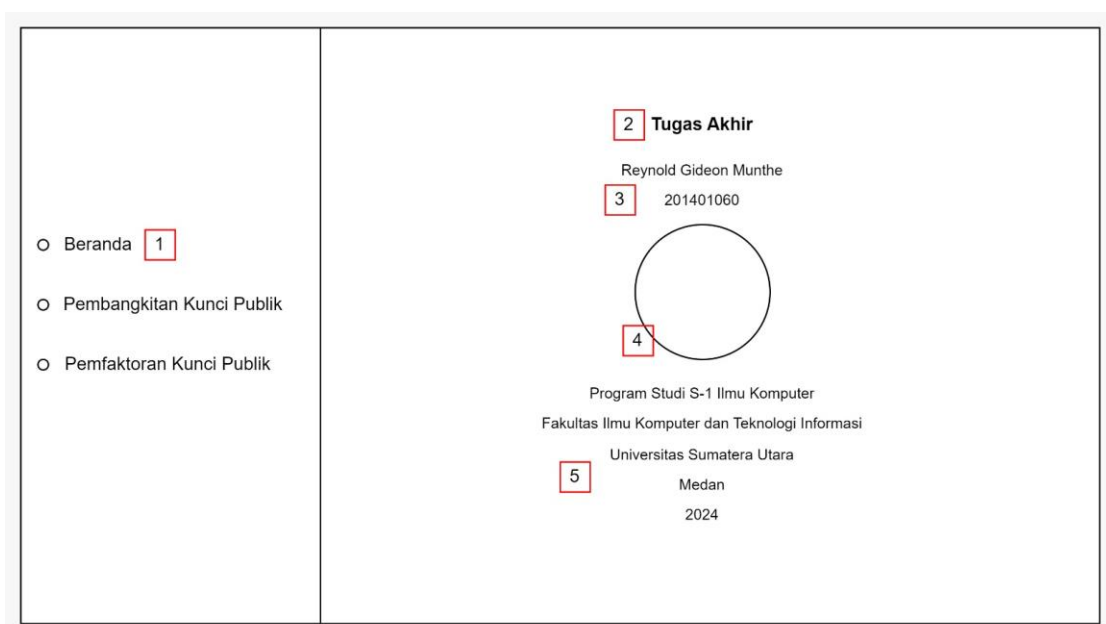
### 3.2.6. Perancangan *User Interface*

Perancangan *User Interface* (UI) adalah proses mendesain antarmuka yang efektif untuk perangkat lunak sehingga pengguna dapat berinteraksi dengan sistem dengan mudah dan efisien. UI merupakan elemen penting yang berfungsi sebagai jembatan antara pengguna dan komputer, memungkinkan interaksi melalui tampilan yang ditampilkan di layar.

Dalam merancang UI, tujuan utamanya adalah menciptakan antarmuka yang intuitif, responsif, dan memenuhi kebutuhan pengguna. Desain UI harus mempertimbangkan berbagai aspek seperti tata letak, warna, tipografi, dan navigasi untuk memastikan pengalaman pengguna yang optimal.

#### 3.2.6.1 Rancangan Menu Beranda

Saat user menggunakan sistem, user akan ditampilkan halaman beranda pertama kali. Pada halaman ini memuat tentang judul sistem, identitas penulis dan logo universitas penulis.



**Gambar 3. 14 Rancangan Menu Beranda**

Keterangan pada Gambar 3.14 ialah sebagai berikut :

1. Pilihan halaman yang ingin digunakan
2. Judul sistem
3. Identitas penulis
4. Logo universitas penulis
5. Program studi penulis

### 3.2.6.2 Rancangan Menu Pembangkitan Kunci Publik

Pada halaman pembangkitan kunci publik menampilkan proses yang dilakukan yaitu untuk melakukan pembangkitan bilangan  $p$  dan  $q$  yang akan dipakai untuk membangkitkan kunci publik.

**Gambar 3. 15 Rancangan Menu Pembangkitan Kunci**

Keterangan pada Gambar 3.15 ialah sebagai berikut :

1. Pilihan halaman yang ingin digunakan
2. Teks judul halaman
3. Teks petunjuk untuk memasukkan jumlah yang diinginkan untuk bilangan  $p$
4. Kotak untuk menginputkan besar digit  $p$

5. Teks petunjuk untuk memasukkan jumlah yang diinginkan untuk bilangan  $q$
6. Kotak untuk menginputkan besar digit  $q$
7. Tombol untuk membangkitkan bilangan  $p$ ,  $q$ , dan kunci publik  $n$
8. Teks untuk menampilkan bilangan  $p$  yang didapatkan
9. Teks untuk menampilkan bilangan  $q$  yang didapatkan
10. Teks untuk menampilkan bilangan  $n$  yang didapatkan

### 3.2.6.3 Rancangan Menu Pemfaktoran Kunci Publik

Pada halaman pemfaktoran kunci publik menampilkan proses untuk melakukan pemfaktoran kunci publik ( $n$ ) menjadi faktornya yaitu  $p$  dan  $q$  serta menampilkan lama waktu pemfaktoran dari kedua metode pemfaktoran yang digunakan.

**Gambar 3. 16 Rancangan Menu Pemfaktoran Kunci Publik**

Keterangan pada Gambar 3.16 ialah sebagai berikut :

1. Pilihan halaman yang ingin digunakan
2. Judul halaman yang sedang digunakan
3. Teks petunjuk untuk menginputkan kunci publik yang ingin difaktorkan

4. Kotak tempat memasukkan kunci publik yang akan difaktorkan
5. Tombol untuk memulai pemfaktoran
6. Hasil pemfaktoran yang didapat menggunakan metode *Kraitchik*
7. Nilai  $p$  dan  $q$  yang didapat oleh metode *Kraitchik*
8. Waktu pemfaktoran yang dibutuhkan oleh metode *Kraitchik*
9. Hasil pemfaktoran yang didapat menggunakan metode pemfaktoran *Fermat*
10. Nilai  $p$  dan  $q$  yang didapatkan oleh metode pemfaktoran *Fermat*
11. Waktu pemfaktoran yang dibutuhkan oleh metode pemfaktoran *Fermat*



## BAB IV

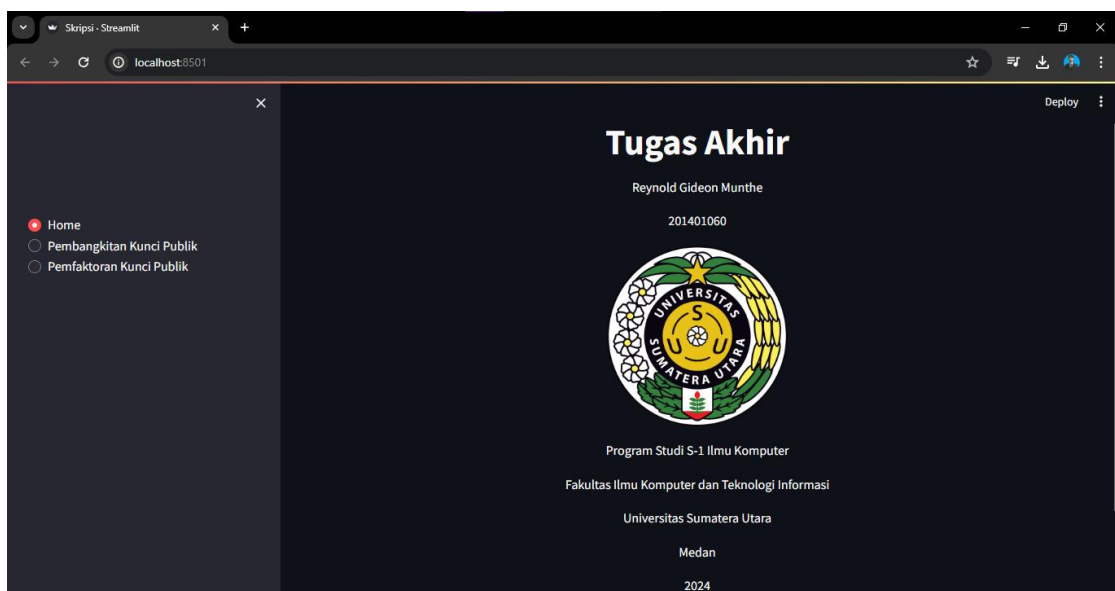
### IMPLEMENTASI DAN PENGUJIAN

#### 4.1. Implementasi Sistem

Pada penelitian ini, sistem dikembangkan menggunakan bahasa pemrograman *Python* sebagai back-end dan framework *streamlit* sebagai front-end dengan *Visual Studio Code* sebagai IDE. Pada sistem yang dikembangkan memiliki 4 laman, yakni laman Home, laman Pembangkitan Kunci, laman Enkripsi, dan laman Dekripsi.

##### 4.1.1. Laman Beranda

Laman Beranda ialah laman yang ditampilkan saat website dibuka. Laman Beranda ditampilkan pada Gambar 4.1.

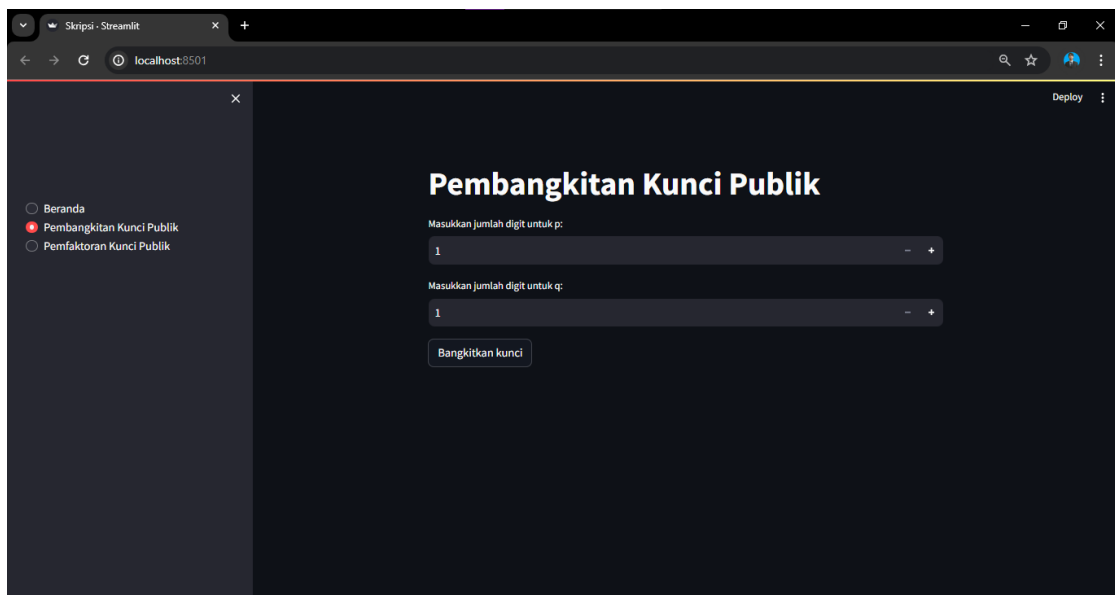


**Gambar 4. 1 Laman Beranda**

Pada Gambar 4.1 menampilkan laman beranda yang menampilkan informasi mengenai judul, nama pembuat sistem, logo universitas, dan program studi si pembuat sistem.

#### 4.1.2. Laman Pembangkitan Kunci Publik

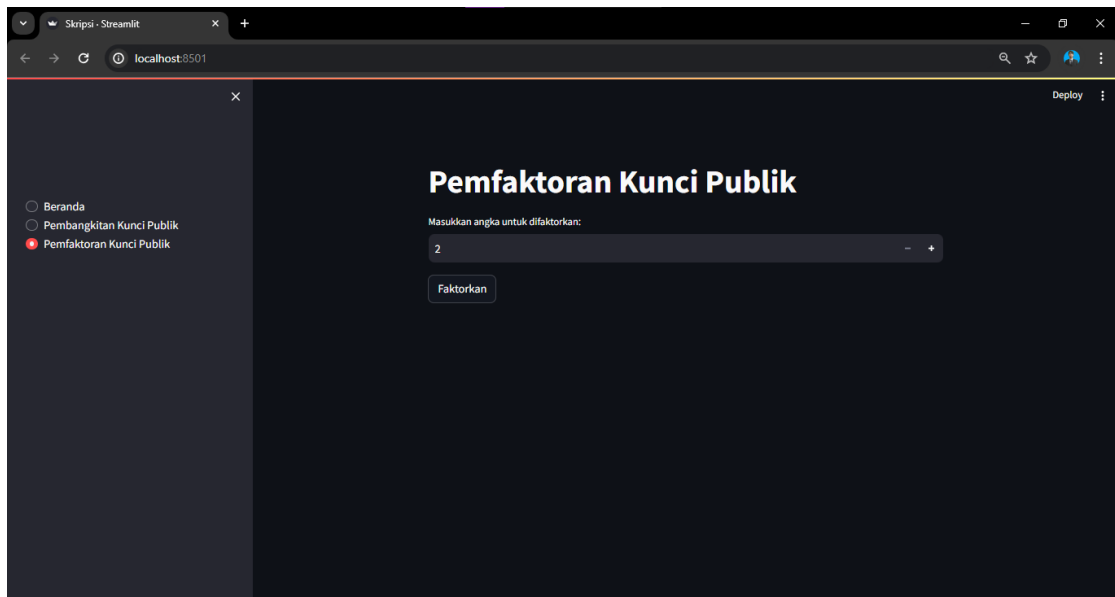
Pada Gambar 4.2 menampilkan laman Pembangkitan Kunci Publik. Laman ini digunakan untuk membangkitkan bilangan prima  $p$  dan  $q$  yang nantinya dipakai untuk mendapatkan kunci publik.

The image shows a web browser window with the address bar displaying 'localhost:8501'. The page title is 'Pembangkitan Kunci Publik'. On the left, there is a sidebar menu with three items: 'Beranda' (selected), 'Pembangkitan Kunci Publik', and 'Pemfaktoran Kunci Publik'. The main content area has the title 'Pembangkitan Kunci Publik' and two input fields. The first field is labeled 'Masukkan jumlah digit untuk p:' and contains the value '1'. The second field is labeled 'Masukkan jumlah digit untuk q:' and also contains the value '1'. Below these fields is a button labeled 'Bangkitkan kunci'. In the top right corner of the application, there is a 'Deploy' button.

Gambar 4. 2 Laman Pembangkitan Kunci Publik

#### 4.1.3. Laman Pemfaktoran Kunci Publik

Pada Gambar 4.3 menampilkan laman Pemfaktoran Kunci Publik. Laman ini digunakan untuk melakukan pemfaktoran terhadap kunci publik dengan metode *Kraitchik* dan metode pemfaktoran *Fermat*, dan mendapatkan hasil faktor dari kunci publik yaitu  $p$  dan  $q$ , serta mendapatkan lama waktu pemfaktoran kedua metode.



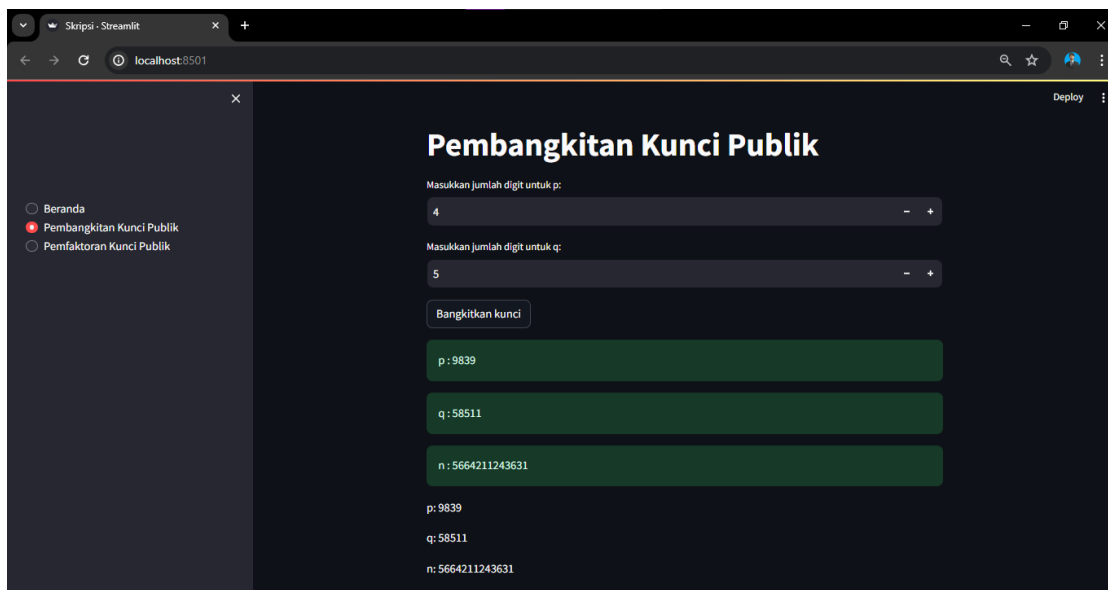
**Gambar 4. 3 Laman Pemfaktoran Kunci Publik**

## **4.2. Pengujian Sistem**

Pada fase ini dilaksanakan uji coba terhadap sistem yang dikembangkan untuk memastikan bahwa sistem mampu melaksanakan pembangkitan kunci publik dan pemfaktoran kunci publik dengan baik.

### **4.2.1. Pengujian Pembangkitan Kunci Publik**

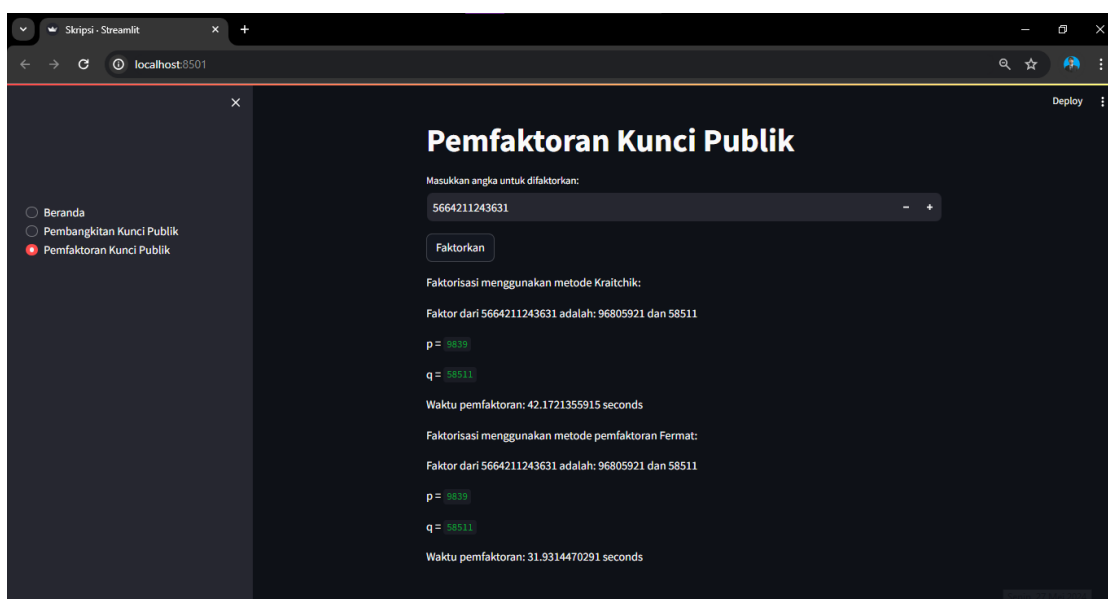
Pada fase ini akan dilaksanakan dengan menginput besar digit  $p$  dan  $q$ . Kemudian tombol "Bangkitkan Kunci" ditekan untuk membangkitkan bilangan prima  $p$  dan  $q$ . Setelah itu, akan diberikan nilai  $p$  dan  $q$  serta nilai  $n$  yang merupakan kunci publik. Kunci publik yang diberi bisa diambil untuk digunakan pada halaman Pemfaktoran Kunci Publik.



**Gambar 4. 4 Pengujian Pembangkitan Kunci Publik**

#### 4.2.2. Pengujian Pemfaktoran Kunci Publik

Pada tahap ini dilakukan faktorisasi terhadap kunci publik ( $n$ ) agar mendapatkan nilai  $p$  dan  $q$  serta lama waktu pemfaktoran dari dua metode pemfaktoran yang digunakan.



**Gambar 4. 5 Pengujian Pemfaktoran Kunci Publik**

Pada Gambar 4. menunjukkan hasil pemfaktoran dari kunci publik yang diberi. Dapat dilihat bahwa kedua metode pemfaktoran mendapatkan nilai  $p$  dan  $q$  yang sama, namun lama waktu pemfaktoran yang berbeda.

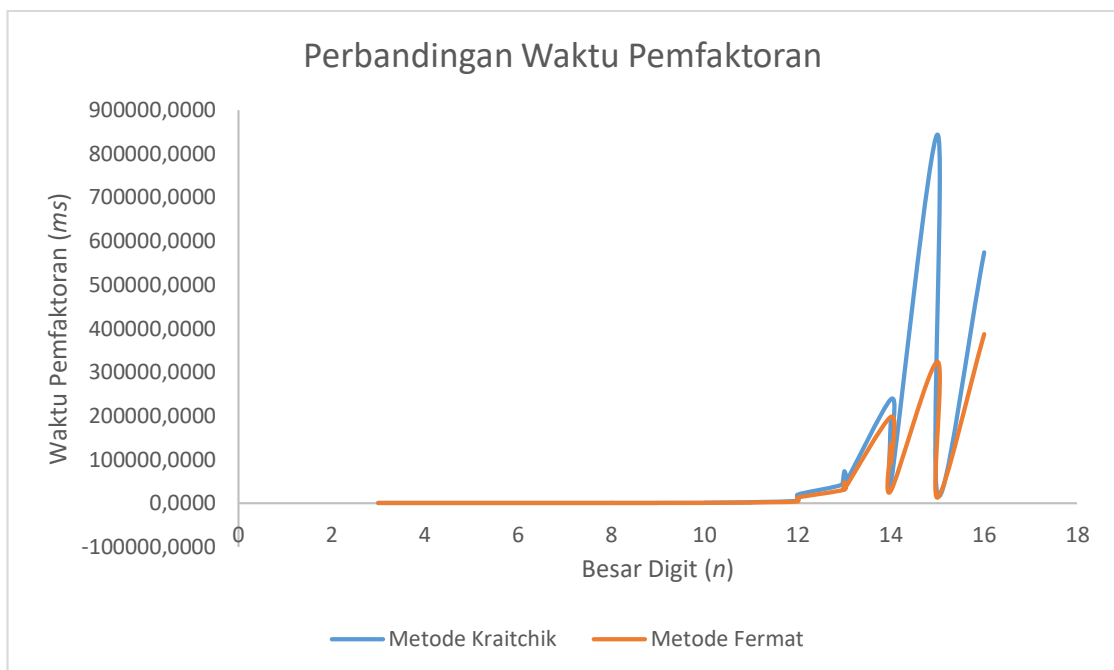
#### 4.2.2.1. Pengujian Panjang Digit Kunci Publik

Pada tahap ini, pengujian dilakukan dengan panjang digit kunci publik yang berbeda. Pada Tabel 4.1 menunjukkan waktu eksekusi untuk proses pemfaktoran kedua metode dengan panjang digit kunci publik yang semakin besar.

**Tabel 4. 1 Tabel Perbandingan Waktu Eksekusi**

Kunci Publik ( $n$ )	Digit $n$	$p$	$q$	Waktu ( $ms$ )	
				<i>Kraitichik</i>	<i>Fermat</i>
847	3	11	7	0,0000	0,0000
5687	4	11	47	0,0000	0,0000
20519	5	17	71	0,0000	0,0000
72557	5	37	53	0,0000	0,0000
99997	5	19	277	0,0000	0,0000
592777	6	37	433	0,0000	0,0000
216407	6	97	23	1,0254	0,0000
2688911	7	67	599	0,9978	0,0000
3660101	7	97	389	1,9939	1,3020
12439303	8	347	9151	25,6100	14,4591
42375457	8	271	577	14,9984	10,9310
47644967	8	179	1487	4,9844	2,9919
341946289	9	461	1609	37,1335	25,7306
945826201	9	797	1489	167,7601	94,6708
3271105637	10	877	4253	164,5913	97,7702
5078158301	10	1483	2309	526,6235	334,2111
25488161707	11	2417	4363	1401,4525	1018,5006
30147599093	11	2693	4157	1803,3836	1207,9606
124407822283	12	4729	5563	5320,1287	3369,0760
866393754547	12	8929	10867	19176,1839	12292,2034
2793215268191	13	13757	14759	43869,1285	30535,9135
6040665476507	13	17483	19763	72813,9148	47231,6093
5664211243631	13	9839	58511	42172,1356	31931,4470
14166465106337	14	23827	24953	237787,5102	197141,4707
14166465106337	14	23827	24953	136084,2376	90315,2032
25583862107137	14	28493	31513	189848,6931	133942,2872

47398356659209	14	9739	499729	46952,3108	28375,4687
131350193001857	15	45439	63617	843756,8371	322847,3351
161417668010327	15	7621	2779247	18703,8925	12227,6411
4704277282038533	16	34337	3989957	574060,4519	386224,4415



**Gambar 4. 6 Grafik Perbandingan Waktu Pemfaktoran**

Dari tabel dan gambar di atas, beberapa informasi penting diperoleh:

- Metode pemfaktoran *Fermat* terbukti lebih efisien dalam hal kecepatan dibandingkan metode *Kraitchik*.
- Tidak ada hubungan yang pasti antara nilai dan panjang kunci publik  $n$  dengan waktu pemfaktoran. Kadang-kadang, nilai  $n$  yang lebih besar tidak selalu menghasilkan waktu pemfaktoran yang lebih lambat.
- Terdapat kejadian di mana kunci dengan panjang yang lebih kecil memerlukan waktu pemfaktoran yang lebih lama dibanding kunci yang lebih besar, dan sebaliknya.

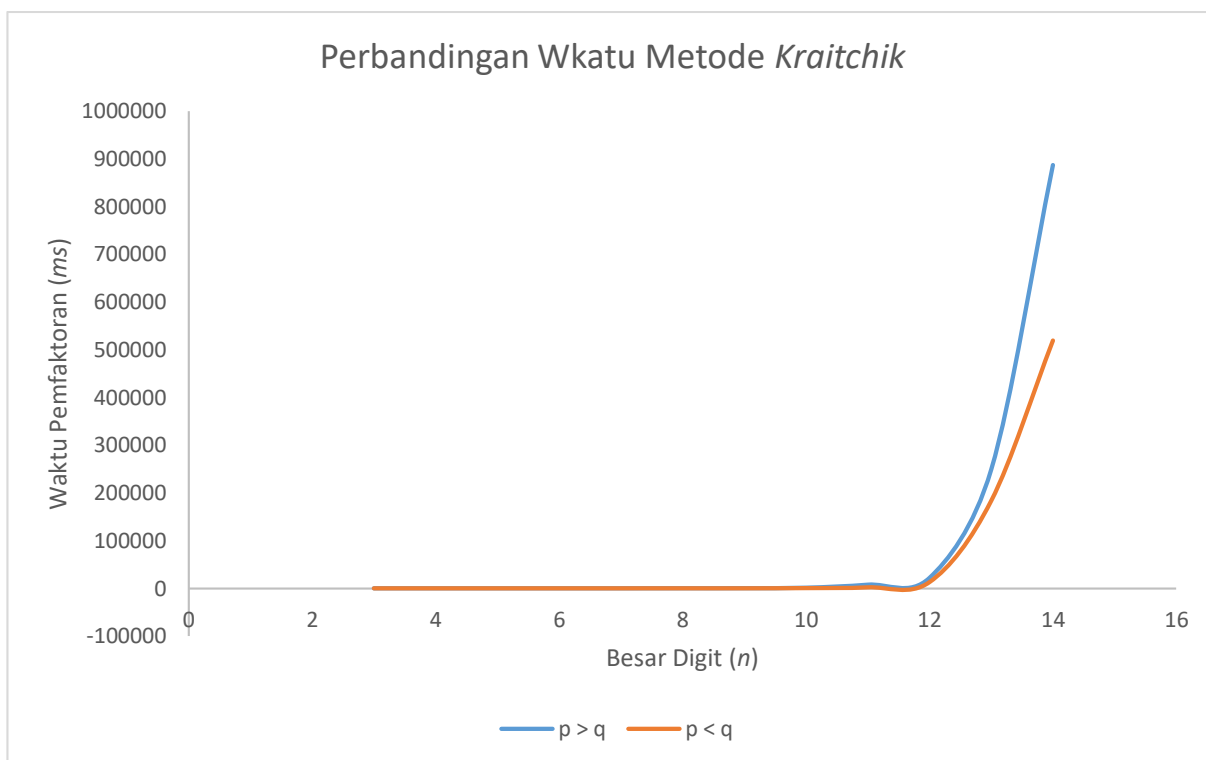
- d. Meningkatkan nilai kunci publik  $n$  dengan panjang digit yang sama bisa mendapatkan variasi waktu pemfaktoran yang signifikan, baik lebih lambat maupun lebih cepat.
- e. Pada digit kunci publik yang sama, ketika nilai  $p$  lebih kecil dari  $q$ , proses pemfaktoran cenderung lebih cepat daripada ketika  $p$  lebih besar dari  $q$ .

#### 4.2.2.2. Pengujian Perbandingan $p$ dan $q$ menggunakan metode *Kraitichik*

Pada tahap ini, pengujian dilakukan dengan melihat perbandingan  $p$  dan  $q$ . Pada Tabel 4. menunjukkan waktu eksekusi untuk proses pemfaktoran metode *Kraitichik* dengan nilai  $p$  lebih besar dibanding  $q$ , dan ketika nilai  $p$  lebih kecil dibanding  $q$ .

**Tabel 4. 2 Perbandingan Waktu Metode *Kraitichik***

digit	n	p > q		time	n	p < q		time
		p	q			p	q	
3	847	11	7	0	931	7	19	0
4	8993	23	17	0	8959	17	31	0
5	85583	61	23	0	81577	29	97	0
6	676757	113	53	1,106500626	695789	83	101	1,894951
7	9700937	293	113	11,36398315	9733261	167	349	11,65676
8	98196911	523	359	83,80270004	98840843	419	563	61,77664
9	975980323	1783	307	238,8315201	977204471	457	4679	64,41021
10	9936125323	3931	643	1223,375559	9986880551	1123	7919	510,6516
11	99349859897	6793	2153	7520,796537	99092674181	2131	21821	2051,182
12	989797608451	21313	2179	21740,21864	992651788873	5281	35593	13006,78
13	9979599617401	21799	21001	248716,9483	9931280252669	19429	26309	183458,2
14	99414878156171	60083	27539	887297,1728	99548320310431	41983	56479	519512,9



**Gambar 4. 7 Grafik Perbandingan Waktu Metode *Kraitichik***

#### 4.2.2.3. Pengujian Perbandingan $p$ dan $q$ Menggunakan Metode *Fermat*

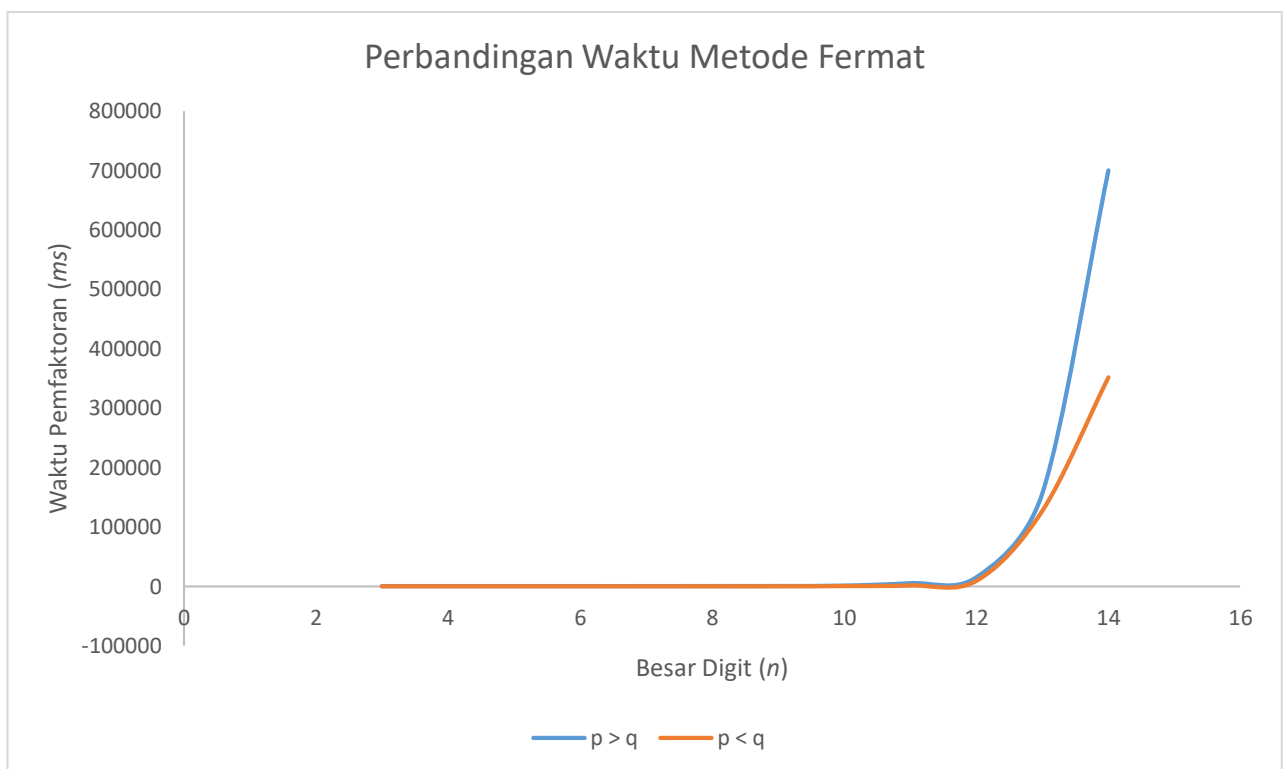
Pada tahap ini, pengujian dilakukan dengan melihat perbandingan  $p$  dan  $q$ . Pada Tabel 4. menunjukkan waktu eksekusi untuk proses pemfaktoran metode pemfaktoran *Fermat* dengan nilai  $p$  lebih besar dibanding  $q$ , dan ketika nilai  $p$  lebih kecil dibanding  $q$ .

**Tabel 4. 3 Perbandingan Waktu Metode Pemfaktoran *Fermat***

digit	n	$p > q$		time	n	$p < q$		time
		p	q			p	q	
3	847	11	7	0	931	7	19	0
4	8993	23	17	0	8959	17	31	0
5	85583	61	23	0	81577	29	97	0
6	676757	113	53	0	695789	83	101	0
7	9700937	293	113	5,153894424	9733261	167	349	0
8	98196911	523	359	62,62850761	98840843	419	563	50,24624



9	975980323	1783	307	167,6802635	977204471	457	4679	44,98053
10	9936125323	3931	643	816,6277409	9986880551	1123	7919	362,8471
11	99349859897	6793	2153	4911,0291	99092674181	2131	21821	1459,491
12	989797608451	21313	2179	14819,80777	992651788873	5281	35593	9067,977
13	9979599617401	21799	21001	156492,7604	9931280252669	19429	26309	127072,7
14	99414878156171	60083	27539	700103,2629	99548320310431	41983	56479	351398,3



**Gambar 4. 8 Grafik Perbandingan Waktu Metode *Fermat***

### 4.3. Kompleksitas Waktu

Pada fase ini dilaksanakan perhitungan kompleksitas waktu terhadap *pseudocode* metode *Kraitichik* dan metode pemfaktoran *Fermat*. Pada Gambar 4.9 dan Gambar 4.10 masing - masing menunjukkan *pseudocode* untuk metode *Kraitichik* dan metode pemfaktoran *Fermat*.

Pseudocode	Kompleksitas Waktu																																																						
<pre>def kraitichik_factorization(n):     x0 = int(math.sqrt(n))     k = 1     while True:         m = 1         while True:             x = x0 + m             y2 = x**2 - k * n              if y2 &gt;= 0:                 if is_perfect_square(y2):                     y = int(math.sqrt(y2))                     factor1 = x - y                     factor2 = x + y                     f1 = math.gcd(n, factor1)                     f2 = math.gcd(n, factor2)                  m +=1             k += 2</pre>	<table> <tr> <th>C</th> <th>#</th> <th>C#</th> </tr> <tr> <td>C<sub>1</sub></td> <td>1</td> <td>C<sub>1</sub></td> </tr> <tr> <td>C<sub>1</sub></td> <td>1</td> <td>C<sub>1</sub></td> </tr> <tr> <td>C<sub>2</sub></td> <td>n</td> <td>C<sub>2</sub> n</td> </tr> <tr> <td>C<sub>1</sub></td> <td>n</td> <td>C<sub>1</sub> n</td> </tr> <tr> <td>C<sub>2</sub></td> <td>n<sup>2</sup></td> <td>C<sub>2</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>3</sub></td> <td>n<sup>2</sup></td> <td>C<sub>3</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>3</sub></td> <td>n<sup>2</sup></td> <td>C<sub>3</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n<sup>2</sup></td> <td>C<sub>1</sub> n<sup>2</sup></td> </tr> <tr> <td>C<sub>1</sub></td> <td>n</td> <td>C<sub>1</sub> n</td> </tr> </table>	C	#	C#	C <sub>1</sub>	1	C <sub>1</sub>	C <sub>1</sub>	1	C <sub>1</sub>	C <sub>2</sub>	n	C <sub>2</sub> n	C <sub>1</sub>	n	C <sub>1</sub> n	C <sub>2</sub>	n <sup>2</sup>	C <sub>2</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>3</sub>	n <sup>2</sup>	C <sub>3</sub> n <sup>2</sup>	C <sub>3</sub>	n <sup>2</sup>	C <sub>3</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>	C <sub>1</sub>	n	C <sub>1</sub> n
C	#	C#																																																					
C <sub>1</sub>	1	C <sub>1</sub>																																																					
C <sub>1</sub>	1	C <sub>1</sub>																																																					
C <sub>2</sub>	n	C <sub>2</sub> n																																																					
C <sub>1</sub>	n	C <sub>1</sub> n																																																					
C <sub>2</sub>	n <sup>2</sup>	C <sub>2</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>3</sub>	n <sup>2</sup>	C <sub>3</sub> n <sup>2</sup>																																																					
C <sub>3</sub>	n <sup>2</sup>	C <sub>3</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n <sup>2</sup>	C <sub>1</sub> n <sup>2</sup>																																																					
C <sub>1</sub>	n	C <sub>1</sub> n																																																					
	$T(n) = (7C_1 + C_2 + 2C_3)n^2 + (2C_1 + C_2)n + 2C_1$ $= \Theta(n^2)$																																																						

**Gambar 4. 9 Pseudocode dan Kompleksitas Waktu metode *Kraitichik***

Pseudocode	Kompleksitas Waktu																								
<pre>def fermat_factorization(n):     r = math.ceil(math.sqrt(n))     s = r * r - n      while not is_perfect_square(s):         r += 1         s = r * r - n      t = math.isqrt(s)     p, q = r - t, r + t</pre>	<table><tr><th>C</th><th>#</th><th>C#</th></tr><tr><td>C<sub>1</sub></td><td>1</td><td>C<sub>1</sub></td></tr><tr><td>C<sub>1</sub></td><td>1</td><td>C<sub>1</sub></td></tr><tr><td>C<sub>2</sub></td><td>n</td><td>C<sub>2</sub> n</td></tr><tr><td>C<sub>1</sub></td><td>n</td><td>C<sub>1</sub> n</td></tr><tr><td>C<sub>1</sub></td><td>n</td><td>C<sub>1</sub> n</td></tr><tr><td>C<sub>1</sub></td><td>1</td><td>C<sub>1</sub></td></tr><tr><td>C<sub>1</sub></td><td>1</td><td>C<sub>1</sub></td></tr></table> $T(n) = (2 C_1 + C_2)n + 4 C_1$ $= \theta (n)$	C	#	C#	C <sub>1</sub>	1	C <sub>1</sub>	C <sub>1</sub>	1	C <sub>1</sub>	C <sub>2</sub>	n	C <sub>2</sub> n	C <sub>1</sub>	n	C <sub>1</sub> n	C <sub>1</sub>	n	C <sub>1</sub> n	C <sub>1</sub>	1	C <sub>1</sub>	C <sub>1</sub>	1	C <sub>1</sub>
C	#	C#																							
C <sub>1</sub>	1	C <sub>1</sub>																							
C <sub>1</sub>	1	C <sub>1</sub>																							
C <sub>2</sub>	n	C <sub>2</sub> n																							
C <sub>1</sub>	n	C <sub>1</sub> n																							
C <sub>1</sub>	n	C <sub>1</sub> n																							
C <sub>1</sub>	1	C <sub>1</sub>																							
C <sub>1</sub>	1	C <sub>1</sub>																							

**Gambar 4. 10 Pseudocode dan Kompleksitas Waktu metode pemfaktoran *Fermat***

Kompleksitas waktu untuk metode *Kraitichik* adalah  $\Theta(n^2)$  dan untuk metode pemfaktoran *Fermat* adalah  $\Theta(n)$ . Hasil ini merupakan hasil perhitungan secara kasar, bukan perhitungan yang lebih detail. Namun perhitungan ini didapatkan dengan mempertimbangkan kasus terburuk yang sangat jarang namun bisa terjadi. Dan berdasarkan hasil yang diperoleh dapat menjelaskan mengapa metode pemfaktoran *Fermat* mendapatkan hasil eksekusi yang lebih cepat dibandingkan metode *Kraitichik*.

## BAB V

### PENUTUP

#### 5.1. Kesimpulan

Sesuai dengan hasil analisis dan pengujian yang sudah dilakukan terhadap sistem perbandingan metode *Kraitchik* dan metode pemfaktoran *Fermat* dalam kriptanalisis kunci publik Algoritma *Schmidt-Samoa*, berikut adalah kesimpulan yang dapat ditarik:

1. Metode *Kraitchik* dan metode pemfaktoran *Fermat* mampu memfaktorkan kunci publik ( $n$ ) algoritma *Schmidt-Samoa* dengan baik dan menghasilkan faktor-faktor  $p$  dan  $q$  yang sesuai.
2. Waktu eksekusi metode pemfaktoran *Fermat* terbukti lebih cepat dibandingkan metode *Kraitchik* dalam memfaktorkan kunci publik Algoritma *Schmidt-Samoa*.
3. Tidak selalu terdapat korelasi langsung antara jumlah digit dan nilai kunci publik  $n$  dengan waktu pemfaktoran. Ada kasus di mana kunci dengan panjang yang lebih pendek memerlukan waktu faktorisasi lebih lambat dibanding kunci yang lebih besar, dan sebaliknya. Penyesuaian nilai  $n$  terhadap panjang digit yang sama bisa mendapatkan variasi waktu faktorisasi yang signifikan, baik lebih lambat maupun lebih cepat.
4. Waktu eksekusi kedua metode, cenderung lebih cepat ketika  $p$  lebih kecil dari  $q$  dibandingkan ketika  $p$  lebih besar dari  $q$ . Hal ini disebabkan oleh proses penghitungan nilai  $n$ , yang melibatkan kuadrat dari  $p$  sebelum dikalikan dengan  $q$ .

## 5.2. Saran

Sesuai dengan hasil temuan dalam penelitian ini, ada sejumlah saran yang bisa dijadikan sebagai bahan pertimbangan untuk penelitian berikutnya, antara lain:

1. Melakukan perbandingan lebih lanjut terhadap berbagai metode pemfaktoran kunci publik  $n$  yang telah dikembangkan, untuk mengevaluasi kekuatan, kelemahan, efektivitas, dan efisiensi masing-masing metode.
2. Melakukan perbandingan kriptanalisis kunci publik antara algoritma *Schmidt-Samoa* dengan algoritma kriptografi yang lain.
3. Menambah besar digit kunci publik ( $n$ ) yang difaktorkan hingga sesuai dengan rekomendasi yang diberikan oleh NIST yaitu sebesar 2048 bit.
4. Menetapkan ketentuan perangkat keras yang tepat dan sesuai untuk melaksanakan pengujian, guna memastikan keakuratan dan konsistensi hasil pengujian.

## DAFTAR PUSTAKA

- Abu Al-Haija, Q., M.Asad, M., & Marouf, I. (2018). A Systematic Expository Review of Schmidt-Samoa Cryptosystem. *International Journal of Mathematical Sciences and Computing*, 4(2), 12–21.
- Abu Al-haija, Q., Marouf, I., Asad, M. M., & Nasr, K. al. (2019). COST-EFFECTIVE FPGA IMPLEMENTATION OF PARALLEL SCHMIDT-SAMOA CRYPTOSYSTEM (SSC). *Journal of Theoretical and Applied Information Technology*, 15(15). [www.jatit.org](http://www.jatit.org)
- Al-Haija, Q. A., Marouf, I., Asad, M. M., & al Nasr, K. (2019). Implementing a lightweight schmidt-samoa cryptosystem (SSC) for sensory communications. *International Journal on Smart Sensing and Intelligent Systems*, 12(1), 1–9.
- Budiman, M. A., Rachmawati, D., & Utami, R. (2019, June). The cryptanalysis of the Rabin public key algorithm using the Fermat factorization method. In *Journal of Physics: Conference Series* (Vol. 1235, No. 1, p. 012084). IOP Publishing.
- Elvina, (2020). Implementasi dan Analisis Perbandingan *Schmidt-Samoa Cryptosystem* dengan RSA, ElGamal, dan *Rabin Cryptosystem*.
- Lydia, M. S., Andri Budiman, M., & Rachmawati, D. (2021). FACTORIZATION OF SMALL RPRIME RSA MODULUS USING FERMAT'S DIFFERENCE OF SQUARES AND KRAITCHIK'S ALGORITHMS IN PYTHON. *Journal of Theoretical and Applied Information Technology*, 15(11).
- Mohana Prabha, K., & Vidhya Saraswathi, P. (2020). Suppressed K-Anonymity Multi-Factor Authentication Based Schmidt-Samoa Cryptography for privacy preserved data access in cloud computing. *Computer Communications*, 158, 85–94.

- Muchlis, B. S., Budiman, M. A., & Rachmawati, D. (2017). Teknik Pemecahan Kunci Algoritma Rivest Shamir Adleman (RSA) dengan Metode Kraitchik. *Jurnal & Penelitian Teknik Informatika*, 2(2).
- Purba, S. (2019). Kriptanalisis Kunci Publik Algoritma Rabin Menggunakan Metode Kraitchik. *Jurnal Sains dan Teknologi*, 11(02), 205-213.
- Ramadhanus, A. (2015). Perbandingan Kriptanalisis RSA dan Schmidt Samoa menggunakan metode faktorisasi elliptic curve dan quadratic sieve.
- Ristanto, W., Raharjo, W. S., & Rachmat, A. (2013). IMPLEMENTASI ALGORITMASCHMIDT SAMOA PADA ENKRIPSI DEKRIPSI EMAIL BERBASIS ANDROID. *Jurnal Informatika*, 9(1), 25-32.