

***CRYPTANALYSIS ALGORITMA KUNCI PUBLIK INAM-KANWAL-  
ZAHID-ABID DENGAN METODE BRUTE-FORCE ATTACK***

**AUDREY MALIKA PUTRI BR SITEPU**

**201401131**



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

**MEDAN**

**2024**

***CRYPTANALYSIS ALGORITMA KUNCI PUBLIK INAM-KANWAL-  
ZAHID-ABID DENGAN METODE BRUTE-FORCE ATTACK***

**SKRIPSI**

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

**AUDREY MALIKA PUTRI BR SITEPU**

**201401131**



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

## PERSETUJUAN

Judul : *CRYPTANALYSIS ALGORITMA KUNCI PUBLIK*  
*INAM-KANWAL-ZAHID-ABID DENGAN METODE*  
*BRUTE-FORCE ATTACK*

Kategori : SKRIPSI

Nama : AUDREY MALIKA PUTRI BR SITEPU

Nomor Induk Mahasiswa : 201401131

Program Studi : SARJANA (SI) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI

UNIVERSITAS SUMATERA UTARA

Medan, 29 April 2024

Komisi Pembimbing :

Dosen Pembimbing I

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

Dosen Pembimbing II

Desilia Selvida S.Kom.,  
M.Kom  
NIP. 198912052020012001

Diketahui/ Disetujui oleh Program

Studi Sarjana Ilmu KomputerKetua,

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

**PERNYATAAN**  
***CRYPTANALYSIS PUBLIC KEY* MENGGUNAKAN ALGORITMA INAM-**  
**KANWAL-ZAHID-ABID**

**SKRIPSI**

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

Medan, 29 April 2024



Audrey Malika Putri br Sitepu

201401131

## **PENGHARGAAN**

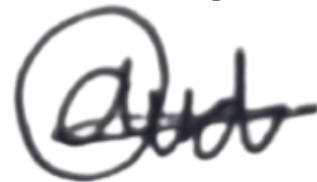
Puji dan Syukur kepada Tuhan Yang Maha Esa atas berkat Rahmat – Nya, penulis dapat menyelesaikan skripsi sebagai persyaratan tugas akhir untuk mendapat gelar Sarjana (S-1) Ilmu Komputer Universitas Sumatera Utara. Dalam menyelesaikan penyusunan laporan ini, penulis sadar sepenuhnya bahwa laporan ini tidak terlepas dari bimbingan serta dukungan dari banyak pihak yang terlibat dalam membantu penyelesaian skripsi penulis baik dari dukungan doa, tenaga, semangat, maupun pemikiran. Penulis mengucapkan terimakasih kepada:

1. Bapak Dr.Muryanto Amin, S. Sos., M. Si selaku Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lydia, M. Sc. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara dan dosen Pembimbing Akademik (PA) penulis yang membimbing penulis selama berkuliah di S1 Ilmu Komputer.
3. Ibu Dr. Amalia ST., M. T. selaku Ketua Program Studi S1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Dosen Pembimbing saya Dr. Mohammad Andri Budiman, S. T., M. Comp. Sc., M.E.M atas bimbingan dan arahan yang senantiasa mendukung penulis untuk menyelesaikan skripsi dengan tepat waktu selama bimbingan terjadi.
5. Dosen pembimbing saya ibu Desilia Selvida S.Kom., M.Kom atas bimbingan dan arahan dalam menyelesaikan tugas akhir penulis.
6. Seluruh dosen dan staff Ilmu Komputer yang telah membantu penulis selama melaksanakan perkuliahan.
7. Bapak Martan Sitepu dan Ibu Herawati Depari selaku orangtua penulis yang selalu mendukung penulis dalam setiap kegiatan yang penulis lakukan selama masa perkuliahan.
8. Getsemane Sitepu dan Evelyn Sitepu selaku saudara penulis yang selalu menghibur dan mendukung penulis selama penulisan skripsi.

9. Helen Ginting, Eunike Sembiring, Naomi Surbakti, Jesikapna, dan Monica Tarigan selaku teman penulis dari lingkungan gereja yang tidak lupa selalu mengajak penulis untuk beribadah.
10. Marsella Angelina dan Pretty Ohara selaku teman penulis dalam yang selalu membantu penulis.
11. Chindy Brigita, Hasnah Dewi, Rizka Aulia, dan Wina Saragih selaku teman-teman selama masa perkuliahan yang selalu menemani selama perkuliahan baik suka maupun duka.
12. Susi Telaumbanua selaku teman penulis atas bantuannya dalam menemani penulis selama bimbingan yang telah berlangsung.
13. *Seventeen*, Tulus, Sheila On 7, Hindia, dan *Stray Kids* yang menyediakan lagu dan konten yang menarik untuk menghibur penulis selama jenuh dalam pengerjaan skripsi.
14. Terutama kepada diri saya sendiri, terima kasih terus berjuang dalam setiap kejenuhan dan rintangan selama pengerjaan skripsi ini. Memang susah, tetapi kehidupan memiliki proses yang harus dijalani.

Kiranya Tuhan memberikan kelimpahan berkat dan kemurahan kepada semua pihak yang telah memberi dukungan doa, semangat, ilmu, dan bantuan kepada penulis dalam menyelesaikan skripsi ini. Semoga skripsi ini bermanfaat bagi penulis dan banyak pihak lainnya.

Medan, 29 April 2024



Audrey Malika Putri br Sitepu

201401131

## ABSTRAK

Sistem kriptografi kunci publik Inam-Kanwal-Zahid-Abid dikembangkan berdasarkan polinomial dengan menggunakan panjang kunci. Penelitian ini menguji serangan *brute-force* dengan menggunakan panjang kunci pendek. Metode yang digunakan penelitian untuk merancang dan menganalisis kriptanalisis terhadap kunci publik dengan mengimplementasikan algoritma *inam-kanwal-zahid-abid*. Hasil penelitian ini adalah penggunaan panjang kunci  $d_2$  mempengaruhi kecepatan dekripsi algoritma *Inam-Kanwal-Zahid-Abid*. Apabila  $d_2$  yang dihasilkan semakin besar maka, kecepatan *brute-force attack* akan semakin lama. Nilai  $N$  yang dihasilkan pembangkit kunci prima mempengaruhi panjang kunci enkripsi dan dekripsi algoritma *Inam-Kanwal-Zahid-Abid*.

**Kata Kunci:** *Public Key, Inam-Kanwal-Zahid-Abid, Brute-Force Attack, Enkripsi, Dekripsi, Kriptografi*

**ABSTRACT**

*The Inam-Kanwal-Zahid-Abid public key cryptography system was developed based on polynomials using key length. This research tested brute-force attacks using short key lengths. The method employed in the study aimed to design and analyze cryptanalysis against public keys by implementing the Inam-Kanwal-Zahid-Abid algorithm. The results of this research indicate that the use of sskey length  $d_2$  affects the decryption running time of the Inam-Kanwal-Zahid-Abid algorithm. As the value of  $d_2$  increases, the velocity of brute-force attacks proportionately extends. The value of  $N$  generated by the prime key generator affects the length of the encryption and decryption keys of the Inam-Kanwal-Zahid-Abid algorithm.*

**Kata Kunci:** *Public Key, Inam-Kanwal-Zahid-Abid, Brute-Force Attack, Encryption, Decryption, Cryptography*



## DAFTAR ISI

PERSETUJUAN .....	ii
PERNYATAAN.....	iii
PENGHARGAAN .....	iv
ABSTRAK.....	v
<i>ABSTRACT</i> .....	vii
DAFTAR ISI.....	viii
DAFTAR TABEL.....	x
DAFTAR GAMBAR .....	xi
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Penelitian.....	3
1.4 Tujuan Penelitian .....	3
1.5 Manfaat Penelitian .....	3
1.6 Metode Penelitian .....	3
1.7 Sistematika Penulisan .....	4
BAB II LANDASAN TEORI.....	6
2.1 Kriptografi .....	6
2.2 Kunci Publik .....	7
2.3 <i>Conjugacy Search Problem</i> .....	8
2.4 Bilangan Prima .....	9
2.5 <i>Fermat's Little Theorem</i> .....	9
2.6 Matriks.....	10
2.7 <i>Inverse Modulo</i> .....	11
2.8 Algoritma Inam-Kanwal-Zahid-Abid .....	11
2.9 <i>Cryptanalysis</i> .....	16
2.10 <i>Brute-Force Attack</i> .....	17
2.11 Penelitian Relevan .....	17
BAB III ANALISIS DAN PERANCANGAN SISTEM .....	20
3.1 Analisis Sistem .....	20
3.2 <i>Flowchart</i> .....	22

3.2.1	<i>Flowchart</i> Pembangkit Kunci <i>Fermat's Little Theorem</i> .....	22
3.2.2	<i>Flowchart</i> Key Generation.....	23
3.2.3	<i>Flowchart</i> Proses Enkripsi .....	25
3.2.4	<i>Flowchart</i> Proses Dekripsi.....	26
3.2.5	<i>Flowchart</i> Proses <i>Brute-Force Attack</i> .....	26
3.3	Diagram Umum Sistem.....	27
3.4	<i>Use Case</i> Diagram .....	28
3.5	<i>Activity</i> Diagram .....	29
BAB IV IMPLEMENTASI DAN PENGUJIAN.....		30
4.1	Implementasi.....	30
4.1.1	Proses <i>Key Generation</i> .....	30
4.1.2	Hasil Program .....	30
4.1.3	Inputan .....	31
4.2	<i>Analysis running time</i> .....	32
4.2.1	<i>Key Generation</i> .....	32
4.2.2	Fungsi Enkripsi.....	33
4.2.3	Fungsi Dekripsi.....	34
4.2.4	Fungsi <i>Brute-Force Attack</i> .....	34
4.3	Pengujian .....	35
BAB V KESIMPULAN DAN SARAN.....		45
5.1	Kesimpulan .....	45
5.2	Saran .....	45
DAFTAR PUSTAKA .....		46
LAMPIRAN.....		A-1

## DAFTAR TABEL

<b>Tabel 4.1</b> Pembangkit Kunci Bilangan Prima .....	32
<b>Tabel 4.2</b> <i>Generate</i> Bilangan Prima .....	32
<b>Tabel 4.3</b> Enkripsi <i>Inam-Kanwal-Zahid-Abid</i> .....	33
<b>Tabel 4.4</b> Dekripsi <i>Inam-Kanwal-Zahid-Abid</i> .....	34
<b>Tabel 4.5</b> Fungsi <i>Brute-Force Attack</i> .....	34
<b>Tabel 4.6</b> <i>Brute-Force Attack</i> dengan p dan q (1,100) .....	35
<b>Tabel 4.7</b> <i>Brute-Force Attack</i> dengan karakter yang berbeda .....	37
<b>Tabel 4.8</b> <i>Running Time</i> Enkripsi .....	38
<b>Tabel 4.9</b> <i>Running Time</i> Enkripsi terhadap M berbeda dan N tetap .....	39
<b>Tabel 4.10</b> Dekripsi <i>Inam-Kanwal-Zahid-Abid</i> .....	40
<b>Tabel 4.11</b> <i>Running Time</i> Dekripsi terhadap M berbeda .....	41
<b>Tabel 4.12</b> <i>Running Time</i> K dengan M yang berbeda .....	42
<b>Tabel 4.13</b> Nilai N dalam mencari mencari K .....	43

## DAFTAR GAMBAR

<b>Gambar 2.1</b> Kriptografi Asimetris.....	6
<b>Gambar 2.2</b> Kriptografi Simetris .....	7
<b>Gambar 2.3</b> Enkripsi-Dekripsi <i>Public Key</i> .....	8
<b>Gambar 2.4</b> <i>Brute-Force Attack</i> .....	17
<b>Gambar 3.1</b> Diagram Ishiskawa .....	20
<b>Gambar 3.2</b> <i>Flowchart</i> Pembangkit Kunci.....	22
<b>Gambar 3.3</b> <i>Flowchart</i> Pembangkit <i>Random Prima</i> .....	23
<b>Gambar 3.4</b> <i>Flowchart Key Generation</i> .....	24
<b>Gambar 3.5</b> <i>Flowchart</i> Enkripsi .....	25
<b>Gambar 3.6</b> <i>Flowchart</i> Dekripsi.....	26
<b>Gambar 3.7</b> <i>Flowchart Brute-Force Attack</i> .....	27
<b>Gambar 3.8</b> Diagram Umum Sistem .....	28
<b>Gambar 3.9</b> <i>Use Case Diagram</i> .....	28
<b>Gambar 3.10</b> <i>Activity Diagram</i> .....	29
<b>Gambar 4.1</b> Nilai <i>key generation</i> .....	30
<b>Gambar 4.2</b> Hasil Program .....	30
<b>Gambar 4.3</b> Input <i>fungsi</i> program .....	31
<b>Gambar 4.4</b> Grafik perbandingan d1, d2 dengan d2 <i>brute-force attack</i> .....	36
<b>Gambar 4.5</b> Grafik perbandingan d2 <i>brute-force attack</i> dengan <i>running time</i> .....	37
<b>Gambar 4.6</b> <i>Running time brute-force attack</i> .....	38
<b>Gambar 4.7</b> Nilai d1, d2, dan N.....	39
<b>Gambar 4.8</b> <i>Running time</i> enkripsi dengan nilai M berbeda .....	40
<b>Gambar 4.9</b> Grafik nilai dekripsi d1,d2, dan N .....	41
<b>Gambar 4.10</b> Perbandingan nilai <i>running time</i> dekripsi.....	42
<b>Gambar 4.11</b> Perbandingan <i>running time</i> K terhadap nilai M berbeda.....	43
<b>Gambar 4.12</b> Nilai N terhadap K.....	44

## BAB I PENDAHULUAN

### 1.1 Latar Belakang

Kriptografi digunakan untuk melindungi privasi pengguna, informasi transaksi, dan memastikan konsistensi data (Zhai, *et al.*, 2019). *Public Key* dikenal dengan kriptografi asimetris yang digunakan untuk mengamankan data dengan melibatkan *public key* dan *private key* (Abdulla & Ehsan Rana, 2021). *Public Key Encryption* (PKE) memungkinkan orang berkomunikasi dengan aman dan berbagi data sensitif kepada orang lain melalui saluran publik (Jarecki, 2020). Dekripsi data hanya dilakukan dengan menggunakan *private key* yang sesuai dari *public key* yang digunakan dalam proses enkripsi dengan menggunakan algoritma yang sama. Siapapun dapat mengenkripsi pesan di bawah kunci publik dan setiap pengguna yang sah dapat mendekripsi *ciphertext* yang dihasilkan menggunakan kunci rahasia masing-masing. Pemicu pertama mengenai kriptografi kunci publik (PKC) dilakukan oleh Diffie dan Hellman. Setelah itu, banyak algoritma kunci publik yang diusulkan seperti RSA, ElGamal *Cryptosystem*, *Elliptic Curve Cryptography* (ECC), dan masalah *Discrete Logarithm Problem* (DLP) dan skema ini dianggap aman. Algoritma Inam-Kanwal-Zahid-Abid merupakan algoritma kunci publik baru yang mengembangkan kunci publik berdasarkan *polynomial* pada cincin *non - commutative* dengan menggunakan kriptografi matriks.

Kriptanalisis tidak memiliki cara untuk mengidentifikasi *polynomial* karena memiliki komputasi yang tak terbatas. Keunggulan komputasi algoritma ini berada pada penggunaan panjang kunci terpendek yang mengurangi semua perhitungan dengan sistem yang aman. Jika ukuran matriks diperbesar, kompleksitas operasi matriks juga meningkat (Inam, *et al.*, 2020). Tidak peduli bagaimana musuh mengatur ulang persamaan, masalah memiliki produk dari dua matriks yang tidak diketahui tidak dapat dihindari yang mengarah ke sistem persamaan *non linier* yang besar dalam sejumlah besar entri yang tidak diketahui (Inam, *et al.*, 2021).

*Brute force attack* adalah serangan *cryptanalysis* yang memecahkan *cryptosystem* dengan menguji semua kunci yang mungkin. Serangan ini memperoleh informasi pribadi pengguna, informasi ini dapat berupa PIN, *password*, dan nama pengguna (Verma, *et al.*, 2022). *Brute force attack* mengambil waktu untuk mencoba semua *private key* yang kemungkinan terjadi. Serangan ini berbasis skema *padding* membutuhkan sejumlah besar memori untuk membangun daftar tabel dari semua kemungkinan *plaintext*, nilai pesan, dan *ciphertext* yang sesuai (Chu Jian & Chai, 2019). Apabila semakin panjang kunci, semakin besar waktu yang dibutuhkan *brute force* untuk menyerang dan kunci semakin kuat. Kunci kurang dari 64 bit tidak terlalu aman terhadap serangan *brute-force attack*, informasi rahasia dapat dicadangkan dengan kunci 128bit (Ribouh, *et al.*, 2020). Algoritma Inam-Kanwal-Zahid-Abid merupakan algoritma yang dikembangkan *cryptosystem public key* berdasarkan *polynomial* pada cincin *non-commutative* dengan penggunaan panjang kunci terpendek untuk mengurangi perhitungan melalui sistem yang aman. Keamanan kriptografi bergantung pada berbagai faktor seperti nilai  $N$ , polinomial, urutan matriks, dan *matrix discrete logarithm problem* (MDLP). Jika ukuran matriks diperbesar, maka kompleksitas operasi matriks meningkat yang setara dengan ukuran bit yang jauh lebih kecil (Inam, *et al.*, 2020). Oleh karena itu, penelitian ini akan membuktikan apakah *cryptanalysis* dapat mengembalikan *plaintext* dengan mengetahui *public key* menggunakan algoritma *Inam-Kanwal-Zahid-Abid*.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah yang menjadi fokus utama dalam penelitian ini adalah menguji apakah *ciphertext* yang dihasilkan algoritma Inam-Kanwal-Zahid-Abid dapat didekripsi dengan mudah melalui kunci publik yang diketahui oleh serangan *brute-force*

### 1.3 Batasan Penelitian

Dalam melakukan penelitian, peneliti membatasi masalah yang akan diteliti. Batasan masalah dalam penelitian ini adalah sebagai berikut:

- 1.3.1. Proses *Key Generation*, Enkripsi, dan Dekripsi menggunakan matriks cincin  $M_n(\mathbb{Z}_N)$ .
- 1.3.2. Program dirancang dengan menggunakan bahasa *Python*
- 1.3.3. Matriks yang digunakan berordo  $2 \times 2$

### 1.4 Tujuan Penelitian

Penelitian ini bertujuan untuk membangun program kriptografi kunci publik yang efisien, efektif, dan aman untuk digunakan terhadap serangan yang terjadi, terutama *brute-force attack*.

### 1.5 Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini adalah mengamankan data dengan menggunakan algoritma Inam-Kanwal-Zahid-Abid dalam meminimalisasi serangan *brute-force*. Algoritma kriptografi dapat membantu mengamankan data dengan penggunaan panjang kunci terpendek untuk mengurangi perhitungan melalui sistem yang aman.

### 1.6 Metode Penelitian

#### 1.6.1. Studi Literatur

Pada tahap penelitian ini, peneliti memulai penelitian dengan mengumpulkan sumber dengan menggunakan metode studi pustaka melalui peninjauan buku, jurnal, *e-book*, serta artikel ilmiah yang berhubungan dengan kriptografi, enkripsi, dekripsi, *cryptosystem*, dan algoritma Inam-Kanwal-Zahid-Abid

### 1.6.2. Analisa dan Perancangan

Pada tahap penelitian ini, peneliti melakukan analisis terhadap algoritma Inam-Kanwal-Zahid-Abid sebagai algoritma kunci publik dalam membantu mendekripsikan *ciphertext* terhadap serangan kriptografi.

### 1.6.3. Implementasi

Pembuatan sebuah sistem dengan menggunakan bahasa pemrograman *python* sesuai dengan diagram alir (*Flowchart*) yang telah dirancang dan menganalisis Kriptanalisis terhadap kunci publik dengan mengimplementasikan algoritma Inam-Kanwal-Zahid-Abid serta mempercepat proses komputasi dengan menggunakan panjang kunci terpendek yang digunakan.

### 1.6.4. Pengujian

Pada tahap ini, peneliti melakukan uji coba apakah algoritma yang digunakan efektif dan efisien dalam komputasi data dengan menggunakan Algoritma Inam-Kanwal-Zahid-Abid.

### 1.6.5. Dokumentasi

Pada tahap ini akan dilakukan pembuatan laporan hasil dari perancangan dan pengujian dalam bentuk skripsi.

## 1.7 Sistematika Penulisan

Sistematika penulisan skripsi penulis terdiri dari lima bab, antara lain:

### **BAB I PENDAHULUAN**

Pada bab ini berkaitan dengan latar belakang dari rumusan masalah yang akan diteliti yang berjudul *Cryptanalysis Public Key* menggunakan Algoritma Inam-Kanwal-Zahid-Abid dengan batasan masalah beserta manfaat penelitian, metode penelitian, dan sistem penulisan yang dijabarkan pada bab ini.



**BAB II        LANDASAN TEORI**

Pada bab ini berisi teoritis yang berkaitan dengan penelitian *Cryptanalysis Public Key* dan Algoritma Inam-Kanwal-Zahid-Abid.

**BAB III       ANALISIS PERANCANGAN**

Pada bab ini akan dilakukan analisis yang berkaitan dengan penelitian.

**BAB IV       IMPLEMENTASI DAN PENGUJIAN SISTEM**

Pada bab ini akan dilakukan implementasi dan pengujian sistem berdasarkan analisis yang berkaitan dengan penelitian.

**BAB III       KESIMPULAN DAN SARAN**

Pada bab ini akan memuat kesimpulan dan saran berdasarkan penelitian yang diharapkan untuk mendukung penelitian selanjutnya.

## BAB II

### LANDASAN TEORI

#### 2.1 Kriptografi

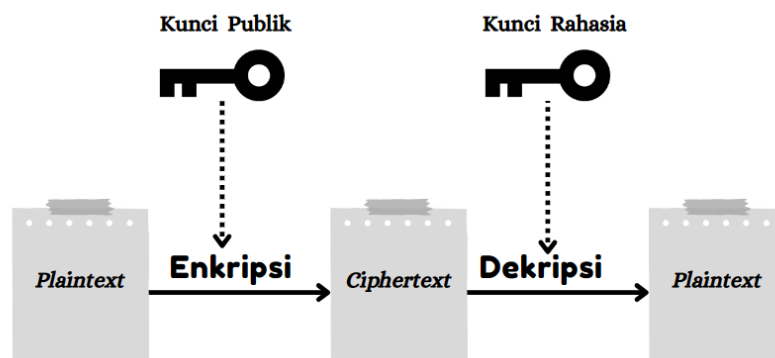
Kriptografi melibatkan matematika dan komputer untuk menyandikan data dalam format tertentu, dimana penguraian kode hanya menggunakan kunci yang sama yang digunakan dalam proses pengkodean (Abdulla & Rama, 2020). Kriptografi bertujuan mengirimkan informasi antar peserta sedemikian rupa sehingga ancaman dari musuh dapat dicegah. Kriptografi adalah metode mengubah pesan rahasia sehingga hanya *sender* dan *recipient* yang berwenang yang telah diberikan kunci rahasia untuk di dekripsi.

Berdasarkan kunci kriptografi ada dua bagian:

##### 1. Kriptografi asimetris

Kriptografi asimetris atau kunci publik memiliki kunci berbeda untuk setiap proses enkripsi dan dekripsi. Kunci enkripsi dibagikan secara publik dan kunci dekripsi disimpan sebagai *private key* (Budiman, *et al.*, 2020).

Contoh: *Rivest–Shamir–Adleman (RSA)*, *Diffie–Hellman*, *Digital Secure Algorithm (DSA)*, *XTR*, *Elliptic Curve Cryptography (ECC)*, dan *Elgamal Encryption System (ESS)*

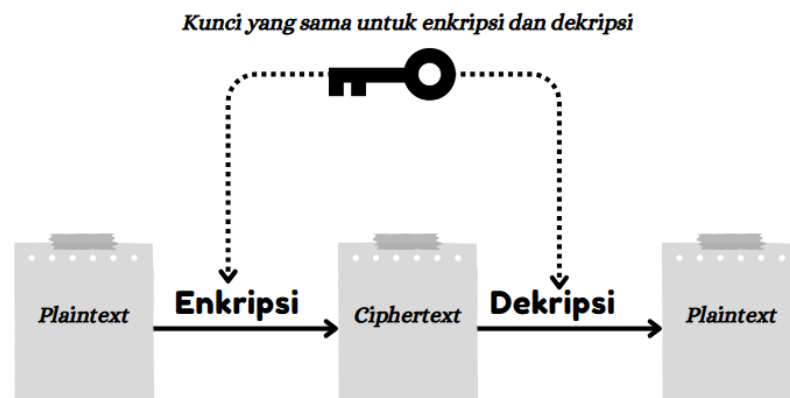


**Gambar 2.1** Kriptografi Asimetris

## 2. Kriptografi simetris

Kriptografi simetris merupakan algoritma yang menggunakan kunci yang sama untuk proses enkripsi dan dekripsi (Budiman, *et al.*, 2020)

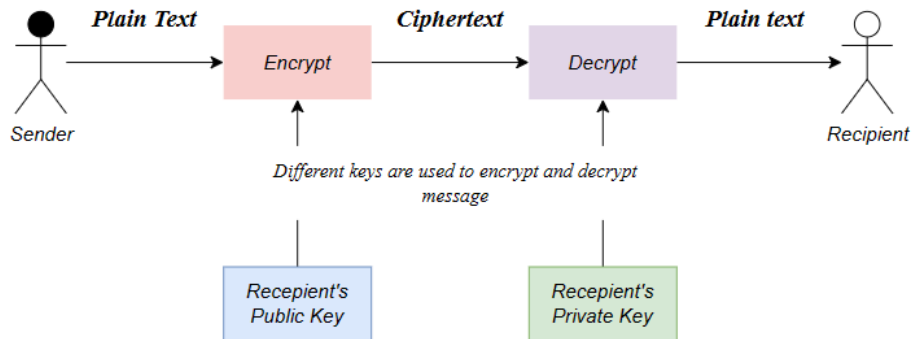
Contoh: DES (*Data Encryption Standard*), *blowfish*, *twofish*, AES (*Advanced Encryption Standard*)



**Gambar 2.2** Kriptografi Simetris

### 2.2 Kunci Publik

Teori kriptografi dimulai pada awal 1970, tetapi konsep kunci publik dimulai sejak 1976 sejak publikasi dari Diffie dan Hellman. Konsep kunci publik menjadi populer sejak dikeluarkannya algoritma RSA pada tahun 1978 yang berjudul "*A Method for Obtaining Digital Signatures and Public Key Cryptosystems*". Algoritma RSA berasal dari Ronald Rivest, Adi Shamir, dan Leonard Adleman. Sampai sekarang RSA paling banyak digunakan sebagai algoritma kriptografi kunci publik. Kriptografi kunci publik digunakan untuk menutup kunci kriptografi simetris dalam enkripsi seperti tanda tangan digital yang digunakan untuk memastikan integritas autentikasi pengguna. *Public key* digunakan untuk keamanan data *public* dan infrastruktur *private cloud* termasuk penyimpanan data secara privat.



**Gambar 2.3** Enkripsi-Dekripsi *Public Key*

Proses enkripsi dan dekripsi kunci publik, antara lain:

- Penerima akan membuat *public key* dan *private key* untuk digunakan dalam proses enkripsi.
- *Private key* akan dibuat dari *public key* dan kedua kunci akan berhubungan secara perhitungan matematis, tetapi tidak akan identik.
- Penerima akan mengirim *public key* penerima kepada pengirim, untuk proses enkripsi data.
- *Plaintext* akan dikonversi menjadi *ciphertext* menggunakan *public key* oleh penerima dan algoritma enkripsi. *Ciphertext* akan dikirimkan ke penerima
- Penerima akan melakukan dekripsi terhadap *ciphertext* menggunakan *private key* milik penerima dan algoritma yang digunakan dalam proses enkripsi

### 2.3 Conjugacy Search Problem

*Conjugacy Search Problem* dalam group  $G$  merupakan permasalahan dalam mendapatkan kembali nilai  $x \in G$  dari nilai yang diberikan  $g \in G$  dan  $h = x^{-1}gx$ . Permasalahan ini terjadi dalam pertukaran *public key* (Shpilrain & Ushakov, 2006).

## 2.4 Bilangan Prima

Bilangan prima adalah bilangan bulat  $p$  ( $p > 1$ ) yang hanya memiliki pembagi positif, yaitu 1 dan dirinya sendiri. Seluruh bilangan prima merupakan bilangan ganjil, tetapi tidak semua bilangan ganjil termasuk prima. Angka 2 termasuk dengan bilangan prima, bilangan selain prima disebut dengan bilangan komposit.

Contoh:

Bilangan prima:

2, 3, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47

Bilangan komposit:

20 ( $4 \times 5$ ), 4 ( $2 \times 2$ ), 6 ( $2 \times 3$ ), 14 ( $2 \times 7$ ), 9 ( $3 \times 3$ ), 143 ( $11 \times 13$ )

## 2.5 Fermat's Little Theorem

Jika  $p$  adalah bilangan prima dan  $a$  tidak habis dibagi  $p$ , maka:

$$a^{p-1} \equiv 1 \pmod{p} \text{ atau } a^p \equiv a \pmod{p}$$

Contoh:

Tentukan  $2^{2024} \pmod{13}$

$$2024 = 168 \times 12 + 8$$

$$2^{2024} \equiv 2^{168 \times 12 + 8} \pmod{13}$$

$$2^{2024} \equiv (2^{12})^{168} 2^8 \pmod{13}$$

$$2^{2024} \equiv (2^{12})^{168} 2^8 \pmod{13}$$

$$2^{2024} \equiv (1)^{168} 2^8 \pmod{13}$$

$$2^{2024} \equiv 2^8 \pmod{13}$$

$$2^{2024} \equiv 256 \pmod{13}$$

$$2^{2024} \equiv 9 \pmod{13}$$

## 2.6 Matriks

Matriks  $A$  berukuran  $m \times n$  memiliki nomor  $a_{ij}$  yang disebut dengan elemen matriks  $A$ .

Jika  $m=n$ , maka  $A$  adalah matriks persegi.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Matriks  $A$  berukuran  $n \times n$  dapat dibalik tepat jika matriks tersebut ada matriks lain  $A^{-1}$

sehingga  $AA^{-1} = A^{-1}A = I$ .

$$I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

Misal matriks  $2 \times 2$  adalah

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Maka nilai *inverse matrix* adalah

$$A^{-1} = \frac{adj(A)}{\det(A)}$$

$$adj(A) = \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$\det(A) = ad - bc$$

Contoh:

Jika  $A$  adalah matriks

$$A = \begin{bmatrix} 2 & 1 \\ 5 & 3 \end{bmatrix}$$

$A^{-1}$  didapat dari :

$$\det(A) = 2 \times 3 - 5 \times 1$$

$$\det(A) = 6 - 5 = 1$$

$$A^{-1} = \frac{\text{adj}(A)}{\det(A)}$$

$$A^{-1} = \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$A^{-1} = \begin{bmatrix} 3 & -1 \\ -5 & 2 \end{bmatrix}$$

Maka *invers* dari A adalah

$$A^{-1} = \begin{bmatrix} 3 & -1 \\ -5 & 2 \end{bmatrix}$$

## 2.7 Inverse Modulo

Diberikan dua bilangan bulat  $a$  dan  $m$ , carilah *inverse* perkalian modulo dari  $A$  pada Modular  $M$ . Apabila  $m$  dan  $a$  merupakan relatif prima dan  $m > 1$ . *Inverse* perkalian modulo adalah bilangan bulat sehingga

$$a \cdot a^{-1} \equiv 1 \pmod{m}.$$

Dimana  $x$  bilangan bulat modulo  $m$  dan tidak boleh sama dengan 0.  $x$  harus berada dalam  $\{1, 2, 3, \dots, m-1\}$ . *Inverse* perkalian dari “ $a$  modulo  $m$ ” ada jika dan hanya jika  $a$  dan  $m$  relatif prima ( $\text{PBB}(a, m) = 1$ ).

## 2.8 Algoritma Inam-Kanwal-Zahid-Abid

Pada tahun 2016, S.Inam dan R. Ali mengembangkan sistem krypto dimana struktur kerja yang mendasarinya adalah *Conjugacy Search Problem* (CSP) dan skema lain yang dirumuskan oleh S. Kanwal dan R. Ali dengan menggunakan kelompok *non-commutative*. Algoritma Inam-Kanwal-Zahid-Abid merupakan algoritma yang dikembangkan *cryptosystem public key* berdasarkan *polynomial* pada cincin *non-commutative* dengan penggunaan panjang kunci terpendek untuk mengurangi

perhitungan melalui sistem yang aman. Skenario protokol kriptografi mengenai otentikasi, pertukaran kunci, enkripsi, dan dekripsi algoritma.

### ***Key Generation***

Algoritma ini mempertimbangkan secara acak dua bilangan prima besar  $p$  dan  $q$ , dimana  $N = pq$ , sebuah matriks cincin  $M_n(Z_N)$ . Dimana  $f(x) = a_0 + a_1x^1 + a_2x^2 + \dots + a_nx^n \in Z_{>0}[x]$  menjadi polinomial koefisien integral positif. Proses *key generation* algoritma ini sebagai berikut:

1. Pilih matriks cincin  $A \in M_n(Z_N)$  dengan  $N$  merupakan perkalian dari bilangan prima  $p$  dan  $q$
2. Hitung matriks dasar  $B = f(A) \bmod N$ .
3. B mengambil bilangan bulat acak rahasia  $d_1$  dalam interval  $[1, n-1]$ .
4. B menghitung  $Q_1 = B^{d_1} \bmod N$ . Diketahui  $Q_1$  publik dan  $d_1$  rahasia.
5. Jadi, kunci rahasia adalah  $(f(A), d_1)$  dan kunci publik adalah  $(Q_1, B)$ .

Contoh:

$$p = 11 \quad q = 13 \quad p \neq q$$

$$N = p \times q$$

$$N = 143$$

$$f(x) = 5x^2 + 3x + 7$$

$$A = \begin{bmatrix} 5 & 7 \\ 3 & 6 \end{bmatrix}$$

Hitung matriks dasar  $B = f(A) \bmod N$

$$B = f(A) \bmod N$$



$$B = 5(A^2) + 3(A) + 7 \bmod N$$

$$B = 5 \left( \begin{bmatrix} 5 & 7 \\ 3 & 6 \end{bmatrix}^2 \right) + 3 \begin{bmatrix} 5 & 7 \\ 3 & 6 \end{bmatrix} + 7 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \bmod 143$$

$$B = 5 \left( \begin{bmatrix} 46 & 77 \\ 33 & 57 \end{bmatrix} \right) + \begin{bmatrix} 15 & 21 \\ 9 & 18 \end{bmatrix} + \begin{bmatrix} 7 & 0 \\ 0 & 7 \end{bmatrix} \bmod 143$$

$$B = \begin{bmatrix} 252 & 406 \\ 174 & 313 \end{bmatrix} \bmod 143$$

$$f(A) = \begin{bmatrix} 252 & 406 \\ 174 & 313 \end{bmatrix}$$

$$B = \begin{bmatrix} 252 & 406 \\ 174 & 313 \end{bmatrix} \bmod 143$$

$$B = \begin{bmatrix} 109 & 120 \\ 31 & 27 \end{bmatrix}$$

B mengambil  $d_1 \in [1, 142]$

$$d_1 = 17$$

B menghitung  $Q_1 = B^{d_1} \bmod N$ . Diketahui  $Q_1$  *public* dan  $d_1$  *rahasia*

$$Q_1 = B^{d_1} \bmod N$$

$$Q_1 = \begin{bmatrix} 109 & 120 \\ 31 & 27 \end{bmatrix}^{17} \bmod 143$$

$$Q_1 = \begin{bmatrix} 142 & 89 \\ 79 & 136 \end{bmatrix}$$

$(f(A), d_1)$  kunci *rahasia* dan  $(Q_1, B)$  kunci *publik*

### ***Encryption***

Proses enkripsi algoritma Inam-Kanwal-Zahid-Abid, A ingin berkomunikasi dengan B, kemudian A mengirim sebuah pesan  $M$  sebagai berikut:

1. A mengambil bilangan bulat acak  $d_2 \in [1, n - 1]$  dan menghitung

$$K = B^{d_2} \bmod N.$$

2. A menghitung  $K_1 = Q_1^{d_2} \bmod N$ .
3. Akhirnya, A mengirimkan pasangan *ciphertext*  $(C, K)$ , dimana C mendefinisikan sebagai  $C = (M * K_1) \bmod N$ .

Contoh:

A mengambil bilangan bulat dari  $d_2 \in [1 - 142]$  dan  $d_2 = 23$

$$K = B^{d_2} \bmod N$$

$$K = \left( \begin{bmatrix} 109 & 120 \\ 31 & 27 \end{bmatrix}^{23} \right) \bmod 143$$

$$K = \begin{bmatrix} 51 & 73 \\ 113 & 75 \end{bmatrix}$$

A menghitung  $K_1 = Q_1^{d_2} \bmod N$

$$K_1 = Q_1^{d_2} \bmod N$$

$$K_1 = \begin{bmatrix} 142 & 89 \\ 79 & 136 \end{bmatrix}^{23} \bmod 143$$

$$K_1 = \begin{bmatrix} 105 & 49 \\ 21 & 31 \end{bmatrix}$$

A mengirimkan pasangan *chipertext*  $(C, K)$ , dimana  $C = (M \times K_1) \bmod N$

$$M = \begin{bmatrix} 80 & 91 \\ 75 & 61 \end{bmatrix}$$

$$C = \left( \begin{bmatrix} 80 & 91 \\ 75 & 61 \end{bmatrix} \times \begin{bmatrix} 105 & 49 \\ 21 & 31 \end{bmatrix} \right) \bmod N$$

$$C = \begin{bmatrix} 10311 & 6741 \\ 9156 & 5566 \end{bmatrix} \bmod 143$$

$$C = \begin{bmatrix} 15 & 20 \\ 4 & 132 \end{bmatrix}$$

### ***Decryption***

Setelah mendapatkan *ciphertext* dari A, B melakukan dekripsi pesan *ciphertext* dengan B menghitung  $K_2 = (K)^{d_1} \bmod N$ . B menghitung untuk mendapatkan kembali *plaintext* aslinya.  $M = (C * K_2^{-1}) \bmod N$

Contoh

Setelah mendapatkan *ciphertext* dari A,B melakukan dekripsi *ciphertext* dengan B menghitung

$$K_2 = (K)^{d_1} \bmod N$$

$$K_2 = \begin{bmatrix} 51 & 73 \\ 113 & 75 \end{bmatrix}^{17} \bmod 143$$

$$K_2 = \begin{bmatrix} 105 & 49 \\ 21 & 31 \end{bmatrix}$$

B menghitung  $M = (C \times K_2^{-1}) \bmod N$

$$M = (C \times K_2^{-1}) \bmod N$$

$$M = \left( \begin{bmatrix} 15 & 20 \\ 4 & 132 \end{bmatrix} \times \begin{bmatrix} 71 & 40 \\ 58 & 139 \end{bmatrix} \right) \bmod 143$$

$$M = \begin{bmatrix} 2225 & 3380 \\ 7940 & 18508 \end{bmatrix} \bmod 143$$

$$M = \begin{bmatrix} 80 & 91 \\ 75 & 61 \end{bmatrix}$$

## 2.9 Cryptanalysis

*Cryptanalysis* adalah studi pemecahan kode dan *decoding* dengan mendapatkan akses ke kunci untuk memecahkan protokol kriptografi, menghancurkan teknik otentikasi, dan mendeteksi serta memperbaiki kelemahan dalam sistem kriptografi (Munir, *et al.*, 2021). *Cryptanalyst* dapat melakukan kriptanalisis dengan memanfaatkan karakteristik algoritma yang memiliki saluran yang tidak aman terhadap jenis serangan. Metode serangan kriptanalisis dapat dikategorikan menjadi beberapa tipe umum, bergantung pada informasi apa yang diketahui dan tidak diketahui oleh pemecah kode.

### i. *Ciphertext-only attack*

*Ciphertext-only attack* hanya diketahui *ciphertext*. *Cryptanalyst* bertujuan untuk menemukan *plaintext* dan kuncinya.

### ii. *Known-plaintext attack*

*Cryptanalyst* diberikan beberapa kata sandi dan *plaintext* yang sesuai. Dalam kasus bagian pertama setiap *ciphertext* dapat didekripsi karena selalu sama. Tujuan *known-plaintext attack* untuk menemukan kunci dari *ciphertext* lain yang akan di dekripsi (Wagstaff, 2019).

### iii. *Chosen-plaintext attack*

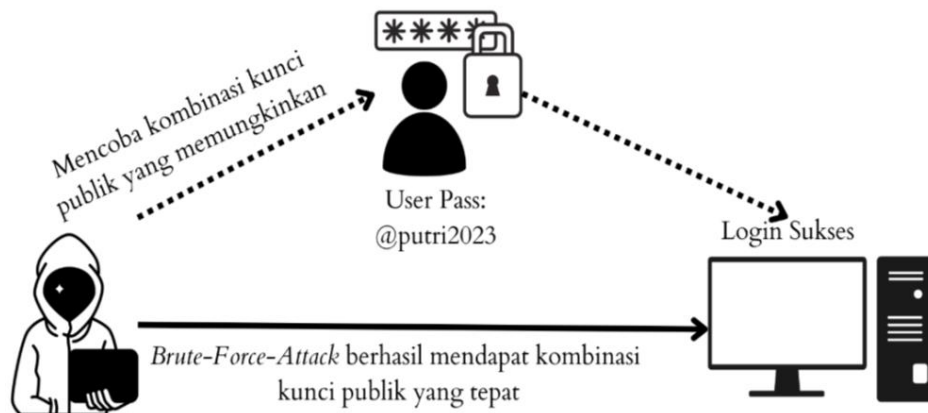
*Cryptanalyst* menentukan *plaintext* bahkan pesan yang tidak berarti dan mempelajari *ciphertext* yang benar. Hal ini bertujuan untuk mengelabui operator mesin *cipher* agar pesan tertentu menangkap *cipher* yang tidak dapat dibaca terukir didalam mesin. Misalnya untuk menemukan kunci (Wagstaff, 2019).

### iv. *Chosen-ciphertext attack*

*Cryptanalyst* menentukan beberapa *ciphertext* dan mempelajari *plaintext* yang sesuai. *Plaintext* yang diperoleh mungkin saja bukan pesan yang berarti, tetapi *plaintext* dapat membantu dalam menemukan kunci.

### 2.10 Brute-Force Attack

Serangan *Brute Force* merupakan serangan yang memakan waktu karena besarnya nilai kemungkinan dari kunci privat yang harus dicari. Serangan ini berbasis skema padding memerlukan memori dalam jumlah besar untuk membuat daftar tabel semua kemungkinan teks biasa, nilai pesan, dan nilai sandi terkait. *Cryptanalysis* diberikan kunci publik dan *ciphertext* untuk mencoba memaksa semua kunci yang mungkin untuk mendekripsikan *ciphertext*. Kriptanalis menghasilkan nilai M acak untuk dienkripsikan dengan dengan kriptosistem. Nilai M enkripsi akan di dekripsi dengan semua kunci privat yang mungkin sehingga nilai dekripsi sama dengan nilai M yang asli. Apabila *weak key* yang dapat dengan mudah diketahui akan menyebabkan kerentanan untuk ditembus oleh serangan *brute force attack* selama panjang kuncinya tidak cukup besar (Tsai & Choi, 2021). *Weak key* mudah diserang ketika kata sandi yang dipilih pendek, umum seperti karakter 1,2,3,4,5,0, @, q, w, e, r, t, y, dan lemah. Apabila menjadi sangat rumit dan tidak memberikan hasil ketika karakter khusus apa pun dimasukkan sebagai kata sandi.



**Gambar 2.4** Brute-Force Attack

### 2.11 Penelitian Relevan

2.11.1 Berdasarkan hasil analisis penelitian yang telah dilakukan oleh Whitfield Diffie dan Martin Hellman pada tahun 1976 dengan judul “*New Directions in*

*Cryptography*” menghasilkan setiap *user* menghasilkan sepasang *inverse transformation*, E dan D, di terminalnya. Transformasi dekripsi D harus tetap dirahasiakan, tetapi tidak perlu dikomunikasikan melalui saluran apa pun. Kunci enkripsi E dapat dipublikasikan dengan menempatkannya di public bersama dengan nama dan alamat *user*. Siapa pun dapat mengenkripsi pesan dan mengirimkannya ke *user*, tetapi tidak ada orang lain yang dapat dekripsi pesan yang ditujukan untuknya. Kriptosistem kunci publik dengan demikian dapat dianggap sebagai *multiple access ciphers* (Diffie & Hellman, 1976).

- 2.11.2 Berdasarkan hasil analisis penelitian yang dilakukan oleh Rivest Shamir Adleman pada tahun 1978 dengan judul “*A Method for Obtaining Digital Signature and Public Key Cryptosystems*” menjelaskan Diffie Hellman menggunakan fungsi *one-way* dari konsep *trap-door*. Fungsi ini dipanggil *one-way* karena mudah untuk dikomputasi dengan satu arah, tetapi sulit untuk dihitung melalui arah yang lain. Dipanggil *trap-door* karena fungsi *invers* sebenarnya mudah dihitung apabila informasi *trap-door* tertentu diketahui. Fungsi *trapdoor* harus berupa permutasi, setiap pesan adalah *ciphertext* untuk beberapa pesan lainnya dan setiap *ciphertext* itu sendiri merupakan pesan yang diperbolehkan (Rivest, *et al.*, 1978).
- 2.11.3 Berdasarkan hasil analisis penelitian yang dilakukan oleh Michael O. Rabin dengan judul “*Digitalized signatures and public-key functions as intractable as factorization*” menjelaskan kemampuan memecahkan kode pesan setara dengan kemampuan memfaktorkan bilangan yang besar. Faktanya, untuk  $n$  tertentu, algoritma *invers* hanya efektif pada sebagian kasus yang menghasilkan faktorisasi  $n$ . Kunci tetap  $n$  yang faktorisasinya tidak diberikan, *invers* pasti sulit dikarenakan hampir semua pesan sangat penting (Rabin, 1979).
- 2.11.4 Berdasarkan hasil analisis penelitian yang dilakukan oleh Taher ElGamal dengan judul “*A Public Key Cryptosystem and a signature scheme based on discrete logarithms*” menjelaskan kriptografi kunci publik berdasarkan kesulitan dalam menghitung logaritma diskrit pada bidang berhingga. Sistem kunci publik dapat dengan mudah diperluas ke  $p$ , tetapi dalam menghitung  $p$

dimana  $m$  besar membuat ukuran kunci yang diperlukan menjadi sangat besar untuk sistem menjadi aman (ElGamal, 1985).

- 2.11.5 Berdasarkan hasil analisis penelitian yang dilakukan oleh Menezes dengan judul “*Elliptic Curve Cryptosystems and Their Implementation*” menjelaskan *Elliptic Curve* mempunyai potensi memberikan keamanan yang setara dengan skema kunci publik yang ada dengan adanya kunci yang lebih pendek. Panjang kunci merupakan faktor yang sangat penting dalam berbagai aplikasi. Aritmatika *processor* pada *smart card* beresiko di ukuran yang berada di sekitaran  $20 \text{ m}^2$  (Menezes, 1993).

## BAB III

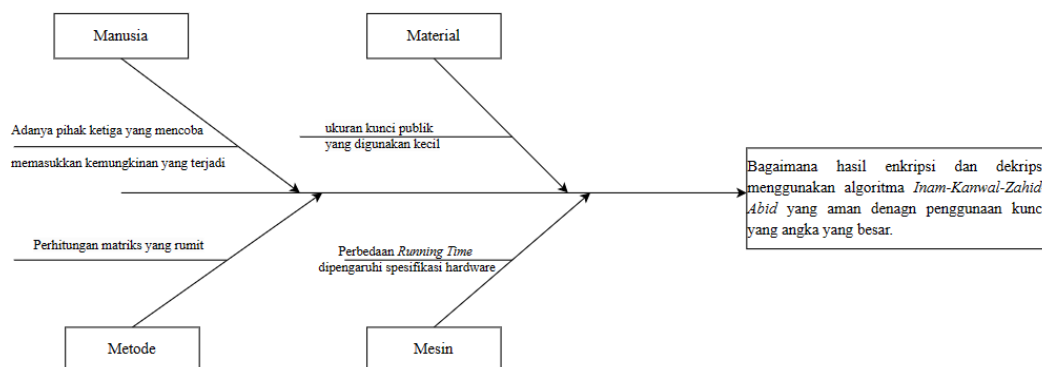
### ANALISIS DAN PERANCANGAN SISTEM

#### 3.1 Analisis Sistem

Perancangan dan analisis sistem dibutuhkan dalam membangun sistem. Metode yang digunakan untuk mengidentifikasi masalah dan mengembangkan solusi untuk mencapai sasaran dan kebutuhan dari sistem.

##### 3.1.1 Analisis Masalah

Permasalahan yang akan diidentifikasi adalah menguji *ciphertext* yang dihasilkan algoritma *Inam-Kanwal-Zahid-Abid* dapat di dekripsi dengan mudah melalui kunci publik yang diketahui oleh *brute-force attack*. Untuk menyelesaikan permasalahan sistem yang dibangun akan digunakan *fishbone diagram* (*Ishikawa Diagram*), *diagram ishikawa* merupakan representasi yang memvisualisasikan hubungan antara suatu masalah dan kemungkinan penyebabnya. Desain *fishbone diagram* terlihat seperti kerangka ikan yang dikerjakan dari kanan ke kiri dengan setiap tulang besar akan bercabang sehingga mencakup tulang-tulang kecil yang berisi lebih banyak detail. Gambaran diagram ishikawa diagram ditunjukkan pada Gambar 3.1.



**Gambar 3.1** Diagram Ishikawa



### 3.1.2 Analisis Kebutuhan

Analisis kebutuhan bertujuan untuk menjelaskan komponen yang menjadi bagian kebutuhan penelitian yang dilakukan.

#### 3.1.2.1 Kebutuhan Fungsional

Kebutuhan yang mendeskripsikan bagian yang akan diteliti seperti *input*, *output*, proses. Berikut kebutuhan fungsional pada penelitian ini:

- a. Sistem mengidentifikasi inputan *string* yang dimasukkan dan menampilkan hasil *plaintext*.
- b. *Plaintext* memiliki arti yang bermakna, sedangkan *cipher* akan menghasilkan ASCII yang tidak dimengerti
- c. Sistem mampu menjalankan proses *enkripsi*, *dekripsi* inputan *string* dengan menggunakan algoritma *Inam-Kanwal-Zahid-Abid*.

#### 3.1.2.2 Kebutuhan non-fungsional

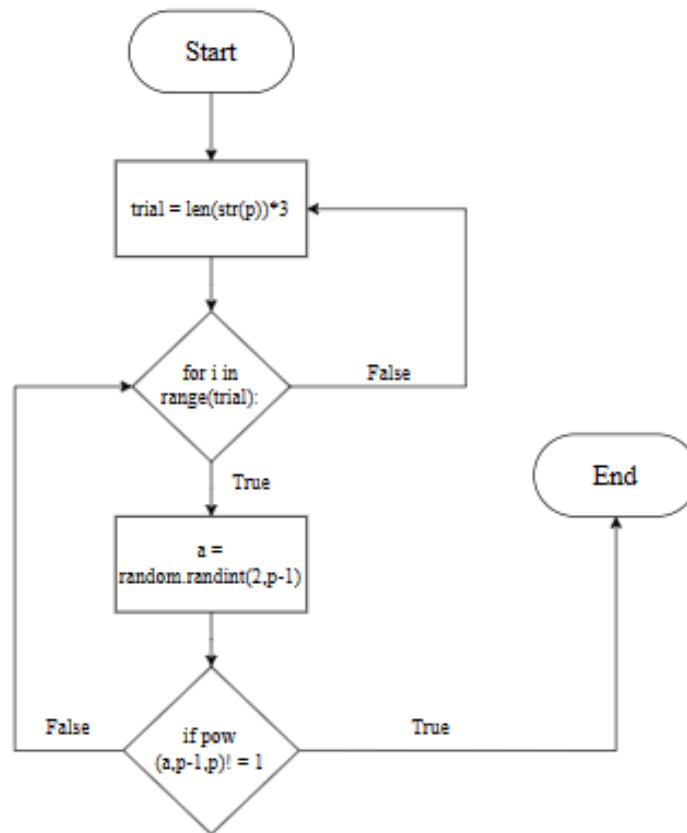
kebutuhan yang digunakan sebagai acuan pada kinerja sistem sesuai atau tidak dengan melengkapi sistem. Berikut kebutuhan non-fungsional pada penelitian ini:

- a. Efisiensi  
Sistem yang dirancang berupa program sehingga mudah dipahami oleh *user*.
- b. Ekonomis  
Sistem tidak membutuhkan biaya tambahan pada proses pembuatan sistem
- c. Kualitas  
Sistem dirancang dengan baik seperti merespon setiap *input* dengan cepat

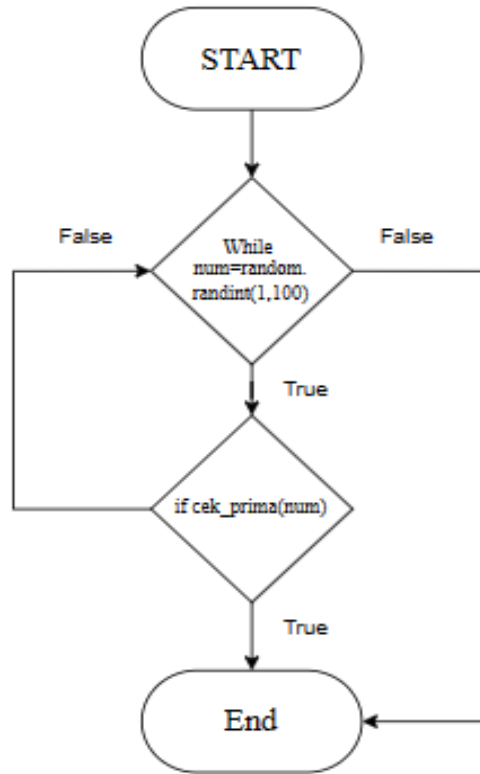
### 3.2 Flowchart

#### 3.2.1 Flowchart Pembangkit Kunci *Fermat's Little Theorem*

Pada pembangkit kunci prima random digunakan *fermat's little theorem* karena mengeluarkan bilangan prima yang besar.



**Gambar 3.2** Flowchart Pembangkit Kunci



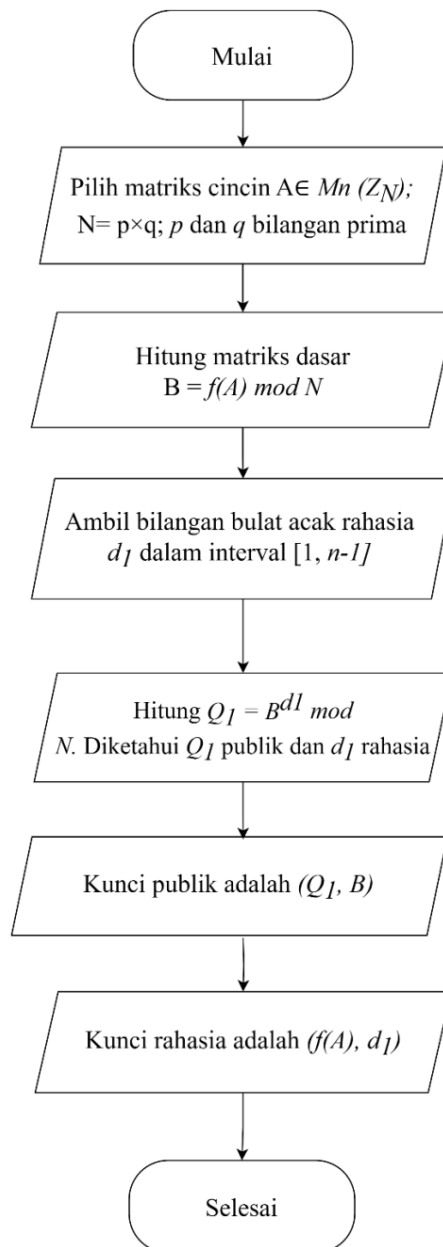
**Gambar 3.3** Flowchart Pembangkit Random Prima

Pada gambar 3.3 memperlihatkan *flowchart* dalam *generate* bilangan prima *random*. Jika angka bernilai kisaran 1 – 100 benar, maka nilai akan diperiksa dengan fungsi *cek\_prima(num)*.

### 3.2.2 Flowchart Key Generation

Pada proses *key generation* algoritma *Inam-Kanwal-Zahid-Abid* diperlukan 2 bilangan prima acak atau acak untuk mendapatkan nilai  $p$  dan  $q$  dengan  $N=pq$ , matriks cincin  $M_n(Z_N)$ .  $f(x) = a_0 + a_1x^1 + a_2x^2 + \dots + a_nx^n \in Z_{>0}[x]$  menjadi polinomial koefisien integral positif. Pada awal *key generation* pilih matriks cincin  $A \in M_n(Z_N)$  dengan  $N$  merupakan perkalian bilangan prima acak  $p$  dan  $q$ . Lalu, hitung matriks dasar  $B = f(A) \bmod N$ ,  $B$  mengambil bilangan bulat acak  $d_I$  dalam interval  $[1,$

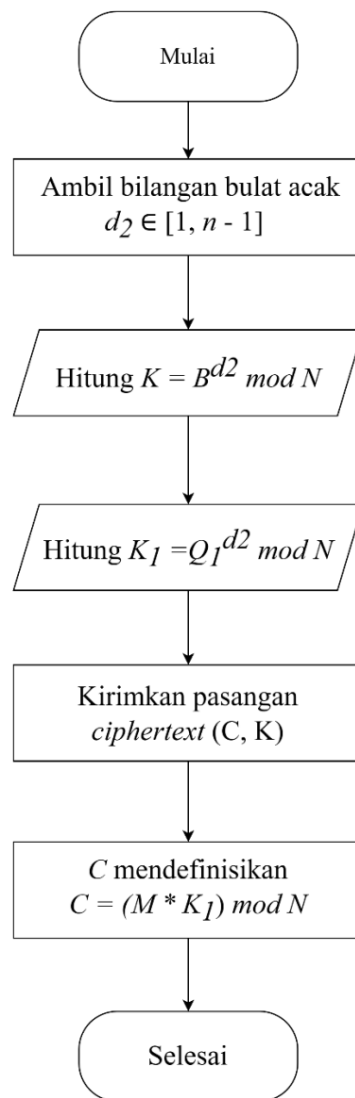
$n-1]$ . Selanjutnya,  $B$  menghitung  $Q_1 = B^{d_1} \bmod N$  dengan  $Q_1$  sudah diketahui publik dan  $d_1$  rahasia. Kunci rahasia ( $f(A)$ ,  $d_1$ ) dan kunci publik adalah ( $Q_1$ ,  $B$ ).



**Gambar 3.4** Flowchart Key Generation

### 3.2.3 Flowchart Proses Enkripsi

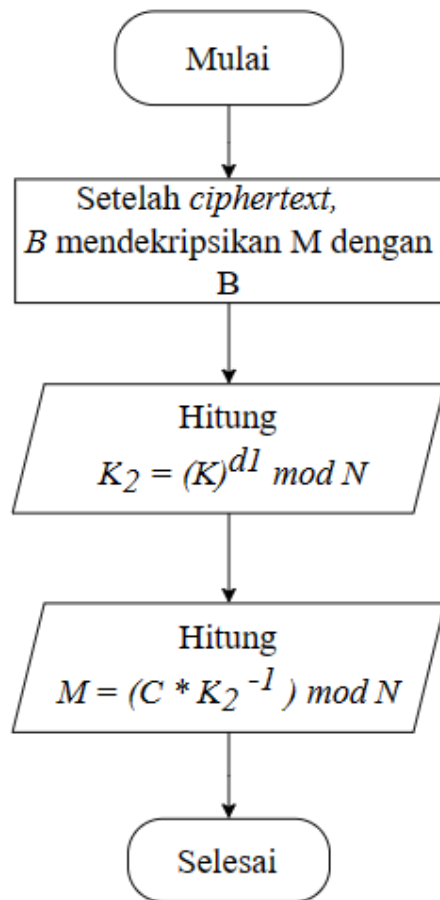
Pada proses enkripsi algoritma *Inam-Kanwal-Zahid-Abid*, A ingin berkomunikasi dengan B, kemudian A mengirim sebuah pesan  $M$  dengan ketentuan A mengambil bilangan bulat acak dari  $d_2 \in [1, n - 1]$  dan menghitung  $K = B^{d_2} \bmod N$ . Selanjutnya, A menghitung  $K_1 = Q_1^{d_2} \bmod N$ . Akhirnya, A mengirimkan pasangan *ciphertext*  $(C, K)$ , dimana C mendefinisikan sebagai  $C = (M * K_1) \bmod N$



**Gambar 3.5** Flowchart Enkripsi

### 3.2.4 Flowchart Proses Dekripsi

Setelah mendapatkan *ciphertext* dari A, B akan dekripsi pesan tersebut dengan B menghitung  $K_2 = (K)^{d_1} \bmod N$ . B menghitung untuk mendapatkan kembali *plaintext* aslinya.  $M = (C * K_2^{-1}) \bmod N$ .

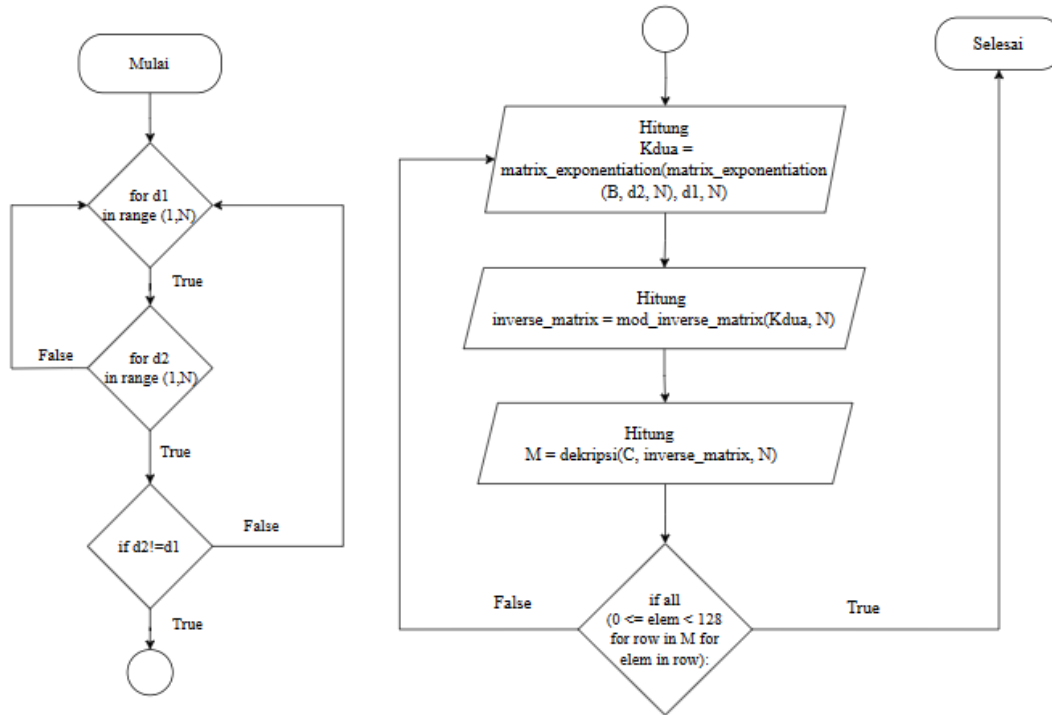


**Gambar 3.6** Flowchart Dekripsi

### 3.2.5 Flowchart Proses Brute-Force Attack

Pada proses *brute-force attack* pada  $d_1$  dalam range 1 hingga N dan  $d_2$  dalam range 1 hingga N. Jika  $d_2$  tidak sama dengan  $d_1$ , hitung  $K_{dua}$  dengan matriks eksponensial dan hitung invers matriks  $K_{dua}$  dengan N. Hitung M dengan menggunakan fungsi

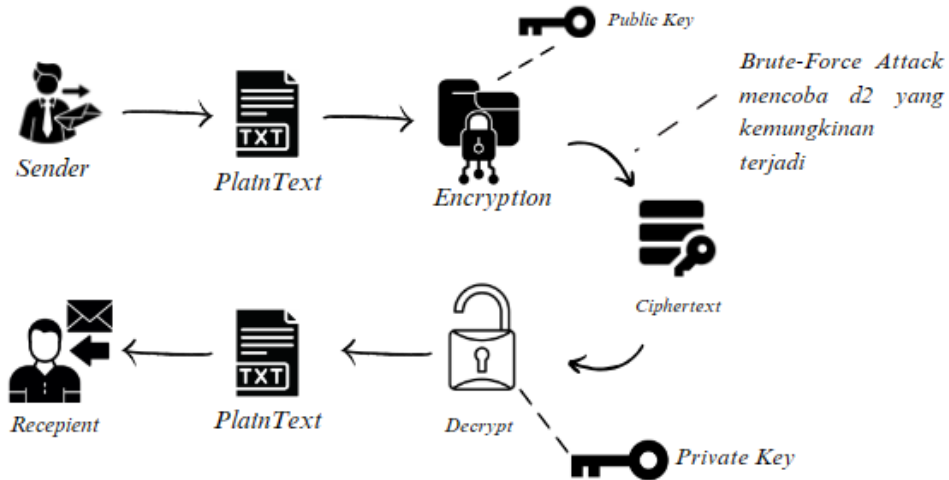
dekripsi, jika semua elemen lebih besar sama dengan 0 dan lebih kecil dari 128 untuk baris di M, kembalikan nilai M, d1, d2.



**Gambar 3.7** Flowchart Brute-Force Attack

### 3.3 Diagram Umum Sistem

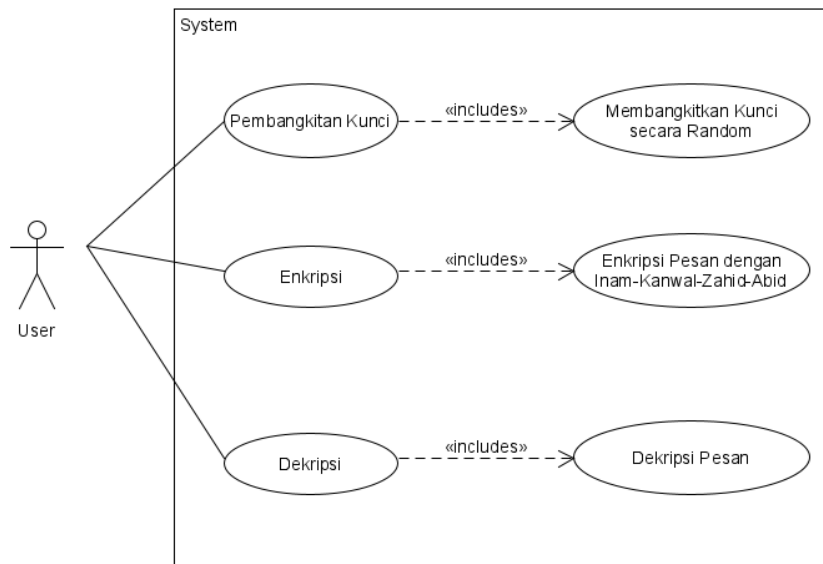
Diagram umum sistem memperlihatkan *sender* mengirimkan *plaintext*, lalu enkripsi dengan menggunakan algoritma *inam-kanwal-zahid-abid* dengan menggunakan *public key* dan menghasilkan *ciphertext*. *Ciphertext* akan melakukan dekripsi dengan *private key* dan mengembalikan nilai *plaintext* untuk *recipient*. *Brute-Force Attack* akan mencoba *d2* yang kemungkinan terjadi dengan *public key* dan *N* telah diketahui.



**Gambar 3.8** Diagram Umum Sistem

### 3.4 Use Case Diagram

Gambar 3.9 memperlihatkan hubungan interaksi dari sisi pengguna (*user*) terhadap sistem. Pengguna menggunakan pembangkit kunci secara *random*, kemudian dilanjutkan dengan proses enkripsi pesan dan dekripsi pesan.

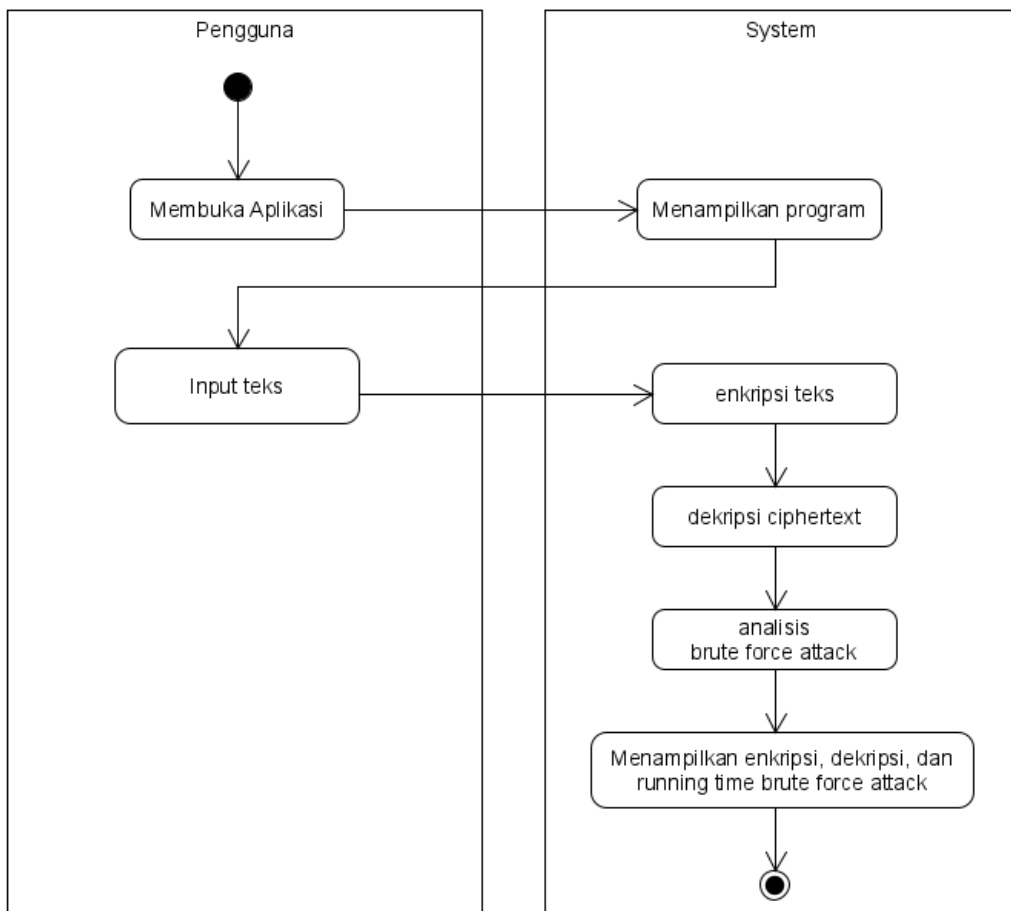


**Gambar 3.9** Use Case Diagram



### 3.5 Activity Diagram

Gambar 3.10 memperlihatkan hubungan interaksi dari sisi pengguna (*user*) terhadap sistem. Pengguna menggunakan pembangkit kunci secara *random*, kemudian dilanjutkan dengan proses enkripsi pesan dan dekripsi pesan.



**Gambar 3.10** Activity Diagram

## BAB IV

### IMPLEMENTASI DAN PENGUJIAN

#### 4.1 Implementasi

##### 4.1.1 Proses *Key Generation*

```
#KEY GENERATION
A = [[5, 7], [3, 6]]
p = prima_generate()
q = prima_generate()
N = p*q
# Polinomial f(x) = 5x^2 + 3x + 7
coefficients = [5, 3, 7]
B = matrix_polynomial_evaluation(A, coefficients, N)
dl = random.randint(1,N-1)
Q1 = matrix_exponentiation(B, dl, N)
```

**Gambar 4.1** Nilai *key generation*

Gambar 4.1 menjelaskan nilai dari *key generation* dimana terdapat nilai matriks A, *random prima* p dan q, pemanggilan fungsi B, dan pemanggilan fungsi dari Q1.

##### 4.1.2 Hasil Program

```
inputan: ini Audrey
p: 61 || q: 59
d1: 2157 || d2: 1009
N: 3599
Q1 [[1531, 1168], [2043, 2212]]
b: [[342, 560], [240, 422]]
K [[3559, 1392], [2139, 673]]
K1 [[294, 552], [1779, 887]]

Ciphertext: p mphymswcfw
Running time Enkripsi: 0.0065021514892578125

Dekripsi: ini audrey
Running time Dekripsi: 0.009949684143066406

Kunci d2 yang ditemukan: 2664
Plaintext hasil brute force decrypt: [[105, 105], [110, 110], [105, 105], [32, 32],
[65, 65], [117, 117], [100, 100], [114, 114], [101, 101], [121, 121]]
Running time Brute-Force Attack: 34.02444577217102
```

**Gambar 4.2** Hasil Program

Pada gambar 4.2 menampilkan hasil program yang menampilkan nilai  $p$ ,  $q$ ,  $d1$ ,  $d2$ ,  $N$ ,  $Q1$ ,  $B$ ,  $K$ ,  $K1$ , hasil *brute-force attack*, *ciphertext*, *plaintext*, *running time* enkripsi, *running time* dekripsi, dan *running time brute-force attack*.

#### 4.1.3 Inputan

```
#ENKRIPSI
d2 = random.randint(1,N-1)
K = matrix_exponentiation(B, d2, N)
K1 = matrix_exponentiation(Q1, d2, N)
inputan = "Halo,ini Audrey"
result = ascii_string_matrix(inputan)
C = enkripsi(inputan,K1,N)
karakter_ciphertext = [''.join([chr(num % 128)
                                for num in row])
                        for row in C]
enkripsi_string = ''.join(karakter_ciphertext)

#DEKRIPSI
Kdua = matrix_exponentiation(K, d1, N)
inverse_matrix = mod_inverse_matrix(Kdua, N)
M = dekripsi(C, inverse_matrix, N)
karakter_dekripsi = [''.join([chr(num % 128)
                               for num in row])
                     for row in M]
dekripsi_string = ''.join(karakter_dekripsi)

#PRINT
print ("inputan:",inputan)
print("d1:",d1,"|| d2:",d2)
print("N:",N)
print("Q1",Q1)
print("b:",B)
print("K",K)
print("K1",K1)
print("Ciphertext:",karakter_ciphertext)
print("Plaintext:",karakter_dekripsi)
```

**Gambar 4.3** Input fungsi program

Gambar 4.3 menjelaskan pemanggilan fungsi program enkripsi, dekripsi, dan *print* program penelitian algoritma *inam-kanwal-zahid-abid*.

## 4.2 Analysis running time

### 4.2.1 Key Generation

**Tabel 4.1** Pembangkit Kunci Bilangan Prima

	C	#	
Function cek_prima(n):			
if n ==2 or n==3:	C1	1	C1
return True	C2	1	C2
if n%2 == 0 or n<2;	C1	1	C1
return false	C2	1	C2
for x in range(5)	C3	1	C3
a = random.randint ( 2 , n-2)	C4	1	C4
if pow (a, n-1, n) != 1:	C1	1	C1
return false	C2	1	C2
return true	C2	1	C2
			<b><math>T(n) = \Theta(1)</math></b>

Pada tabel 4.1 menyajikan informasi perhitungan kompleksitas waktu pembangkit kunci bilangan prima dengan menggunakan algoritma fermat dan menghasilkan nilai  $T(n) = \Theta(1)$ .

**Tabel 4.2** Generate Bilangan Prima

	C	#	
Function prima_generate()			
While True	C1	1	C1
num=random.randomint(1,100)	C2	1	C2
If cek_prima(num	C3	1	C3
Return	C4	1	C4
			<b><math>= C1+C2+C3 T</math></b> <b><math>T(n) = \Theta(1)</math>.</b>

Pada tabel 4.2 menyajikan informasi analisis kompleksitas waktu *generate* bilangan prima dan menghasilkan nilai  $T(n) = \Theta(1)$ .

#### 4.2.2 Fungsi Enkripsi

**Tabel 4.3** Enkripsi *Inam-Kanwal-Zahid-Abid*

	C	#	
Function enkripsi(M, K, N):			
result = {	<b>C1</b>	<b>1</b>	<b>C1</b>
{			
sum char * K %N	<b>C2</b>	<b>n</b>	<b>C2n</b>
For x in K % N	<b>C3</b>	<b>n</b>	<b>C3n</b>
For y in K1	<b>C3</b>	<b>nn</b>	<b>C3n<sup>2</sup></b>
}			
for char in M	<b>C3</b>	<b>n</b>	<b>C3n</b>
}			
return result	<b>C4</b>	<b>1</b>	<b>C4</b>
			<b>= C1 +C2n+2C3n+</b> <b>C3n<sup>2</sup>+C4</b> <b>T(n) = <math>\Theta(n^2)</math></b>

Pada tabel 4.3 menyajikan informasi analisis perhitungan kompleksitas waktu enkripsi dengan menggunakan algoritma *Inam-Kanwal-Zahid-Abid* menghasilkan  $T(n) = \Theta(n^2)$ .

#### 4.2.3 Fungsi Dekripsi

**Tabel 4.4** Dekripsi *Inam-Kanwal-Zahid-Abid*

	C	#	
def dekripsi(C, X, N):			
result = {sum(C * y) % N	<b>C1</b>	<b>1</b>	<b>C1</b>
for k in X % N	<b>C2</b>	<b>n</b>	<b>C2n</b>
for j in X	<b>C2</b>	<b>n<sup>2</sup></b>	<b>C2n<sup>2</sup></b>
for i in C	<b>C2</b>	<b>n<sup>3</sup></b>	<b>C2 n<sup>3</sup></b>
}			
return result	<b>C3</b>	<b>1</b>	<b>C3</b>
			<b>= C1 + C2n + C2n<sup>2</sup> +</b> <b>C2n<sup>3</sup> + C3</b> <b>T(n) = Θ (n<sup>3</sup>)</b>

Pada tabel 4.4 menyajikan informasi perhitungan analisis kompleksitas waktu dekripsi dengan menggunakan algoritma *Inam-Kanwal-Zahid-Abid* menghasilkan  $T(n) = \Theta(n^3)$ .

#### 4.2.4 Fungsi *Brute-Force Attack*

**Tabel 4.5** Fungsi *Brute-Force Attack*

	C	#	
def brute_force_decrypt(C, Q, B, N):			
for d1 in range(1, N):	<b>C1</b>	<b>n</b>	<b>C1n</b>
for d2 in range(1, N):	<b>C1</b>	<b>n<sup>2</sup></b>	<b>C1 n<sup>2</sup></b>
if d2 != d1:	<b>C2</b>	<b>1</b>	<b>C2</b>
Kdua=expo(expo (B, d2, N), d1, N)			
invers= mod_inverse(Kdua, N)			
M = dekripsi(C, inverse_matrix, N)			

if all( $0 \leq \text{elem} < 128$ for row in M for elem in row):	<b>C2</b>	<b>1</b>	<b>C2</b>
return M, d1, d2	<b>C4</b>	<b>1</b>	<b>C4</b>
return None	<b>C4</b>	<b>1</b>	<b>C4</b>
			$= 2C2 + C1n + 2C4 + C1 n^2$ $T(n) = \Theta(n^2)$

Pada tabel 4.4 menyajikan informasi perhitungan analisis kompleksitas waktu *brute-force attack* dan menghasilkan nilai  $T(n) = \Theta(n^2)$ .

#### 4.3 Pengujian

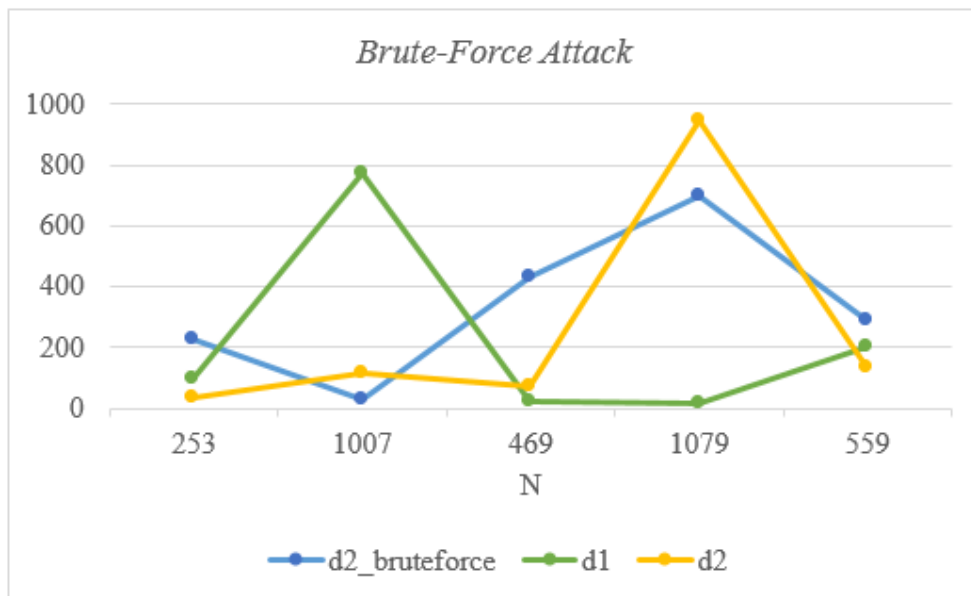
Dengan menggunakan *plaintext* “ini Audrey”, matriks A = [[5, 7], [3, 6]].

**Tabel 4.6** Brute-Force Attack dengan p dan q (1,100)

Percobaan	N	d1		d2		Running Time (ms)
		Sebelum BFA	Brute Force Attack	Sebelum BFA	Brute Force Attack	
Ke – 1	253	96	Tidak digunakan	37	228	498.887
Ke – 2	1007	770	Tidak digunakan	114	30	28.574
Ke – 3	469	24	Tidak digunakan	72	432	1636.048
Ke – 4	1079	16	Tidak digunakan	948	696	3254.36
Ke – 5	559	201	Tidak digunakan	133	287	1449.805

Pada Tabel 4.6 menunjukkan perbandingan nilai sebelum dan sesudah *brute-force attack* untuk mendapatkan *plaintext*. Berikut grafik yang menunjukkan perbandingan:

- d1, d2, dan d2\_bruteforce terhadap N

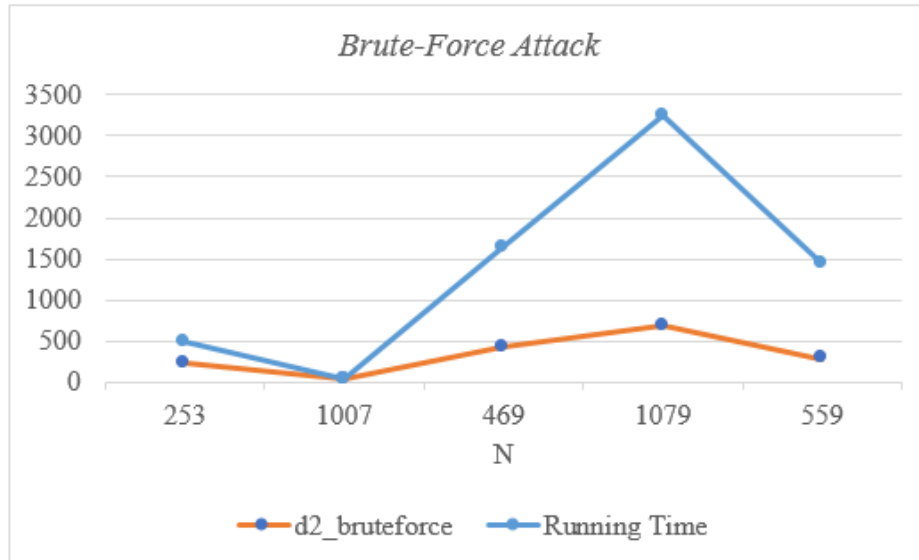


**Gambar 4.4** Grafik perbandingan d1, d2 dengan d2 *brute-force attack*

Pada Gambar 4.4 menampilkan grafik perbandingan nilai d1 sebelum *brute-force attack*, d2 sebelum *brute-force attack*, dan d2 setelah *brute-force attack* terhadap hasil nilai N.

- d2\_bruteforce dan *running time* terhadap N





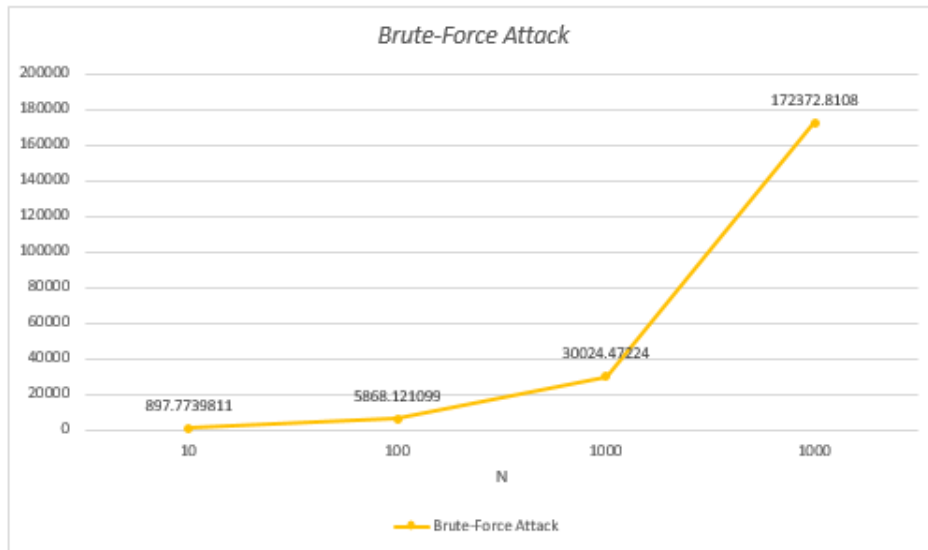
**Gambar 4.5** Grafik perbandingan d2 *brute-force attack* dengan *running time*

Pada gambar 4.5 menampilkan grafik perbandingan antara d2 setelah *brute-force attack* dengan *running time* setelah *brute-force attack* terhadap nilai N yang dihasilkan. Grafik menunjukkan d2 yang dihasilkan *brute-force attack* mempengaruhi kecepatan *running time*.

**Tabel 4.7** *Brute-Force Attack* dengan karakter yang berbeda

N	Running Time Brute-Force Attack (ms)					Rata – Rata (ms)
	Percobaan 1	Percobaan 2	Percobaan 3	Percobaan 4	Percobaan 5	
10	712.55016	1994.7726	713.4859	911.93485	156.12626	897.774
100	3382.4684	202.18491	25277.36	418.89047	59.693574	5868.12
1000	5699.1643	17945.114	22363.01	103501.13	613.92664	30024.4
10000	818325.15	831.29715	1943.460	22119.381	18644.761	172372.8

Pada tabel 4.7 menunjukkan tabel percobaan *running time brute-force attack* dengan menggunakan N yang berbeda.



**Gambar 4.6** *Running time brute-force attack*

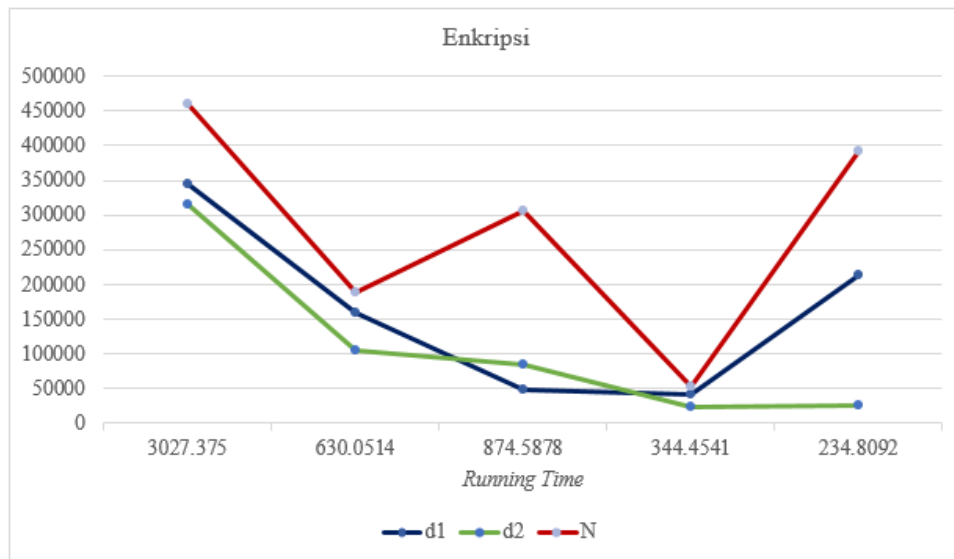
Pada gambar 4.6 menampilkan grafik *running time brute-force attack* terhadap nilai N yang berbeda. Grafik menunjukkan semakin panjang nilai karakter akan memperlambat *running time* untuk mendapatkan nilai *plaintext brute-force attack*.

#### 4.4 Pengujian Enkripsi

**Tabel 4.8** *Running Time Enkripsi*

Percobaan	d1	d2	N	<i>Running Time (milliseconds)</i>
Ke – 1	344134	314676	460631	3027.375
Ke – 2	158306	105676	189029	630.0514
Ke – 3	48272	84774	305713	874.5878
Ke – 4	40967	23899	52909	344.4541
Ke – 5	214062	26076	391271	234.8092

Pada tabel 4.8 menampilkan informasi mengenai nilai *running time* enkripsi algoritma *Inam-Kanwal-Zahid-Abid* dengan nilai d1, d2, dan hasil N yang berbeda.



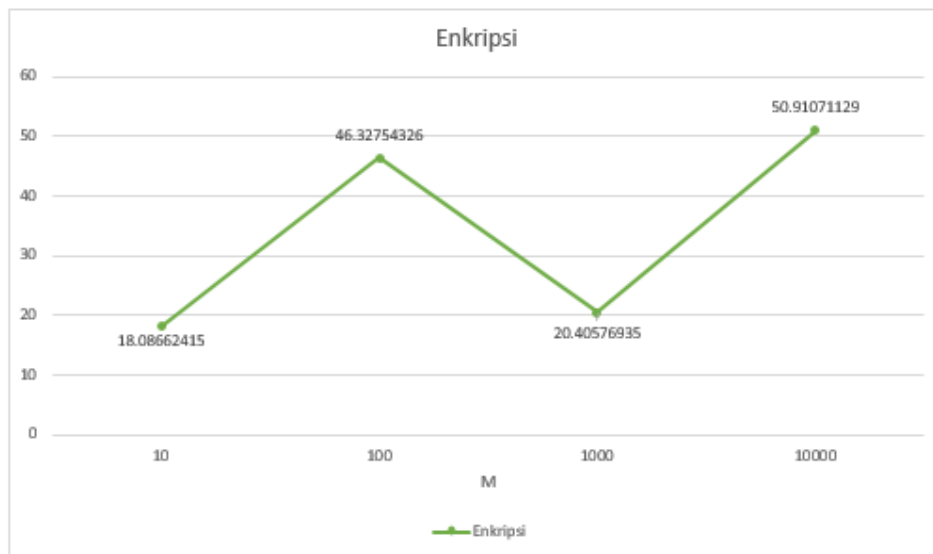
**Gambar 4.7** Nilai d1, d2, dan N

Pada gambar 4.7 menampilkan grafik pengujian enkripsi terhadap *running time* dengan menggunakan hasil nilai N yang berbeda.

**Tabel 4.9** *Running Time* Enkripsi terhadap M

M	Running Time Enkripsi (ms)					Rata – Rata (ms)
	Percobaan 1	Percobaan 2	Percobaan 3	Percobaan 4	Percobaan 5	
10	41.68176	10.617256	4.4245719	3.8132667	29.896259	18.0866
100	15.90299	105.99947	46.503782	38.732528	24.498939	46.3275
1000	6.536006	62.503337	7.3130130	13.513088	12.163400	20.4057
10000	53.54738	39.641380	30.147790	51.864147	79.352855	50.9107

Pada tabel 4.9 menampilkan tabel pengujian rata-rata *running time* enkripsi terhadap M dengan menggunakan algoritma *inam-kanwal-zahid-abid*.



**Gambar 4.8** *Running time* enkripsi dengan nilai M berbeda

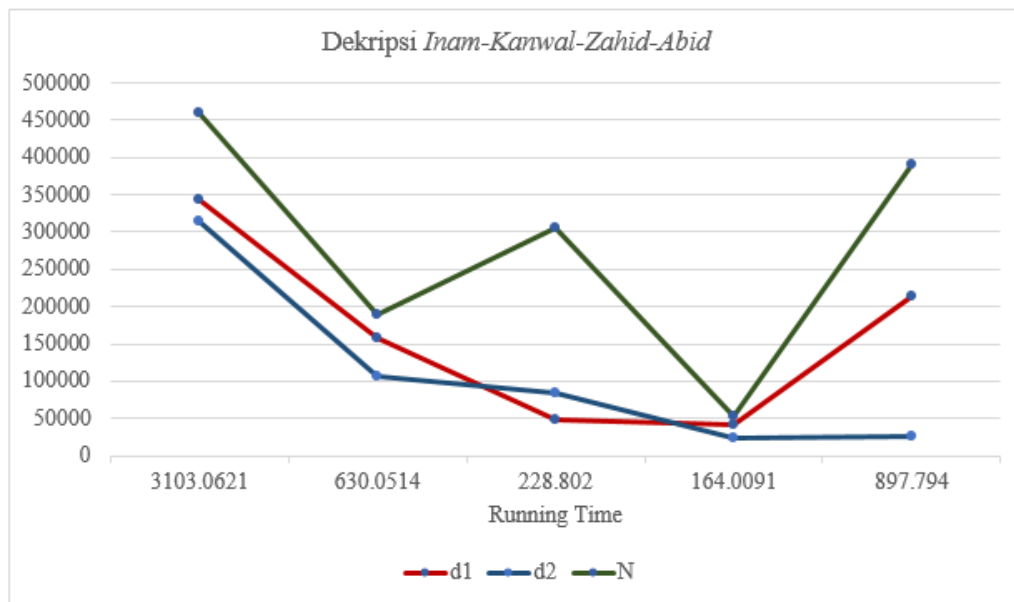
Pada gambar 4.8 menampilkan grafik *running time* enkripsi dengan menggunakan panjang karakter yang berbeda.

#### 4.5 Pengujian Dekripsi

**Tabel 4.10** Dekripsi *Inam-Kanwal-Zahid-Abid*

Percobaan			N	<i>Running Time (ms)</i>
	d1	d2		
Ke – 1	344134	314676	460631	3103.0621
Ke – 2	158306	105676	189029	630.0514
Ke – 3	48272	84774	305713	228.802
Ke – 4	40967	23899	52909	164.0091
Ke – 5	214062	26076	391271	897.794

Pada tabel 4.10 menampilkan informasi *running time* dekripsi dengan algoritma *Inam-Kanwal-Zahid-Abid* menggunakan nilai N, d1, dan d2 yang berbeda.



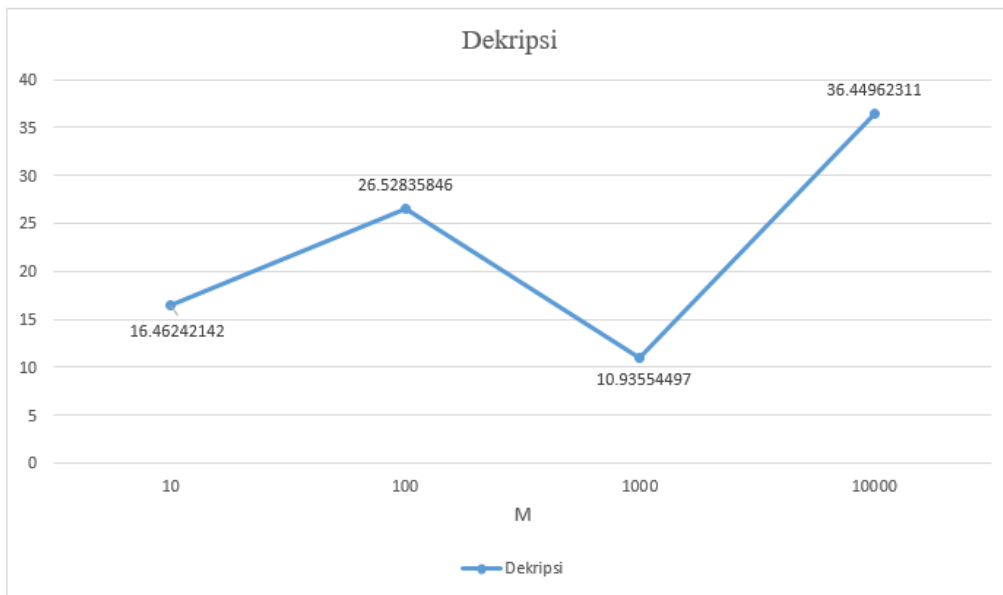
**Gambar 4.9** Grafik nilai dekripsi d1,d2, dan N

Pada gambar 4.9 menampilkan grafik nilai dekripsi d1,d2, dan N terhadap *running time* dekripsi *inam-kanwal-zahid-abid*.

**Tabel 4.11** *Running Time* Dekripsi terhadap M berbeda

M	Running Time Dekripsi (ms)					Rata – Rata (ms)
	Percobaan 1	Percobaan 2	Percobaan 3	Percobaan 4	Percobaan 5	
10	7.0936679 84	41.456460 95	13.612985 61	7.6274871 83	12.521505 36	16.46242 142
100	24.507045 75	45.062780 38	26.060819 63	17.138242 72	19.872903 82	26.52835 846
1000	5.0613880 16	27.640819 55	4.5757293 7	11.863470 08	5.5363178 25	10.93554 497
10000	41.861534 12	27.520179 75	33.753633 5	36.646366 12	42.466402 05	36.44962 311

Pada tabel 4.11 menampilkan tabel *running time* dekripsi terhadap panjang karakter M yang berbeda.



**Gambar 4.10** Perbandingan nilai *running time* dekripsi

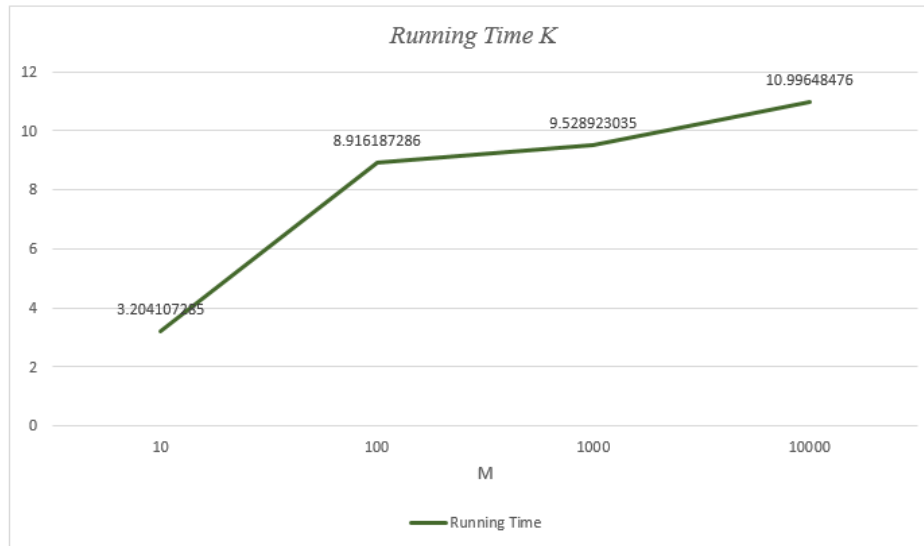
Pada gambar 4.10 menampilkan grafik pengujian nilai *running time* dekripsi dengan panjang karakter M yang berbeda. Grafik menunjukkan nilai *running time* tidak berbanding lurus terhadap nilai M.

#### 4.6 Pengujian *running time* K dengan menggunakan M yang berbeda

**Tabel 4.12** *Running Time K* dengan M yang berbeda

M	Running Time K (ms)					Rata-rata (ms)
	Percobaan 1	Percobaan 2	Percobaan 3	Percobaan 4	Percobaan 5	
10	2.99931	0.9985	3.99995	6.02078	2.002	3.20411
100	23.4811	0.99587	10.047	6.05702	3.99995	8.91619
1000	10.8416	13.5064	7.50852	7.73692	8.05116	9.52892
10000	0	16.8827	10.1879	16.0077	11.9042	10.9965

Pada tabel 4.12 menampilkan tabel rata - rata *running time K* dengan menggunakan panjang karakter M yang berbeda.



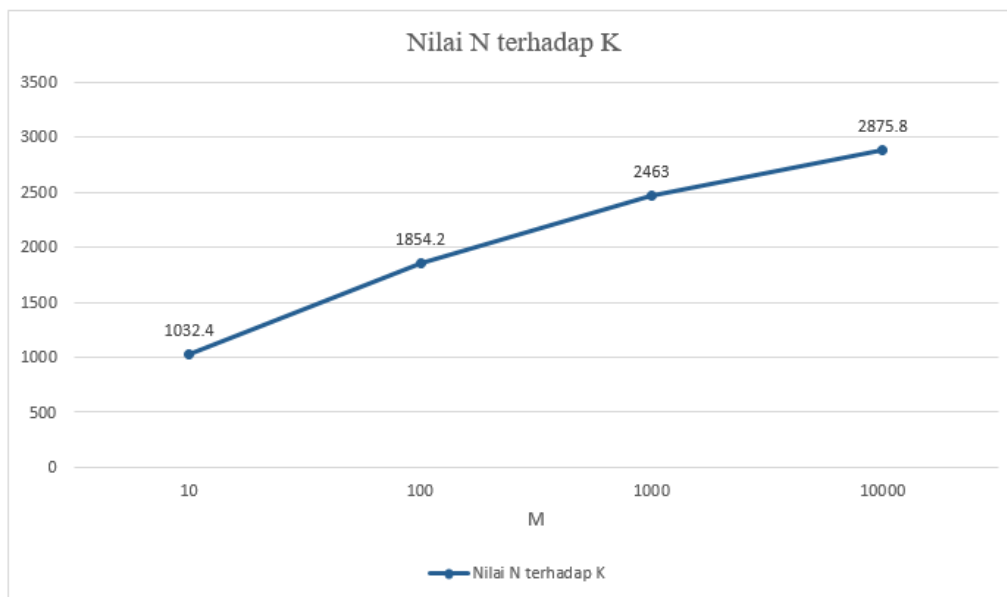
**Gambar 4.11** Perbandingan *running time K* terhadap nilai M berbeda

Pada gambar 4.11 menampilkan grafik *running time* untuk mencari K dengan panjang karakter M yang berbeda. Grafik menunjukkan *running time K* berbanding lurus dengan panjang karakter M.

**Tabel 4.13** Nilai N dalam mencari mencari K

M	Nilai N					Rata – Rata N
	Percobaan 1	Percobaan 2	Percobaan 3	Percobaan 4	Percobaan 5	
10	693	115	1653	2033	668	1032.4
100	3742	62	2630	1674	116	1854.2
1000	2592	2260	2777	2162	254	2463
10000	47	4337	3417	4596	1982	2875.8

Pada tabel 4.13 menampilkan nilai N yang dihasilkan untuk mencari K dengan Panjang Karakter M yang berbeda.



**Gambar 4.12** Nilai N terhadap K

Pada gambar 4.12 menampilkan grafik nilai N yang dihasilkan dalam mencari K. Grafik menunjukkan apabila panjang karakter M dan nilai N besar memungkinkan nilai K yang dihasilkan besar.



## **BAB V**

### **KESIMPULAN DAN SARAN**

#### **5.1 Kesimpulan**

Berdasarkan analisis penelitian “*Cryptanalysis Public Key Menggunakan Algoritma Inam-Kanwal-Zahid-Abid*” maka dapat disimpulkan:

1. Panjang kunci  $d_2$  mempengaruhi kecepatan dalam dekripsi algoritma *Inam-Kanwal-Zahid-Abid*. Apabila panjang  $d_2$  yang dihasilkan semakin besar maka, kecepatan *brute-force attack* akan semakin lama.
2. Nilai  $N$  yang dihasilkan pembangkit kunci prima mempengaruhi panjang kunci pada enkripsi dan dekripsi algoritma *Inam-Kanwal-Zahid-Abid*.

#### **5.2 Saran**

Saran yang bisa dijadikan bahan pertimbangan dalam penelitian berikutnya, antara lain:

1. Merancang program dapat melakukan enkripsi dan dekripsi dengan menggunakan *extension file* berupa .pdf, .jpg, .png.
2. Menghasilkan enkripsi dan dekripsi dengan menggunakan nilai  $p$  dan  $q$  satuan.

## DAFTAR PUSTAKA

- Abdulla, M., & Rana, M. E. (2021, September). Vulnerabilities in public key cryptography. In *3rd International Conference on Integrated Intelligent Computing Communication & Security (ICIIC 2021)* (pp. 627-631). Atlantis Press.
- Amin, A. M., & Mahamud, M. S. (2019, April). An alternative approach of mitigating arp based man-in-the-middle attack using client site bash script. In *2019 6th International Conference on Electrical and Electronics Engineering (ICEEE)* (pp. 112-115). IEEE.
- Brualdi, R. A., & Cvetkovic, D. (2008). *A combinatorial approach to matrix theory and its applications*. CRC press.
- BUDIMAN, A. (2020). RSA PUBLIC KEY SOLVING TECHNIQUE BY USING GENETIC ALGORITHM. *Journal of Theoretical and Applied Information Technology*, 98(15).
- Bufalo, Michele, Daniele Bufalo, and Giuseppe Orlando. "A Note on the Computation of the Modular Inverse for Cryptography." *Axioms* 10.2 (2021): 116.
- Diffie, W., & Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on information theory*, 22(6), 644-654.
- ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4), 469-472.
- Gottumukkala, R., Merchant, R., Tauzin, A., Leon, K., Roche, A., & Darby, P. (2019, April). Cyber-physical system security of vehicle charging stations. In *2019 IEEE Green Technologies Conference (GreenTech)* (pp. 1-5). IEEE.
- Inam, S., Kanwal, S., & Ali, R. (2021). A New Encryption Scheme Based on Grouping. *Contemporary Mathematics*, 103–112.  
<https://doi.org/10.37256/cm.222021611>

- Inam, S., Kanwal, S., Zahid, A., & Abid, M. (2020). A Novel Public Key Cryptosystem and Digital Signatures. *European Journal of Engineering Science and Technology*, 3(1), 22-30.
- Jarecki, S. (Ed.). (2020). *Topics in Cryptology—CT-RSA 2020: The Cryptographers' Track at the RSA Conference 2020, San Francisco, CA, USA, February 24–28, 2020, Proceedings* (Vol. 12006). Springer Nature.
- Johnson, C. R. (1990). *Matrix theory and applications* (Vol. 40). American Mathematical Soc.
- Mok, C. J., & Chuah, C. W. (2019). An Intelligence Brute Force Attack on RSA Cryptosystem. *Communications in Computational and Applied Mathematics*, 1(1).
- Munir, N., Khan, M., Shah, T., Alanazi, A. S., & Hussain, I. (2021). Cryptanalysis of nonlinear confusion component based encryption algorithm. *Integration*, 79, 41-47.
- Ott, D., & Peikert, C. (2019). Identifying research challenges in post quantum cryptography migration and cryptographic agility. *arXiv preprint arXiv:1909.07353*.
- Rabin, M. O. (1979). Digitalized signatures and public-key functions as intractable as factorization.
- Ribouh, S., Phan, K., Malawade, A. V., Elhillali, Y., Rivenq, A., & Al Faruque, M. A. (2020). Channel state information-based cryptographic key generation for intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 22(12), 7496-7507.
- Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2), 120-126.
- Shpilrain, V., & Ushakov, A. (2006). The conjugacy search problem in public key cryptography: unnecessary and insufficient. *Applicable Algebra in Engineering, Communication and Computing*, 17(3-4), 285-289.

- Tsai, M. Y., & Cho, H. H. (2021). A high security symmetric key generation by using genetic algorithm based on a novel similarity model. *Mobile Networks and Applications*, 26, 1386-1396.
- Verma, R., Dhanda, N., & Nagar, V. (2022). Enhancing security with in-depth analysis of brute-force attack on secure hashing algorithms. In *Proceedings of Trends in Electronics and Health Informatics: TEHI 2021* (pp. 513-522). Singapore: Springer Nature Singapore.
- Wagstaff Jr, S. S. (2019). *Cryptanalysis of number theoretic ciphers*. Chapman and Hall/CRC.
- Zhai, S., Yang, Y., Li, J., Qiu, C., & Zhao, J. (2019, February). Research on the Application of Cryptography on the Blockchain. In *Journal of Physics: Conference Series* (Vol. 1168, No. 3, p. 032077). IOP Publishing.

## **LAMPIRAN**