

***SIGNCRYPTION* DENGAN ALGORITMA RSA DAN SKEMA
TANDA TANGAN DIGITAL INAM – KANWAL – ZAHID – ABID**

SKRIPSI

A. NURCAHAYA TAMPUBOLON

211401071



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2025**

***SIGNCRYPTION* DENGAN ALGORITMA RSA DAN SKEMA TANDA TANGAN
DIGITAL INAM – KANWAL – ZAHID – ABID**

SKRIPSI

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

A. NURCAHAYA TAMPUBOLON

211401071



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2025**

PERSETUJUAN

Judul : *SIGNCRYPTION* DENGAN ALGORITMA RSA DAN
SKEMA TANDA TANGAN DIGITAL INAM –
KANWAL – ZAHID – ABID

Kategori : SKRIPSI

Nama : A. NURCAHAYA TAMPUBOLON

Nomor Induk Mahasiswa. : 211401071

Program Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA

Medan, 25 Maret 2025

Komisi Pembimbing:

Pembimbing II



Handrizal, S.Si, M.Comp.Sc
NIP. 197706132017061001

Pembimbing I



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

Diketahui/Disetujui Oleh

Program Studi S-1 Ilmu Komputer

Ketua



Dr. Amalia, ST, MT

NIP 197812212014042001

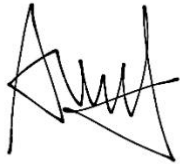
PERNYATAAN

SIGNCRYPTION DENGAN ALGORITMA RSA DAN SKEMA TANDA TANGAN
DIGITAL INAM – KANWAL – ZAHID – ABID

SKRIPSI

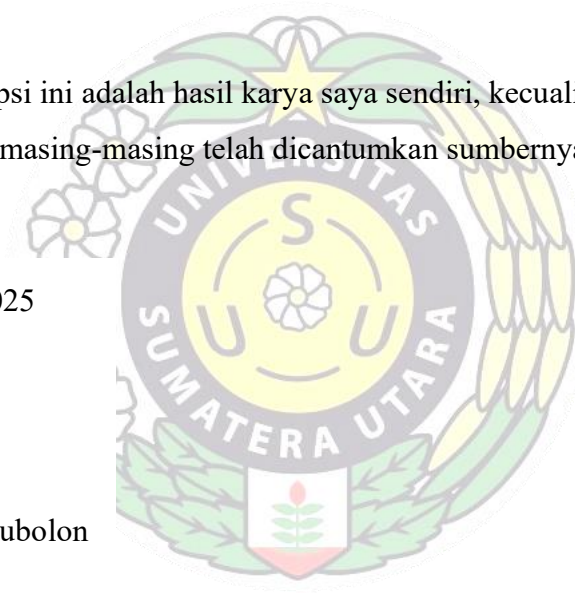
Saya mengakui skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah dicantumkan sumbernya.

Medan, 25 Maret 2025



A. Nurcahaya Tampubolon

211401071



PENGHARGAAN

Segala puji dan syukur kepada Tuhan Yesus Kristus atas berkat, penyertaan-Nya, dan juga perlindungan-Nya yang selalu menguatkan penulis dalam menyelesaikan skripsi yang berjudul “*Signcryption* dengan Algoritma RSA dan Skema Tanda Tangan Digital Inam – Kanwal – Zahid – Abid” sebagai syarat untuk memperoleh gelar Sarjana Komputer di Program Studi S-1 Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Sumatera Utara.

Penulis menyadari bahwa proses pengerjaan skripsi ini tidak lepas dari bimbingan, dukungan, dan bantuan dari berbagai pihak baik secara langsung maupun secara tidak langsung. Oleh karena itu, penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada :

1. Orangtua penulis yang tercinta, yang selalu mendukung, mendoakan, dan memberikan kasih sayang yang luar biasa sehingga penulis selalu mempunyai motivasi untuk mengerjakan dan menyelesaikan skripsi ini. Secara khusus, untuk Ibu *-my real superhero-* yang dengan penuh cinta selalu mengusahakan yang terbaik untuk anak-anaknya. Terima kasih atas segala kerja keras, pengorbanan, dan letih yang tak pernah Ibu tunjukkan, namun selalu penulis rasakan. Ibu adalah inspirasi terbesar penulis dan sumber kekuatan penulis dalam setiap langkah kehidupan.
2. Bapak Prof. Dr. Muryanto Amin, S.Sos., M Si selaku Rektor Universitas Sumatera Utara.
3. Ibu Dr. Maya Silvi Lydia, B.Sc., M.Sc. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Ibu Dr. Amalia, S.T., M.T., selaku Ketua Program Studi S1 Ilmu Komputer Universitas Sumatera Utara.
5. Bapak Dr. Mohammad Andri Budiman, S. T., M. Comp. Sc., M.E.M selaku Dosen Pembimbing I penulis yang telah memberikan bimbingan dan arahan serta mendukung penulis untuk menyelesaikan skripsi dengan tepat waktu.

6. Bapak Handrizal, S.Si, M.Comp.Sc selaku Dosen Pembimbing II penulis yang telah memberikan bimbingan dan arahan dalam menyelesaikan skripsi ini.
7. Bapak Fauzan Nurahmadi S.Kom., M.Cs., selaku Dosen Pembimbing Akademik (PA) penulis yang membimbing penulis selama berkuliah di S-1 Ilmu Komputer.
8. Bapak Prof. Drs. Poltak Sihombing, M.Kom., Ph.D dan Ibu Anandhini Medianty Nababan S.Kom., M.T selaku dosen penguji atas segala kritik dan saran yang membangun, yang telah membantu penulis untuk meningkatkan kualitas skripsi ini. Semoga kebaikan Bapak/Ibu dibalas dengan kebaikan pula.
9. Seluruh Bapak/Ibu Dosen dan Staf Pegawai Ilmu Komputer yang telah membantu penulis selama melaksanakan perkuliahan.
10. Saudara penulis, yaitu Sry, Septa, Herkules dan Lamsihar yang selalu memberikan semangat dan motivasi dalam setiap langkah di hidup penulis. Terkhusus *pudan* Alex Tampubolon yang selalu mendukung, mendoakan, dan memberikan humor di saat penulis *stuck* pada saat proses pengerjaan skripsi.
11. Ponakan tercinta penulis, Ivan, yang telah menjadi sumber hiburan dan keceriaan. Kelucuannya mampu menghadirkan senyum dan meningkatkan semangat penulis di tengah penatnya proses penyusunan skripsi.
12. Sahabat penulis Natalia Sibuea dan Johana Sihotang yang selalu menanyakan kabar penulis dan *progress* skripsi penulis, juga memotivasi penulis selama pengerjaan skripsi.
13. Sahabat penulis dalam masa perkuliahan Rodiatul Sitepu dan Yessica Situmorang yang selalu memotivasi dan mengajak penulis untuk mengerjakan skripsi supaya lulus tepat waktu juga menyediakan *basecamp* untuk mengerjakan skripsi bersama-sama.
14. Sahabat *HayDay* penulis, yakni Sammytha, Elin, dan Agatha yang telah memberikan motivasi dan semangat selama masa perkuliahan.
15. Member EXO, yakni Xiumin, Suho, Lay, Baekhyun, Chanyeol, Chen, D.O, Kai, Sehun yang sudah menemani penulis sejak SMA hingga masa kuliah dan telah menjadi penghibur dan penyemangat sepanjang perjalanan menyusun skripsi ini. Terutama untuk 도경수 (Do Kyungsoo), botak cimolku tersayang, yang selalu hadir lewat suara lembut, akting yang lucu, dan senyum sederhana yang

menenangkan. 고마워요, 경수야. Meski tak saling mengenal secara langsung, karya dan kehadiran kalian terasa dekat dan tulus. Terima kasih karena sudah ada, sudah bertahan, dan sudah terus berkarya. EXO, 사랑하자!

16. Terima kasih juga buat para *villain* Korea yang selalu hadir di waktu yang tepat — saat skripsi bikin pusing dan otak rasanya buntu. Lee Jun-yeong dengan misterinya, Wi Ha-joon dengan pesonanya, dan Ma Dong-seok yang selalu siap menghajar siapa pun (termasuk rasa malas). Tanpa sadar, kalian ikut bantu nyusun bab demi bab skripsi ini. Salut, *Oppa-oppa* jahat kesayangan penulis!
17. Semua pihak yang terlibat dalam mendukung dan membantu penulis yang tidak dapat penulis sebutkan satu-persatu dalam menyelesaikan skripsi ini. Semoga semua kebaikan, bantuan, perhatian, serta dukungan yang telah diberikan kepada penulis mendapatkan berkat dari Tuhan Yang Maha Esa.

Skripsi ini bukan hanya hasil dari berpikir dan menulis, tapi juga bentuk dari bertahan, percaya, dan memeluk setiap proses. Untuk segala ketidaksempurnaan yang dipeluk dengan sabar, untuk setiap detik yang tak terlihat oleh siapa pun, aku ucapkan: terima kasih, diriku. Kamu sudah sejauh ini. Dan kamu tidak pernah berjalan sendirian.

“Percayalah kepada TUHAN dengan segenap hatimu, dan janganlah bersandar kepada pengertianmu sendiri.”

— Amsal 3:5

Medan, 25 Maret 2025



A. Nurcahya Tampubolon

211401071

ABSTRAK

Di era digital saat ini, kebutuhan akan komunikasi yang aman semakin kritis mengingat ancaman terhadap privasi dan integritas data yang terus meningkat. Kebutuhan mendasar akan mekanisme keamanan yang komprehensif mendorong dilakukannya penelitian untuk merancang sistem yang dapat melindungi kerahasiaan data, menjamin integritas pesan, dan memberikan autentikasi yang kuat. Penelitian ini mengimplementasikan sistem *signcryption* yang menggabungkan enkripsi algoritma RSA dengan skema tanda tangan digital Inam-Kanwal-Zahid-Abid (IKZA) menyediakan komunikasi digital yang aman. Sistem *signcryption* pada penelitian ini menggunakan metode *encrypt-then-sign*, dimana algoritma RSA digunakan untuk proses enkripsi dan dekripsi, sedangkan skema tanda tangan digital IKZA diimplementasikan untuk proses penandatanganan dan verifikasi. Proses *signcryption* menunjukkan efisiensi bahkan untuk pesan yang panjang (500 karakter), dengan waktu rata-rata 0.017964 detik untuk *encrypt-then-sign* dan 0.03491 detik untuk *verify-then-decrypt*. Pengujian kinerja menunjukkan ketahanan yang kuat terhadap serangan brute-force, dengan angka 30-bit yang membutuhkan rata-rata 148,4 detik untuk difaktorkan, sementara angka 35-bit tidak dapat difaktorkan dalam batas waktu 3600 detik. Sistem *signcryption* yang diusulkan berhasil membuktikan efektivitasnya di *computer stand alone* dalam menyediakan keamanan komunikasi digital.

Kata kunci: *Signcryption*, Algoritma RSA, Tanda Tangan Digital, Inam-Kanwal-Zahid-Abid, *Encrypt-then-Sign*, Keamanan Informasi

SIGNCRYPTION WITH RSA ALGORITHM AND DIGITAL SIGNATURE SCHEME INAM-KANWAL-ZAHID-ABID

ABSTRACT

In today's digital age, the need for secure communication is increasingly critical given the ever-increasing threats to privacy and data integrity. The fundamental need for a comprehensive security mechanism prompted the research to design a system that can protect data confidentiality, guarantee message integrity, and provide strong authentication. This research implements a signcryption system that combines RSA algorithm encryption with Inam-Kanwal-Zahid-Abid (IKZA) digital signature scheme to provide secure digital communication. The signcryption system in this study uses the encrypt-then-sign method, where the RSA algorithm is used for the encryption and decryption process, while the IKZA digital signature scheme is implemented for the signing and verification process. The signcryption process shows efficiency even for long messages (500 characters), with an average time of 0.017964 seconds for encrypt-then-sign and 0.03491 seconds for verify-then-decrypt. Performance testing shows strong resistance to brute-force attacks, with 30-bit numbers taking an average of 148.4 seconds to factorize, while 35-bit numbers cannot be factored within the 3600-second time limit. The proposed signcryption system successfully proved its effectiveness on a stand alone computer in providing digital communication security.

Keywords: Signcryption, RSA Algorithm, Digital Signature, Inam-Kanwal-Zahid-Abid, Encrypt-then-Sign, Information Security

DAFTAR ISI

PERSETUJUAN	i
PERNYATAAN	ii
PENGHARGAAN	iii
ABSTRAK	vi
ABSTRACT	vii
DAFTAR ISI	viii
DAFTAR TABEL	xi
DAFTAR GAMBAR	xii
BAB I PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	2
1.3. Batasan Masalah	3
1.4. Tujuan Penelitian	3
1.5. Manfaat Penelitian	4
1.6. Metode Penelitian	4
1.7. Sistematika Penelitian	5
BAB II LANDASAN TEORI	6
2.1. Kriptografi.....	6
2.2. Sistem Kriptografi (<i>Cryptosystem</i>)	8
2.3. Matematika Kriptografi.....	9
2.3.1. <i>Bilangan prima</i>	9
2.3.2. <i>Matriks</i>	9

2.3.3.	<i>Prime generator</i>	10
2.3.4.	<i>Operasi modulo</i>	11
2.3.5.	<i>Invers modulo</i>	11
2.3.6.	<i>Greatest Common Divisor (GCD)</i>	12
2.3.7.	<i>Algoritma Square and Multiply (SAM)</i>	12
2.4.	<i>Algoritma RSA</i>	13
2.5.	<i>Digital Signature</i>	15
2.6.	<i>Digital Signature Inam – Kanwal – Zahid – Abid (IKZA)</i>	16
2.7.	<i>Teknik Signcryption dengan Algoritma RSA dan Digital Signature Inam – Kanwal – Zahid – Abid (IKZA)</i>	18
2.8.	<i>Penelitian Relevan</i>	18
BAB III ANALISIS DAN PERANCANGAN SISTEM		20
3.1.	<i>Analisis Sistem</i>	20
3.1.1.	<i>Analisis masalah</i>	20
3.1.2.	<i>Analisis kebutuhan</i>	22
3.1.3.	<i>Analisis proses</i>	23
3.2.	<i>Pemodelan Sistem</i>	24
3.2.1.	<i>Diagram umum sistem</i>	24
3.2.2.	<i>Use case diagram</i>	26
3.2.3.	<i>Activity diagram</i>	26
3.3.	<i>Flowchart (Diagram Alir)</i>	29
3.3.1.	<i>Flowchart sistem</i>	30
3.3.2.	<i>Flowchart Fermat's Litte Theorem</i>	30
3.3.3.	<i>Flowchart RSA</i>	31
3.3.4.	<i>Flowchart Inam-Kanwal-Zahid-Abid (IKZA)</i>	32

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM	35
4.1. Implementasi.....	35
4.1.1. Laman Home.....	35
4.1.2. Laman Key Generation	36
4.1.3. Laman Encrypt & Sign.....	36
4.1.4. Laman Verify & Decrypt	37
4.1.5. Laman Brute-force	38
4.2. Pengujian.....	38
4.2.1. Pengujian Key Generation.....	39
4.2.2. Pengujian Real Running Time Key Generation RSA	40
4.2.3. Pengujian Real Running Time Key Generation Inam-Kanwal-Zahid- Abid (IKZA).....	40
4.2.4. Pengujian Encrypt-then-Sign.....	41
4.2.5. Pengujian Real Running Time Encrypt-then-Sign	42
4.2.6. Pengujian Real Running Time Verify-then-Decrypt.....	44
4.2.7. Pengujian Brute-Force Attack	45
4.2.8. Pengujian Real Running Time Faktorisasi Brute-Force.....	46
BAB V PENUTUP.....	48
5.1. Kesimpulan	48
5.2. Saran.....	49
DAFTAR PUSTAKA	50

DAFTAR TABEL

Tabel 2.1 Perhitungan <i>Fermat's Little Theorem</i>	11
Tabel 2.2 Perhitungan invers dari $m \pmod{n}$	11
Tabel 2.3 Perhitungan d menggunakan Algoritma <i>Extended Euclidean</i>	14
Tabel 4.1 Waktu eksekusi <i>key generation</i> RSA terhadap panjang bit	40
Tabel 4.2 Waktu eksekusi <i>key generation</i> IKZA terhadap panjang bit.....	41
Tabel 4.3 Waktu eksekusi <i>encrypt RSA then sign</i> IKZA terhadap panjang pesan	43
Tabel 4.4 Waktu eksekusi <i>verify IKZA then decrypt RSA</i> terhadap panjang pesan....	44
Tabel 4.5 Waktu eksekusi Faktorisasi <i>Brute-Force</i>	46



DAFTAR GAMBAR

Gambar 2.1 Proses Enkripsi dan Dekripsi.....	6
Gambar 2.2 Proses Kriptografi Kunci Simetris.....	8
Gambar 2.3 Proses Kriptografi Kunci Asimetris.....	9
Gambar 2.4 Skema <i>Digital Signature</i>	16
Gambar 3.1 Diagram Umum <i>Key Generation</i> RSA.....	24
Gambar 3.2 Diagram Umum <i>Key Generation</i> IKZA	25
Gambar 3.3 Diagram Umum <i>Signcryption</i>	25
Gambar 3.4 <i>Use Case Diagram</i>	26
Gambar 3.5 <i>Activity Diagram</i> <i>Key Generation</i> RSA Penerima.....	27
Gambar 3.6 <i>Activity Diagram</i> <i>Key Generation</i> IKZA Pengirim	27
Gambar 3.7 <i>Activity Diagram</i> <i>Encrypt</i> dan <i>Sign</i>	28
Gambar 3.8 <i>Activity Diagram</i> <i>Verify</i> dan <i>Decrypt</i>	29
Gambar 3.9 <i>Flowchart</i> Sistem.....	30
Gambar 3.10 <i>Flowchart</i> <i>Fermat's Little Theorem</i>	31
Gambar 3.11 <i>Flowchart</i> <i>Key Generation</i> RSA.....	31
Gambar 3.12 <i>Flowchart</i> <i>Encryption</i> RSA	32
Gambar 3.13 <i>Flowchart</i> <i>Decryption</i> RSA.....	32
Gambar 3.14 <i>Flowchart</i> <i>Initialization</i> IKZA	33
Gambar 3.15 <i>Flowchart</i> <i>Key Generation</i> IKZA.....	33
Gambar 3.16 <i>Flowchart</i> <i>Signature Generation</i> IKZA	34
Gambar 3.17 <i>Flowchart</i> <i>Verification</i> IKZA	34
Gambar 4.1 Laman <i>Home</i>	35

Gambar 4.2 Laman <i>Key Generation</i>	36
Gambar 4.3 Laman <i>Encrypt and Sign</i>	37
Gambar 4.4 Laman <i>Verify and Decrypt</i>	37
Gambar 4.5 Laman <i>Brute-force</i>	38
Gambar 4.6 Pengujian <i>Key Generation</i>	39
Gambar 4.7 Hasil Pengujian <i>Key Generation</i>	39
Gambar 4.8 Grafik Pengujian <i>Real Running Time Key Generation RSA</i>	40
Gambar 4.9 Grafik Pengujian <i>Real Running Time Key Generation IKZA</i>	41
Gambar 4.10 Pesan asli	41
Gambar 4.11 Pengujian <i>Encrypt-then-Sign</i>	42
Gambar 4.12 Hasil Pengujian <i>Encrypt-then-Sign</i>	42
Gambar 4.13 Grafik Pengujian <i>Real Running Time Encrypt-then-Sign</i>	43
Gambar 4.14 Pengujian <i>Verify-then-Decrypt</i>	44
Gambar 4.15 Hasil Pengujian <i>Verify-then-Decrypt</i>	44
Gambar 4.16 Grafik Pengujian <i>Real Running Time Verify-then-Decrypt</i>	45
Gambar 4.17 Hasil Pengujian <i>Brute-Force</i>	45
Gambar 4.18 Grafik Pengujian <i>Real Running Time Brute-Force</i>	46

BAB I

PENDAHULUAN

1.1. Latar Belakang

Seiring dengan kemajuan teknologi saat ini, keamanan informasi menjadi aspek krusial dalam komunikasi elektronik. Pengiriman *file* digital, baik dalam bentuk dokumen, gambar, maupun data sensitif lainnya, sering kali menghadapi berbagai ancaman seperti peretasan, pencurian data, atau manipulasi informasi (Basri, 2015). Informasi di dalam *file* digital akan menjadi tidak valid jika sudah diketahui atau diretas oleh pihak yang tidak berkepentingan (Basharat et al., 2012).

Kebutuhan akan komunikasi yang aman juga menjadi semakin penting seiring dengan meningkatnya ancaman terhadap privasi dan integritas data. Dalam komunikasi, bertukar pesan harus menjamin keamanan, autentikasi, dan integritas pesan, sehingga pengirim tidak dapat menyangkal keaslian atau keterlibatannya dalam pengiriman pesan (Ginting et al., 2024). Hal ini memberikan jaminan yang lebih kuat karena penerima dapat memverifikasi dan memastikan bahwa pesan benar-benar berasal dari pengirim yang sah (Rivest et al., 1978).

Berbagai metode kriptografi telah dikembangkan untuk melindungi data digital dari ancaman seperti peretasan, pencurian data, dan manipulasi informasi. Salah satu inovasi dalam bidang ini adalah *signcryption*, yaitu teknik yang menggabungkan proses tanda tangan digital (*digital signature*) dan enkripsi dalam satu langkah yang efisien (Zheng, 1997). Istilah *signcryption* pertama kali diperkenalkan oleh Zheng. Metode ini termasuk dalam *Public Key Cryptography* yang dirancang untuk memenuhi kebutuhan tanda tangan digital dan enkripsi secara bersamaan. Dengan menggabungkan kedua fungsi tersebut, *signcryption* menawarkan keuntungan berupa pengurangan biaya komputasi dan ukuran data, tanpa mengorbankan aspek keamanan.

Salah satu algoritma yang banyak digunakan untuk implementasi *signcryption* adalah algoritma RSA, yang dikenal luas dalam dunia kriptografi berkat keamanan dan fleksibilitasnya. Algoritma RSA dibuat oleh 3 orang peneliti dari MIT (*Massachusetts*

Institute of Technology) pada tahun 1976, yaitu: Ron (R)ivest, Adi (S)hamir dan Leonard (A)dleman. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima. Selama belum ada algoritma yang cukup kuat atau efisien untuk memfaktorkan bilangan besar dengan cepat, keamanan algoritma RSA tetap terjaga (Arifin, 2016). RSA memiliki keunggulan utama dalam enkripsi pesan, yaitu mampu mengamankan komunikasi dengan menggunakan pasangan kunci publik dan privat. Kunci publik memungkinkan pesan dienkripsi oleh siapa saja, tetapi hanya pemilik kunci privat yang dapat mendekripsi dan membaca pesan tersebut. Hal ini menjadikan RSA sangat efektif dalam melindungi kerahasiaan data, terutama dalam sistem yang membutuhkan autentikasi dan integritas informasi.

Di sisi lain, skema tanda tangan digital yang dirancang oleh Inam - Kanwal - Zahid - Abid memberikan solusi unik untuk menjamin integritas dan otentikasi pesan. Skema ini telah diakui memiliki struktur matematika yang kuat dan efisien dalam menjamin keabsahan dokumen elektronik. Kombinasi *signcryption* dengan tanda tangan digital Inam - Kanwal - Zahid - Abid menawarkan potensi untuk menciptakan sistem yang tidak hanya aman, tetapi juga lebih efisien dibandingkan metode konvensional.

Penelitian ini bertujuan untuk mengimplementasikan *signcryption* berbasis algoritma RSA dengan mengintegrasikan skema tanda tangan digital Inam - Kanwal - Zahid - Abid. Selain itu, penelitian ini juga bertujuan untuk mengevaluasi efisiensi dan keamanan kombinasi algoritma ini terhadap serangan yang mungkin terjadi, sehingga dapat memberikan kontribusi signifikan dalam pengembangan teknologi keamanan data yang lebih baik.

1.2. Rumusan Masalah

Dalam penelitian ini, algoritma RSA digunakan sebagai dasar untuk proses enkripsi dan dekripsi, sedangkan skema tanda tangan digital yang dirancang oleh Inam - Kanwal - Zahid - Abid (IKZA) diimplementasikan untuk proses tanda tangan dan verifikasi. Permasalahan yang perlu diteliti adalah analisis keamanan implementasi *signcryption* berbasis RSA dengan skema tanda tangan digital IKZA dalam melindungi integritas dan kerahasiaan data terhadap berbagai jenis serangan siber yang mungkin terjadi. Selain itu, mengingat skema IKZA masih belum banyak digunakan secara luas, penelitian ini bertujuan untuk mengeksplorasi, mengevaluasi performa, dan memperkenalkan

algoritma ini sebagai solusi yang lebih efisien dan andal dalam pengembangan teknologi keamanan data modern.

1.3. Batasan Masalah

Adapun batasan penelitian agar tidak menyimpang dari rumusan masalah adalah:

1. Objek penelitian adalah implementasi dan analisis sistem *signcryption* yang mengintegrasikan algoritma RSA untuk proses enkripsi-dekripsi dengan skema tanda tangan digital IKZA berbasis matriks untuk proses tanda tangan dan verifikasi.
2. Skema tanda tangan digital menggunakan pendekatan berbasis matriks yang memiliki ordo 2×2 (representasi 4 elemen) dengan operasi matematis dalam ring Z_N .
3. Penelitian dilakukan di lingkungan pengembangan lokal pada komputer pribadi dengan menggunakan *file input* pada proses pengujian yang terbatas pada *file* berformat teks (.txt), dokumen (.docx), dan PDF (.pdf) tanpa mencakup format lain seperti gambar, video, atau data biner dengan maksimal jumlah karakter sebanyak 500.
4. Implementasi sistem menggunakan metode *encrypt-then-sign*, di mana pesan dienkripsi terlebih dahulu dengan RSA sebelum ditandatangani menggunakan skema IKZA.
5. Proses konversi teks menggunakan pengkodean ASCII dengan penambahan nilai 100 untuk setiap karakter supaya lebih mudah untuk proses dekripsi.
6. Proses *hashing* pada implementasi tanda tangan menggunakan pendekatan deterministik berbasis determinan matriks dan operasi aritmatika modular.
7. Implementasi sistem dilakukan menggunakan bahasa pemrograman *Python*.
8. Sistem tidak mencakup proses *cryptanalysis* untuk mendapat pesan asli.

1.4. Tujuan Penelitian

Penelitian ini bertujuan untuk mengimplementasikan dan menganalisis sistem *signcryption* yang mengintegrasikan algoritma RSA dengan skema tanda tangan digital IKZA berbasis matriks menggunakan pendekatan *encrypt-then-sign*. Selain itu, penelitian ini berusaha untuk menggali potensi algoritma IKZA yang masih belum banyak digunakan.

1.5. Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini, yakni:

1. Menyediakan implementasi sistem *signcryption* yang dapat diterapkan untuk mengamankan transmisi data sensitif pada berbagai format dokumen teks.
2. Penelitian ini memberikan kontribusi pada pengembangan ilmu pengetahuan dalam bidang kriptografi, khususnya terkait teknik *signcryption* yang menggabungkan enkripsi dan tanda tangan digital.
3. Mendorong adopsi dan pengembangan lebih lanjut terhadap algoritma IKZA yang masih belum banyak digunakan, sekaligus membuktikan potensinya dalam menghadapi tantangan keamanan modern.

1.6. Metode Penelitian

1. Studi Pustaka

Pada tahap ini penelitian dimulai dengan mencari referensi dari berbagai sumber terpercaya dan melakukan peninjauan pustaka melalui buku-buku, jurnal, e-book, artikel ilmiah, makalah ataupun situs internet yang berhubungan dengan skema *digital signature* IKZA dan algoritma RSA.

2. Perancangan Sistem

Tahap ini akan dilakukan analisis masalah yang dihadapi dengan mengidentifikasi masalah, memodelkan masalah secara konseptual dengan *flowchart* (diagram alir). Kemudian dilakukan perancangan *interface* aplikasi berisikan tahapan-tahapan operasi dalam proses pengolahan data dan prosedur untuk mendukung aplikasi tersebut.

3. Implementasi Sistem

Pada tahap ini, akan dilakukan implementasi algoritma RSA dan skema *digital signature* IKZA menggunakan bahasa pemrograman *Python* sesuai dengan *flowchart* yang telah dirancang sebelumnya.

4. Pengujian Sistem

Tahap ini akan dilakukan pengujian pada file teks untuk proses *sign* dan *verify* menggunakan *digital signature* IKZA, kemudian enkripsi dan dekripsi menggunakan algoritma RSA.

5. Dokumentasi

Seluruh tahapan penelitian, mulai dari studi pustaka, perancangan sistem, implementasi, dan pengujian, akan didokumentasikan dalam bentuk skripsi.

1.7. Sistematika Penelitian

Sistematika penulisan dalam penelitian ini disusun menjadi lima bab, yaitu:

BAB I PENDAHULUAN

Bab ini memaparkan berbagai aspek penting, seperti latar belakang yang melatarbelakangi penelitian, rumusan masalah, batasan ruang lingkup penelitian, tujuan penelitian, manfaat yang diharapkan. Selain itu, terdapat metodologi penelitian dan sistematika penulisan skripsi yang dijelaskan.

BAB II LANDASAN TEORI

Pada bab ini dibahas teori-teori dasar yang relevan dengan penelitian, termasuk konsep dasar *signcryption*, algoritma RSA untuk enkripsi, *digital signature* IKZA, dan penerapannya dalam sistem *signcryption*.

BAB II ANALISIS DAN PERANCANGAN SISTEM

Bab ini menguraikan analisis terhadap permasalahan serta perancangan sistem *signcryption* yang menggabungkan algoritma RSA untuk enkripsi dan *digital signature* IKZA untuk tanda tangan digital.

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini mencakup implementasi sistem *signcryption* yang telah dirancang sebelumnya, pengujian sistem menggunakan data teks, serta analisis hasil pengujian untuk mengevaluasi kinerja dan efektivitas sistem.

BAB V KESIMPULAN DAN SARAN

Bab terakhir menyajikan kesimpulan dari penelitian yang dilakukan, serta memberikan saran untuk pengembangan lebih lanjut di masa mendatang.

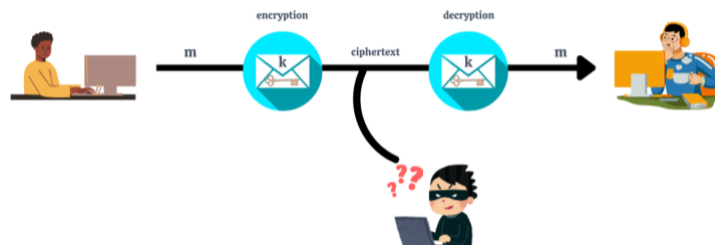
BAB II

LANDASAN TEORI

2.1. Kriptografi

Menurut *The Concise Oxford English Dictionary*, kriptografi adalah seni menulis atau memecahkan kode. Definisi ini hanya berfokus pada kode-kode yang telah digunakan selama berabad-abad untuk memungkinkan komunikasi rahasia. Namun, kriptografi saat ini mencakup lebih dari itu: kriptografi berhubungan dengan mekanisme untuk memastikan integritas, teknik untuk bertukar kunci rahasia, protokol untuk mengotentikasi pengguna, lelang dan pemilihan elektronik, uang digital, dan masih banyak lagi (Jonathan, 2015).

Kriptografi berkaitan dengan perancangan dan penggunaan kode (juga disebut sandi) yang memungkinkan dua pihak untuk berkomunikasi secara diam-diam di hadapan penyadap yang dapat memonitor/memantau semua komunikasi di antara mereka, dimana kode-kode tersebut disebut skema enkripsi. Keamanan dari semua skema enkripsi bergantung pada sebuah rahasia - sebuah kunci - yang dibagikan oleh pihak-pihak yang berkomunikasi sebelumnya dan tidak diketahui oleh penyadap. Skenario ini dikenal sebagai pengaturan *private-key* (atau *shared-/secret-key*), dan enkripsi *private-key* hanyalah salah satu contoh dari primitif kriptografi. Gambar 2.1 menunjukkan proses enkripsi dekripsi *private-key* secara umum.



Gambar 2.1 Proses Enkripsi dan Dekripsi

Dalam proses kriptografi, terdapat beberapa komponen dan tahapan penting (Munir, 2019).

1. Pengirim (*sender*), pihak yang mengirim pesan.
2. *Plaintext*
 - Merupakan data atau pesan asli sebelum dienkripsi.
 - Berisi informasi yang dapat langsung dibaca dan dipahami.
 - Menjadi input awal dalam proses kriptografi.
3. Proses Enkripsi
 - Menggunakan algoritma kriptografi tertentu.
 - Mengubah *plaintext* menjadi bentuk tersandi (*ciphertext*).
 - Membutuhkan kunci kriptografi sebagai parameter pengamanan.
4. Penerima (*recipient*), pihak yang menerima pesan.
5. *Ciphertext*
 - Hasil dari proses enkripsi.
 - Bentuknya berbeda dari data asli dan tidak dapat dibaca secara langsung.
 - Aman untuk disimpan atau ditransmisikan.
6. Proses Dekripsi
 - Menggunakan algoritma yang bersesuaian dengan proses enkripsi.
 - Memerlukan kunci yang tepat untuk mengubah ciphertext.
 - Mengembalikan data ke bentuk *plaintext*.
7. Kunci
 - Merupakan parameter rahasia yang berperan penting dalam proses enkripsi dan dekripsi data.
 - Kunci harus diketahui oleh pihak-pihak yang berwenang dalam komunikasi tersebut, namun harus tetap dijaga kerahasiaannya dari pihak-pihak yang tidak berkepentingan atau potensi penyerang.

Kriptografi memiliki empat tujuan atau layanan yang menjadi aspek keamanan informasi (Munir, 2019), yaitu:

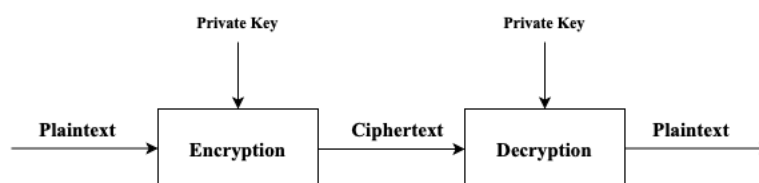
1. Kerahasiaan (*Confidentiality*), bertujuan untuk menjaga kerahasiaan informasi dari pihak-pihak yang tidak berhak mengaksesnya.
2. Integritas Data (*Data Integrity*), bertujuan untuk menjamin informasi masih asli/utuh dan tidak dimanipulasi pada saat pengiriman.

3. Otentikasi (*Authentication*), berhubungan dengan verifikasi kebenaran pihak-pihak yang berkomunikasi (*user authentication* atau *entity authentication*) maupun verifikasi kebenaran sumber pesan (*data origin authentication*).
4. Penyangkalan (*Nonrepudiation*), bertujuan untuk mencegah pihak-pihak yang berkomunikasi melakukan penyangkalan, yaitu pihak pengirim menyangkal melakukan pengiriman dan pihak penerima menyangkal telah menerima informasi.

2.2. Sistem Kriptografi (*Cryptosystem*)

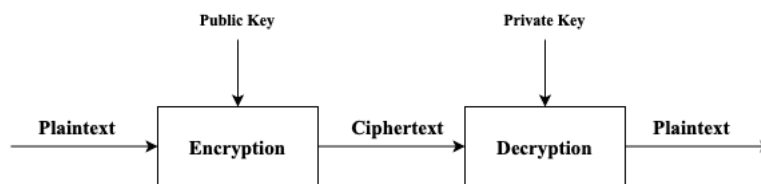
Secara umum ada dua jenis sistem kriptografi (Muchlis, 2015), yaitu:

1. Sistem kriptografi simetris (*Symmetric Cryptosystem*), disebut juga algoritma konvensional adalah algoritma kriptografi dimana kunci yang digunakan untuk proses enkripsi dan dekripsi sama, yaitu kunci privat. Pada Gambar 2.2 pengirim dan penerima harus menyepakati kunci privat yang akan digunakan sebelum mengirim pesan nantinya. Contoh sistem ini adalah DES, Rijndael, Blowfish, IDEA, dan GOST.



Gambar 2.2 Proses Kriptografi Kunci Simetris

2. Sistem kriptografi asimetris (*Asymmetric Cryptosystem*), disebut juga sistem kriptografi kunci-publik adalah algoritma kriptografi dimana kunci yang digunakan untuk proses enkripsi dan proses dekripsi berbeda. Kunci publik digunakan untuk proses enkripsi dan kunci privat digunakan untuk proses dekripsi seperti terlihat pada Gambar 2.3. Contoh sistem ini adalah RSA, Rabin, ElGamal, DSA, dan ECC.



Gambar 2.3 Proses Kriptografi Kunci Asimetris

Skema kunci publik hampir selalu hanya digunakan untuk mentransmisikan rahasia singkat per pesan, seperti *session key*. Hal ini dikarenakan skema kunci publik tidak efisien untuk digunakan mengenkripsi data dalam jumlah besar. Data yang sebenarnya kemudian dienkripsi menggunakan cipher simetris. Pendekatan seperti ini disebut skema *hybrid encryption* (Smart, 2016).

2.3. Matematika Kriptografi

Berikut beberapa dasar matematika yang digunakan dalam penelitian ini.

2.3.1. Bilangan prima

Bilangan prima adalah bilangan bulat positif p dimana $p > 1$ dan tepat hanya memiliki dua buah faktor, yaitu bilangan itu sendiri dan bilangan satu (Stallings, 2017). Bilangan prima sudah pasti adalah bilangan ganjil, sedangkan bilangan ganjil belum tentu adalah bilangan prima. Contoh bilangan prima, yaitu 2, 3, 11, 17, 19, 23, 97, dan sebagainya. Angka selain bilangan prima disebut bilangan komposit, yaitu bilangan yang memiliki faktor lebih dari 2. Contoh bilangan komposit adalah 4, 6, 8, 9, 10, dan lain-lain.

2.3.2. Matriks

Matriks adalah kumpulan skalar (bilangan riil atau kompleks) yang tersusun dalam bentuk persegi panjang berdasarkan baris dan kolom. Sebuah matriks, misalnya $A = (a_{ij})$, menunjukkan bahwa matriks A memiliki elemen-elemen a_{ij} , di mana indeks i menunjukkan baris ke- i , dan j menunjukkan kolom ke- j dari elemen tersebut (Nuraini, 2015).

Secara umum, sebuah matriks $A = (a_{ij})$ dengan indeks $i = 1, 2, \dots, m$ dan $j = 1, 2, \dots, n$, memiliki m baris dan n kolom pada matriks. Dua buah matriks $A = (a_{ij})$ dan $B = (b_{ij})$ disebut sama ($A = B$) jika kedua matriks tersebut memiliki kolom dan baris yang sama ($m_a \times n_a = m_b \times n_b$).

Contoh:

$$A = \begin{bmatrix} 2 & 3 \\ 5 & 7 \end{bmatrix}, B = \begin{bmatrix} 11 & 13 \\ 17 & 19 \end{bmatrix}$$

Matriks A dan B sama-sama memiliki baris 2 dan kolom 2.

Jika matriks A memiliki baris dan kolom yang sama dan jika matriks A^{-1} dapat dicari sedemikian rupa sehingga $AA^{-1} = A^{-1}A = I$, maka A dikatakan dapat dibalik (*invertible*) dan A^{-1} dinamakan invers dari A (Cipta, 2020). Misalkan matriks $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, maka nilai inverse dari matriks A adalah $A^{-1} = \frac{adj(A)}{\det(A)}$, dimana $adj(A) = \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$ dan $\det(A) = ad - bc$.

Contoh: $A = \begin{bmatrix} 2 & 3 \\ 5 & 7 \end{bmatrix}$

$$Adj(A) = \begin{bmatrix} 7 & -3 \\ -5 & 2 \end{bmatrix}$$

$$\det(A) = 2 \times 7 - 3 \times 5 = 14 - 15 = -1$$

$$\text{Maka } A^{-1} = \frac{\begin{bmatrix} 7 & -3 \\ -5 & 2 \end{bmatrix}}{-1} = \begin{bmatrix} -7 & 3 \\ 5 & -2 \end{bmatrix}.$$

2.3.3. Prime generator

Prime generator adalah sebuah metode dalam kriptografi yang digunakan untuk menghasilkan bilangan prima besar dan menguji apakah bilangan tersebut merupakan bilangan prima. Algoritma *Fermat's Little Theorem* (Riehm & Dudley, 1971) adalah salah contoh *prime generator* yang akan digunakan pada penelitian ini yang menyatakan “Bila ada sebuah bilangan bulat positif p yang memenuhi: $F = a^{p-1} \bmod p = 1$ untuk seluruh nilai a dalam rentang $1 < a < p$, maka p pasti adalah bilangan prima.” Namun, kebalikannya tidak selalu benar. Ada bilangan komposit n yang memenuhi $a^{p-1} \equiv 1 \pmod{p}$ untuk beberapa basis a . Bilangan-bilangan inilah yang disebut *Fermat's Liar* atau pseudoprima terhadap basis a (fenomena dalam teori bilangan ketika sebuah bilangan komposit lolos dari tes primalitas Fermat dan keliru diidentifikasi sebagai bilangan prima).

Untuk mengurangi kemungkinan terjadinya *Fermat's Liar*, peneliti menggunakan metode pengujian berulang dengan basis berbeda. Peneliti melakukan uji Fermat sebanyak jumlah digit p dikali 3 untuk mengurangi kemungkinan kesalahan. Semakin banyak basis yang diuji, semakin kecil kemungkinan bilangan komposit lolos seluruh tes. Hal ini meningkatkan keandalan

dalam mengidentifikasi bilangan prima sejati dan mengurangi risiko *false positive* dalam penerapan teorema Fermat.

Contoh: Apakah 13 adalah bilangan prima?

$$p = 13 \Rightarrow p - 1 = 12$$

$$1 < a < 13$$

Tabel 2.1 Perhitungan *Fermat's Little Theorem*

- | | |
|---|---|
| • $a = 2 \Rightarrow F = 2^{12} \bmod 13 = 1$ | • $a = 9 \Rightarrow F = 9^{12} \bmod 13 = 1$ |
| • $a = 3 \Rightarrow F = 3^{12} \bmod 13 = 1$ | • $a = 10 \Rightarrow F = 10^{12} \bmod 13 = 1$ |
| • $a = 4 \Rightarrow F = 4^{12} \bmod 13 = 1$ | • $a = 11 \Rightarrow F = 11^{12} \bmod 13 = 1$ |
| • $a = 5 \Rightarrow F = 5^{12} \bmod 13 = 1$ | • $a = 12 \Rightarrow F = 12^{12} \bmod 13 = 1$ |
| • $a = 6 \Rightarrow F = 6^{12} \bmod 13 = 1$ | • $a = 13 \Rightarrow F = 13^{12} \bmod 13 = 1$ |
| • $a = 7 \Rightarrow F = 7^{12} \bmod 13 = 1$ | • $a = 14 \Rightarrow F = 14^{12} \bmod 13 = 1$ |
| • $a = 8 \Rightarrow F = 8^{12} \bmod 13 = 1$ | |

Dari Tabel 2.1 dapat disimpulkan bahwa 13 adalah sebuah bilangan prima.

2.3.4. Operasi modulo

Disebut $m \bmod n = r$, dikarenakan ada k sedemikian rupa sehingga $k \cdot n + r = m$.

Contoh: Disebut $27 \bmod 5 = 2$, karena ada $k = 2$ sehingga $5 \cdot 2 + 2 = 27$.

2.3.5. Invers modulo

Invers dari $m \pmod{n}$ disebut m^{-1} apabila $m^{-1} \cdot m \equiv 1 \pmod{n}$

Contoh: Berapa invers dari 3 ($\bmod 17$)?

$$m = 3; n = 17$$

Tabel 2.2 Perhitungan invers dari $m \pmod{n}$

m^{-1}	$m^{-1} \cdot m \pmod{n}$ $m^{-1} \cdot 3 \pmod{17}$
1	3
2	6
3	9
4	12
5	15

6	1
---	---

$$m^{-1} = 6$$

$$m^{-1} = 6 \pmod{17}$$

Dari tabel 2.2 dapat dilihat bahwa hasil invers dari 3 (mod 17) adalah 6.

2.3.6. Greatest Common Divisor (GCD)

Greatest Common Divisor (GCD) atau faktor Persekutuan terbesar (FPB) adalah bilangan bulat positif terbesar yang dapat membagi dua atau lebih bilangan tanpa bersisa.

Contoh *Euclidean GCD*:

$$\text{GCD}(54, 14) = ?$$

$$54 \bmod 14 = 12$$

$$14 \bmod 12 = 2$$

$$12 \bmod 2 = 0$$

Maka GCD (54, 14) adalah 2.

2.3.7. Algoritma Square and Multiply (SAM)

Algoritma *Square and Multiply* (SAM) adalah metode efisien yang digunakan untuk menghitung modulo ekponensial. Algoritma ini sangat berguna dalam menyelesaikan komputasi eksponensial besar dalam kriptografi, karena mengurangi jumlah operasi yang diperlukan dibandingkan dengan metode langsung. *Recall* modulo eksponensial adalah *term* matematika diskrit yang berbentuk $x^y \bmod k$.

Langkah – langkah algoritma *Square and Multiply*:

1. Ubah nilai y ke bentuk biner
2. Atur nilai $z = 1$
3. Lakukan iterasi dengan menelusuri nilai bit y dari kiri ke kanan
 - Bila bertemu 0, maka perbaharui nilai z menjadi: $z = z^2 \bmod k$
 - Bila bertemu 1, maka perbaharui nilai z menjadi: $z = xz^2 \bmod k$
4. Setelah semua bit diproses, nilai z terakhir adalah nilai dari $x^y \bmod k$ yang dicari.

Contoh: $136^{173} \bmod 247 = \dots?$

$$x = 136$$

$$y = 173 = 10101101_2$$

$$k = 247$$

$$z = 1$$

Iterasi:

- $1 \rightarrow z = xz^2 \bmod k = 136 \cdot 1^2 \bmod 247 = 136$
- $0 \rightarrow z = z^2 \bmod k = 136^2 \bmod 247 = 218$
- $1 \rightarrow z = xz^2 \bmod k = 136 \cdot 218^2 \bmod 247 = 15$
- $0 \rightarrow z = z^2 \bmod k = 15^2 \bmod 247 = 225$
- $1 \rightarrow z = xz^2 \bmod k = 136 \cdot 225^2 \bmod 247 = 122$
- $1 \rightarrow z = xz^2 \bmod k = 136 \cdot 122^2 \bmod 247 = 59$
- $0 \rightarrow z = z^2 \bmod k = 59^2 \bmod 247 = 23$
- $1 \rightarrow z = xz^2 \bmod k = 136 \cdot 23^2 \bmod 247 = 67$

Maka hasil dari $136^{173} \bmod 247$ adalah 67.

2.4. Algoritma RSA

Rivest Shamir Adleman (RSA) adalah salah satu algoritma kriptografi asimetris atau berbasis kunci-publik, yang menggunakan sepasang kunci berbeda, yaitu kunci privat dan kunci public (Muchlis, 2015). Dinamai berdasarkan penemunya, Ron Rivest, Adi Shamir, dan Len Adleman, algoritma ini dikembangkan pada tahun 1976. Keunggulan RSA tidak hanya bergantung pada panjang kunci (semakin panjang kunci, semakin lama waktu prosesnya) atau pada penggunaan umum kunci privat dan publik. Kekuatan utama RSA terletak pada kesulitan untuk memfaktorkan bilangan besar ke dalam dua bilangan primanya, p dan q , sehingga memenuhi persamaan $n = p \cdot q$.

RSA terdiri dari tiga tahapan utama, yaitu *key generation*, *encryption*, dan *decryption*.

- **Key Generation**

1. Bangkitkan dua buah bilangan prima yang sangat besar, yaitu p dan q .

$$p = 13, q = 19$$

2. Hitung $n = p \cdot q$

$$n = 13 \cdot 19 = 247$$

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

$$\phi(n) = (13 - 1) (19 - 1)$$

$$\phi(n) = 12 \cdot 18 = 216$$

4. Bangkitkan nilai e , dengan syarat:

- $1 < e < \phi(n)$
- e harus bilangan ganjil
- $\text{GCD}(e, \phi(n)) = 1$

Test $e = 5$

$\text{GCD}(5, 216) = 1$ (memenuhi)

$216 \bmod 5 = 1$ (menggunakan *Euclidean GCD*)

$5 \bmod 1 = 0$

5. Hitung $d \equiv e^{-1} \pmod{\phi(n)}$

$d \equiv 5^{-1} \pmod{216}$

$5x + 216y = 1$ (ubah ke persamaan *Diophantine Linear*)

Tabel 2.3 Perhitungan d menggunakan Algoritma *Extended Euclidean*

x	y	r	k
0	1	216	
1	0	5	43
-43	1	1	

$x = -43$

$x = -43 \pmod{216}$

$x = -43 + 216 = 173$

$d = 173$

Tabel 2.3 menampilkan tahapan perhitungan algoritma *Extended Euclidean* untuk menentukan nilai d (kunci privat) dengan menunjukkan proses iterasi perhitungan menggunakan variabel x , y , r , dan k .

6. Rahasiakan nilai p , q , $\phi(n)$, dan d sebagai *private key*.

Private key: $(p, q, \phi(n), d) = (13, 19, 216, 173)$

7. *Publish* nilai e dan n sebagai *public key*.

Public key: $(e, n) = (5, 247)$

- **Encryption**

1. Sepakati tabel *encoding* yang akan digunakan

Tabel *encoding* yang digunakan adalah tabel ASCII.

2. Dapatkan *public key* milik *recipient*

Public key: $(e, n) = (5, 247)$

3. *Encode* pesan yang akan dikirim dengan melihat tabel *encoding* sebelumnya
 $m = \text{"CHY"}$

$$C = 67, H = 72, Y = 89$$

4. Enkripsi pesan m dengan rumus:

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

$$c_c = 67^5 \bmod 247 = 1350125107 \bmod 247 = 136$$

$$c_h = 72^5 \bmod 247 = 1934917632 \bmod 247 = 154$$

$$c_y = 89^5 \bmod 247 = 5584059449 \bmod 247 = 33$$

5. Kirim nilai c dalam bentuk angka kepada *recipient*.

$$c = [136, 154, 33]$$

- **Decryption**

1. Terima nilai c dari *sender*

$$c = [136, 154, 33]$$

2. Dekripsi c menjadi m dengan rumus:

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

$$m_1 = 136^{173} \bmod 247 = 67$$

$$m_2 = 154^{173} \bmod 247 = 72$$

$$m_3 = 33^{173} \bmod 247 = 89$$

Hasil pesan asli $m : [67, 72, 89] = \text{"CHY"}$

RSA sangat populer dan sering digunakan karena memiliki beberapa keunggulan, di antaranya:

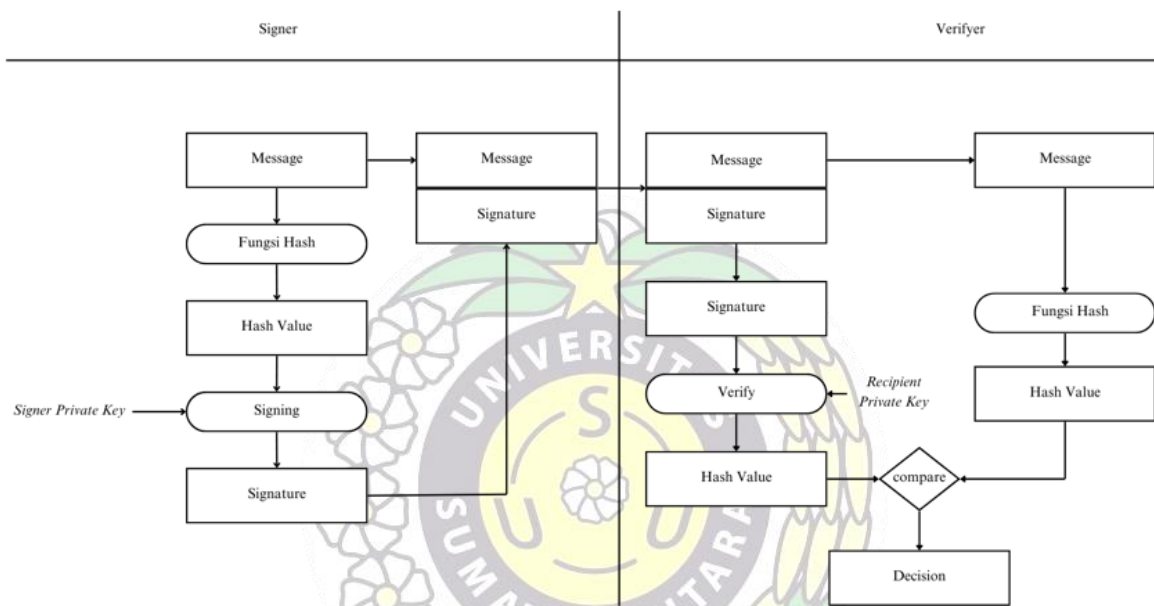
- Tingkat keamanan yang tinggi: RSA memanfaatkan teknologi kriptografi kunci publik yang dianggap sangat sulit untuk diretas.
- Mudah diimplementasikan: RSA dapat diterapkan dengan mudah pada berbagai platform dan sistem operasi.
- Performa yang memadai: Meskipun proses pembuatan kunci membutuhkan waktu, RSA memiliki kecepatan yang cukup baik dalam menjalankan operasi kriptografi.

2.5. Digital Signature

Digital signature merupakan skema kriptografi yang bergantung pada pesan dan identitas pengirimnya. Tanda tangan ini berfungsi untuk menjamin integritas data, mendeteksi adanya perubahan atau manipulasi pada pesan yang dikirim, serta

memverifikasi keaslian pesan. Proses pembuatan tanda tangan digital pada pesan biasanya dilakukan dengan memanfaatkan fungsi *hash* (Ginting et al., 2024).

Gambar 2.4 menggambarkan skema digital signature yang terbagi menjadi dua sisi (*Signer* dan *Verifier*), menunjukkan alur proses penandatanganan digital mulai dari persiapan pesan, pembangkitan kunci, proses *signing*, hingga tahap verifikasi tanda tangan dengan menggunakan *hash value* dan *signature* sebagai komponen kunci dalam memvalidasi keaslian dan integritas pesan.



Gambar 2.4 Skema *Digital Signature*

Keamanan skema tanda tangan digital memiliki beberapa aspek penting, seperti kerahasiaan (*confidentiality*), yang menjamin bahwa informasi hanya dapat diakses oleh pihak yang berwenang; integritas (*integrity*), yang memastikan bahwa informasi tidak diubah tanpa izin; dan otentikasi (*authentication*), yang memverifikasi identitas pengirim pesan (Muchlis, 2015). Skema tanda tangan digital yang diusulkan adalah skema IKZA, yang bertujuan untuk memenuhi aspek-aspek keamanan tersebut.

2.6. *Digital Signature Inam – Kanwal – Zahid – Abid (IKZA)*

Pada tahun 2020, S. Inam, S. Kanwal, A. Zahid, dan M. Abid mengembangkan algoritma tanda tangan digital yang dikenal sebagai *Digital Signature Inam-Kanwal-Zahid-Abid (IKZA)*. *Digital signature* IKZA dirancang sebagai tanda tangan digital yang menggunakan operasi berbasis matriks pada cincin *non-komutatif* (Inam et al.,

2020). Pendekatan ini memberikan keamanan tambahan dengan memanfaatkan sifat kompleksitas aljabar dari struktur matriks dan polinomial. Selain itu, algoritma ini dirancang untuk menghasilkan tanda tangan yang aman dengan efisiensi perhitungan tinggi, berkat penggunaan panjang kunci yang lebih pendek dibandingkan algoritma konvensional.

Fungsi hash yang digunakan pada tahap *signature generation* adalah metode *hashing* berbasis aritmetika dengan memanfaatkan kombinasi linear elemen-elemen matriks dan determinan sebagai faktor tambahan. *Hash* ini lebih ringan dibandingkan metode kriptografi umum seperti SHA-256, karena hanya menggunakan operasi dasar matematika dan modulus. Meskipun sederhana, metode ini tetap menjaga keunikan *hash* untuk setiap matriks yang diberikan.

Dalam pengimplentasiannya *digital signature* ini memiliki beberapa tahap, yaitu:

- **Initialization**

1. Hitung $N = p \cdot q$, dimana p dan q merupakan dua bilangan prima besar secara acak.
2. Buat matriks A dan B dengan syarat, yaitu $A, B \in M_2(Z_N)$
3. Buat fungsi polinomial $f(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$, dimana koefisien merupakan bilangan bulat positif
4. Hitung $P_1 = f(A) \bmod N$ dan $Q_1 = f(B) \bmod N$

- **Key Generation**

1. Buat fungsi polinomial $g(x) = b_0 + b_1x + b_2x^2 + \dots + b_nx^n$, dimana $g(x) \in Z_N[x]$. Lalu hitung $g(P_1) = N_1 \neq 0$.
2. Hitung $Y = (N_1^{m_1} Q_1 N_1^{m_2}) \bmod N$, dimana m_1 dan m_2 adalah bilangan bulat.
3. Kunci privat adalah N_1 , sedangkan kunci publik adalah $(P_1, Q_1, Y) \in M_2(Z_N)$

- **Signature Generation**

1. Buat fungsi polinomial $h(x) = c_0 + c_1x + c_2x^2 + \dots + c_nx^n$, dimana $h(x) \in Z_N[x]$. Lalu hitung $N_2 = h(P_1)$.
2. Untuk pesan M , hitung nilai *hash* $H(M)$ dan elemen sebagai berikut:
 - $\alpha = (N_2^{m_1} Q^{m_1} N_2^{m_2}) \bmod N$,
 - $\beta = (N_1^{m_1} \{H(M)\alpha\} N_1^{m_2}) \bmod N$
 - $\gamma = (N_2^{m_1} \beta N_2^{m_2}) \bmod N$

- $\varphi = (N_2^{m1} \beta N_1^{m2}) \bmod N$
- $\psi = (N_1^{m1} H(M) N_2^{m2}) \bmod N$
- $\eta = (N_2^{m1} H(M) N_2^{m2}) \bmod N$

3. Kirim tanda tangan $(\alpha, \gamma, \varphi, \psi, \eta)$ untuk proses verifikasi

- **Verification**

1. Terima tanda tangan $(\alpha, \gamma, \varphi, \psi, \eta)$
2. Hitung $V = \varphi Y^{-1} \psi \bmod N$
3. Tanda tangan diterima jika dan hanya jika:

$$(\alpha^{-1} \eta) \bmod N = (\gamma^{-1} V) \bmod N$$

4. Jika $H(M)$ tidak memiliki invers multiplikatif di Z_N , maka validasi dilakukan dengan:

$$(\gamma \alpha^{-1}) \bmod N = V \bmod N$$

5. Jika syarat verifikasi terpenuhi, maka tanda tangan dianggap valid.

2.7. Teknik *Signcryption* dengan Algoritma RSA dan *Digital Signature* Inam – Kanwal – Zahid – Abid (IKZA)

Penelitian ini menggunakan metode *encrypt-then-sign*. Dalam proses ini, pesan (*plaintext*) dienkripsi menggunakan algoritma RSA untuk menghasilkan *ciphertext*. Selanjutnya, *ciphertext* ditandatangani menggunakan algoritma IKZA. Metode ini menggabungkan kriptografi kunci publik dan tanda tangan digital guna meningkatkan tingkat keamanan. Proses ini dikenal sebagai autentikasi dan enkripsi.

2.8. Penelitian Relevan

Dibawah ini merupakan sejumlah riset yang berhubungan terkait Algoritma RSA dengan *Digital Signature* Inam-Kanwal-Zahid-Abid (IKZA).

1. Dalam penelitian berjudul “*Signcryption with Matrix Modification of RSA Digital Signature Scheme and Cayley-Purser Algorithm*” yang dilakukan oleh (Ginting et al., 2024), membahas modifikasi matriks pada skema tanda tangan digital RSA dapat meningkatkan keamanan melalui representasi pesan dalam bentuk matriks persegi dengan ordo tertentu. Penelitian ini menunjukkan bahwa metode ini tetap menggunakan modulus n , yang merupakan hasil perkalian dari dua buah bilangan prima besar p dan q , sehingga memberikan tingkat keamanan yang tinggi.

2. Dalam penelitian berjudul “*A Novel Public Key Cryptosystem and Digital Signatures*” yang dilakukan oleh (Inam et al., 2020), membahas tentang *Digital Signature Inam-Kanwal-Zahid-Abid* (IKZA), yaitu tanda tangan digital berbasis cincin non-komutatif, yang memanfaatkan sifat kompleksitas aljabar dari struktur matriks. Keunggulan utama algoritma ini terletak pada efisiensinya dalam menghasilkan tanda tangan dengan panjang kunci yang lebih pendek, namun tetap menjaga tingkat keamanan yang tinggi.
3. Dalam penelitian berjudul “*Sign-Then-Encrypt Scheme with Cramer-Shoup Cryptosystem and Dissanayake Digital Signature*” yang dilakukan oleh (Bahri et al., 2023), disimpulkan bahwa skema *signcryption* dapat menjaga keamanan data dengan menggabungkan autentikasi melalui tanda tangan digital dan kerahasiaan melalui enkripsi.
4. Dalam penelitian berjudul “*Optimasi Pemrosesan Enkripsi dan Dekripsi RSA pada Single Board Computer (SBC) dengan Pembagian Beban Komputasi dalam Sistem Terdistribusi*” yang dilakukan oleh (Prasetyo et al., 2021), disimpulkan bahwa algoritma RSA menunjukkan efisiensi tinggi ketika diimplementasikan pada arsitektur cluster SBC. Penelitian ini menunjukkan bahwa algoritma RSA tetap aman karena dasar keamanannya yang sulit diretas, selama bilangan prima besar dan kunci enkripsi yang digunakan cukup besar, misalnya 2048 bit. Hal ini menunjukkan potensi besar RSA untuk digunakan dalam skenario keamanan data berskala besar dengan dukungan teknologi terdistribusi.

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1. Analisis Sistem

Tahapan analisis sistem bertujuan untuk mengidentifikasi kebutuhan agar sistem dapat bekerja secara optimal sesuai harapan. Analisis sistem ini mencakup tiga fase utama, yaitu analisis masalah, analisis kebutuhan, dan analisis proses. Analisis masalah dilakukan untuk mengidentifikasi isu-isu yang muncul. Analisis kebutuhan digunakan untuk menggambarkan fungsi-fungsi yang harus disediakan oleh sistem. Tujuan dari analisis proses adalah untuk memodelkan perilaku sistem agar sesuai dengan kebutuhan pengguna.

3.1.1. Analisis masalah

Tahapan analisis masalah dilakukan untuk mengidentifikasi potensi kendala yang dapat muncul dalam sistem *signcryption* berbasis RSA dan *digital signature* Inam-Kanwal-Zahid-Abid (IKZA). Tujuannya adalah memastikan bahwa sistem mampu memenuhi kebutuhan pengguna dan memberikan performa yang optimal.

Analisis ini menggunakan pendekatan SWOT untuk mengidentifikasi potensi dan kendala dalam pengembangan sistem *signcryption* berbasis RSA dan tanda tangan digital IKZA. Pendekatan SWOT dipilih karena metode ini memungkinkan peneliti untuk mengidentifikasi secara menyeluruh faktor internal dan eksternal yang berpengaruh pada sistem.

Strengths

- Keamanan Tinggi:

RSA dikenal memiliki tingkat keamanan tinggi karena didasarkan pada kompleksitas faktorisasi bilangan prima besar. Skema tanda tangan digital IKZA memiliki struktur matematika yang kuat untuk memastikan autentikasi dan

integritas pesan. Kombinasi enkripsi dan tanda tangan digital memberikan dua lapisan perlindungan terhadap ancaman manipulasi dan peretasan.

- Efisiensi dalam Proses:

Teknik *signcryption* menggabungkan proses enkripsi dan tanda tangan digital menjadi satu langkah, yang secara signifikan mengurangi *overhead* komputasi dibandingkan dengan menjalankan keduanya secara terpisah. RSA memiliki fleksibilitas untuk digunakan dalam berbagai skenario komunikasi yang membutuhkan autentikasi dan enkripsi.

- Penerimaan yang Luas:

RSA adalah algoritma kriptografi yang sudah dikenal luas, sehingga memudahkan penerimaan dan implementasi dalam sistem keamanan yang sudah ada.

Weaknesses

- Kompleksitas Komputasi:

RSA membutuhkan operasi komputasi yang berat, terutama pada kunci dengan ukuran besar, yang dapat memengaruhi performa pada perangkat dengan sumber daya terbatas. Implementasi tanda tangan digital IKZA memerlukan penyesuaian khusus untuk kompatibilitas dengan algoritma RSA.

- Ketergantungan pada Panjang Kunci:

Keamanan RSA sangat bergantung pada panjang kunci. Kunci pendek rentan terhadap serangan *brute-force* atau serangan lainnya. Tanda tangan digital berbasis matriks (IKZA) memerlukan pengelolaan parameter yang kompleks untuk mencegah cacat implementasi.

- Kurangnya Studi Implementasi:

Kombinasi antara RSA dan tanda tangan digital IKZA belum ada dieksplorasi, sehingga mungkin memerlukan waktu lebih lama untuk pengembangan dan pengujian.

Opportunities

- Meningkatnya Permintaan Keamanan Data:

Ancaman terhadap privasi dan integritas data semakin meningkat, sehingga menciptakan kebutuhan akan solusi keamanan yang lebih baik, seperti *signcryption*. Banyak sektor (seperti perbankan, *e-commerce*, dan layanan

kesehatan) membutuhkan perlindungan data tingkat tinggi, yang dapat dimanfaatkan untuk aplikasi solusi ini.

- **Perkembangan Regulasi Keamanan Data:**

Adanya regulasi seperti GDPR atau UU Perlindungan Data semakin mendorong pengadopsian teknologi kriptografi untuk perlindungan data pribadi.

- **Inovasi dalam Teknologi *Signcryption*:**

Kombinasi RSA dan IKZA menawarkan efisiensi dan keamanan yang lebih baik dibandingkan metode konvensional, menjadikannya inovasi yang relevan di bidang keamanan digital.

Threats

- **Perkembangan Komputer Kuantum:**

Algoritma RSA rentan terhadap ancaman dari komputer kuantum di masa depan, yang dapat memecahkan faktorisasi bilangan prima dengan cepat. Jika solusi berbasis kuantum mulai diadopsi, algoritma RSA mungkin akan digantikan oleh metode kriptografi baru.

- **Serangan yang Lebih Canggih:**

Serangan siber, seperti side-channel attacks, terus berkembang, sehingga memerlukan penyesuaian algoritma secara berkala. Skema tanda tangan digital IKZA masih perlu diuji lebih lanjut untuk memastikan ketahanan terhadap serangan modern.

- **Kompetisi dengan Algoritma Lain:**

Algoritma seperti ECC (*Elliptic Curve Cryptography*) menawarkan keamanan serupa dengan efisiensi komputasi yang lebih tinggi, yang dapat menjadi pesaing utama RSA.

3.1.2. Analisis kebutuhan

Analisis kebutuhan mencakup identifikasi data-data yang diperlukan dan alur sistem yang dirancang untuk mencapai tujuan penelitian. Dalam penelitian ini, kebutuhan sistem dikelompokkan menjadi dua kategori, yaitu:

1. **Kebutuhan Fungsional**

Kebutuhan fungsional adalah proses yang harus dilakukan sistem. Adapun kebutuhan fungsional yang diperlukan adalah:

- a. Sistem mampu melakukan pembangkitan kunci publik dan kunci privat untuk RSA.
 - b. Sistem mampu mengenkripsi pesan menjadi cipherteks menggunakan algoritma RSA.
 - c. Sistem mampu membuat tanda tangan digital menggunakan *Digital Signature* Inam – Kanwal – Zahid – Abid.
 - d. Sistem mampu menjalankan proses *signcryption* yang mengintegrasikan algoritma RSA dan *Digital Signature* Inam – Kanwal – Zahid – Abid.
 - e. Sistem mampu melakukan verifikasi tanda tangan digital dan dekripsi pesan.
2. Kebutuhan *Non-Fungsional*

Kebutuhan *non-fungsional* mencakup batasan pengembangan dan kemampuan sistem. Adapun kebutuhan *non-fungsional* yang diperlukan adalah:

- a. Menampilkan antarmuka pengguna yang intuitif dan mudah digunakan, serta mampu menampilkan hasil proses *signcryption*. Setelah mengeksplor sistem pengguna dapat memahami sistem dalam waktu kurang lebih sepuluh menit.
- b. Menampilkan pesan kesalahan jika pengguna memasukkan *input* yang tidak valid atau tidak lengkap.
- c. Ketahanan sistem terhadap jenis serangan kriptografi seperti *brute-force*.

3.1.3. Analisis proses

Analisis proses menjelaskan bagaimana sistem menjalankan fungsinya. Berikut adalah alur proses utama:

- Pembangkitan kunci: Sistem menghasilkan pasangan kunci publik dan privat RSA untuk pengguna.
- Enkripsi pesan: Pesan pengguna dienkripsi menggunakan kunci publik RSA.
- Pembuatan tanda tangan digital: Sistem menghasilkan tanda tangan digital menggunakan algoritma IKZA dengan input berupa pesan terenkripsi.
- Proses *signcryption*: Sistem mengintegrasikan hasil enkripsi dan tanda tangan digital dalam satu langkah proses.

- Verifikasi dan dekripsi: Sistem memverifikasi tanda tangan digital untuk memvalidasi integritas dan keaslian pesan kemudian mendekripsi pesan menggunakan kunci privat RSA.
- Output sistem: Sistem menampilkan pesan asli dan status validasi tanda tangan digital kepada pengguna. Jika terjadi kesalahan, sistem memberikan pesan error.

3.2. Pemodelan Sistem

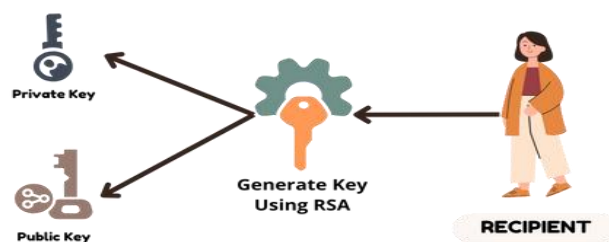
Pemodelan sistem adalah representasi visual dari sistem yang digunakan untuk memahami, menganalisis, dan merancang sistem tersebut sebelum proses implementasi dilakukan. Pemodelan ini bertujuan untuk memberikan gambaran yang lebih jelas mengenai bagaimana sistem bekerja, alur informasi, serta interaksi antar komponen dalam sistem. Dalam pemodelan sistem, peneliti memanfaatkan diagram umum dan diagram aktivitas untuk merepresentasikan alur serta struktur sistem yang dirancang.

3.2.1. Diagram umum sistem

Diagram umum sistem merupakan gambaran alur, proses, dan hubungan antar komponen dalam sistem *signcryption*. Pada penelitian ini, diagram umum sistem dibagi menjadi dua bagian, yaitu diagram umum pembangkitan kunci dan diagram umum *signcryption*.

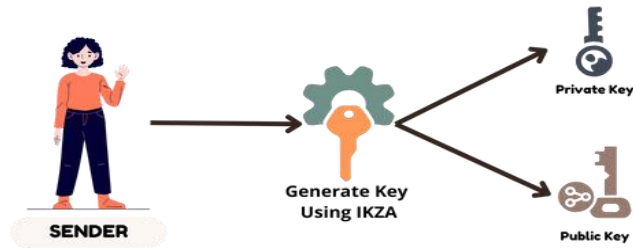
3.2.1.1 Diagram Umum Pembangkitan Kunci

Pada Gambar 3.1 ditunjukkan proses pembangkitan kunci yang digunakan untuk proses enkripsi dan dekripsi oleh penerima. Penerima akan membangkitkan sepasang kunci publik (e, n) dan kunci privat (d).



Gambar 3.1 Diagram Umum Key Generation RSA

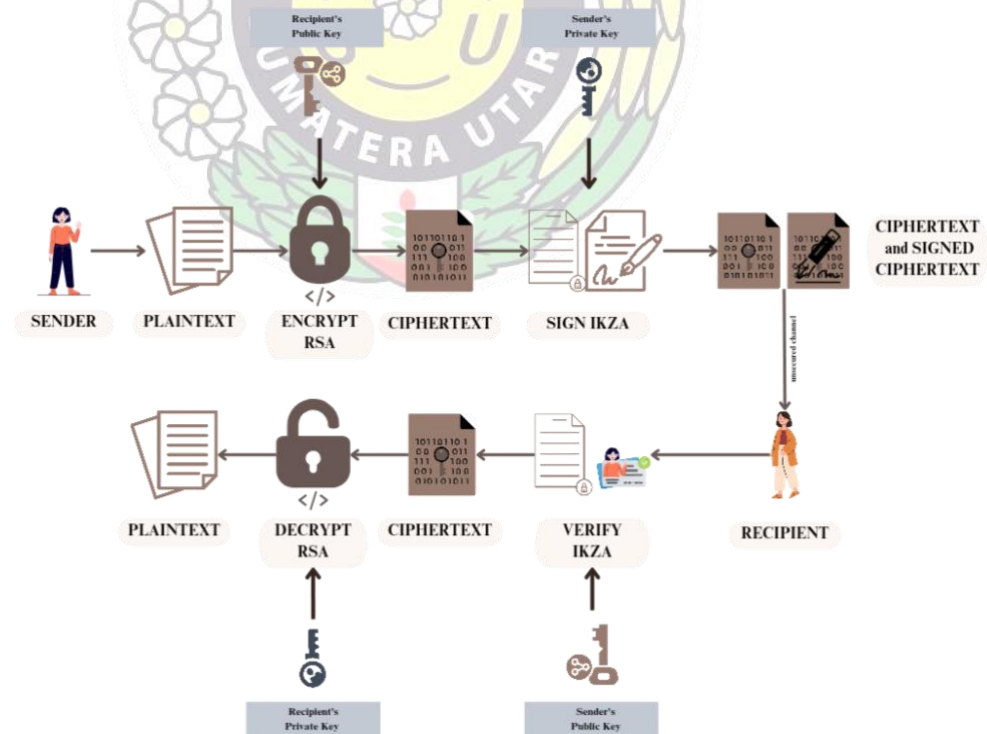
Selain itu, pada Gambar 3.2 pengirim juga akan membangkitkan kunci tanda tangan digital yang terdiri dari kunci privat (N_1) untuk proses penandatanganan dan kunci publik ($(P_1, Q_1, Y) \in M_2(Z_N)$) untuk proses verifikasi.



Gambar 3.2 Diagram Umum Key Generation IKZA

3.2.1.2 Diagram Umum Signcryption

Pada Gambar 3.3 menunjukkan proses *signcryption* yang dilakukan untuk mengamankan pesan antara pengirim dan penerima. Pertama-tama pengirim mengenkripsi pesan menggunakan algoritma RSA dengan kunci publik penerima, menghasilkan *ciphertext* yang aman. Setelah itu, pengirim menandatangani *ciphertext* menggunakan skema tanda tangan digital IKZA dengan kunci pribadinya, menghasilkan *signed ciphertext*. Setelah itu *ciphertext* dan *signed ciphertext* dikirim ke penerima melalui *unsecured channel*. Penerima akan memverifikasi *signed ciphertext* tersebut dengan menggunakan kunci publik pengirim untuk memastikan keaslian pesan.



Gambar 3.3 Diagram Umum Signcryption

Jika verifikasi berhasil atau valid, penerima kemudian melakukan dekripsi pada *ciphertext* menggunakan kunci privatnya untuk mengambil pesan asli yang dikirim oleh pengirim. Dengan demikian, penerima mendapatkan pesan yang telah diverifikasi keasliannya namun tetap terjaga kerahasiaannya.

3.2.2. Use case diagram

Use case diagram adalah diagram yang memvisualisasikan interaksi antara sistem dan aktor yang terlibat di dalamnya. Diagram ini menunjukkan bagaimana pengguna berinteraksi dengan sistem serta jenis hubungan yang terbentuk dalam proses tersebut.

Pada Gambar 3.4 terdapat dua aktor yang berinteraksi dengan sistem, yaitu pengirim dan penerima. Diagram *use case* yang telah dibuat menunjukkan sistem ini dimulai dengan proses membangkitkan bilangan prima secara acak, yang digunakan untuk menghasilkan kunci bagi algoritma RSA dan IKZA. Pengirim kemudian mengenkripsi pesan menggunakan algoritma RSA agar tidak dapat dibaca oleh pihak yang tidak berwenang. Setelah itu, pengirim menggunakan kunci IKZA untuk melakukan tanda tangan pesan, yang berfungsi untuk menjamin integritas dan autentikasi pesan.



Gambar 3.4 Use Case Diagram

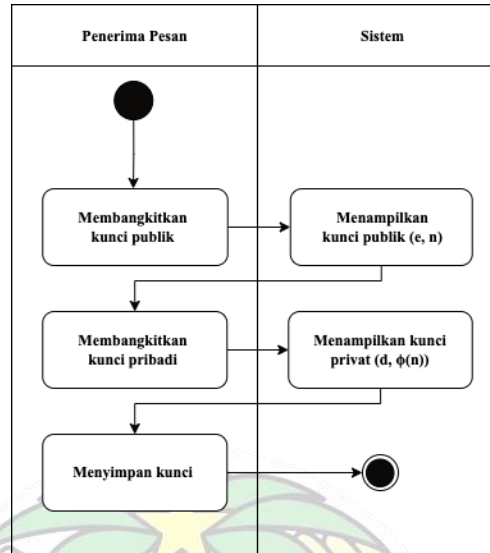
Di sisi lain penerima melakukan verifikasi tanda tangan digital dengan IKZA untuk memastikan bahwa pesan berasal dari pengirim yang sah dan tidak mengalami perubahan selama pengiriman. Jika valid, pesan yang telah dienkripsi akan melalui proses dekripsi dengan RSA untuk mendapatkan isi pesan aslinya.

3.2.3. Activity diagram

Activity diagram adalah salah satu jenis diagram yang digunakan untuk memodelkan alur kerja (*workflow*) atau proses dalam suatu sistem. Diagram ini

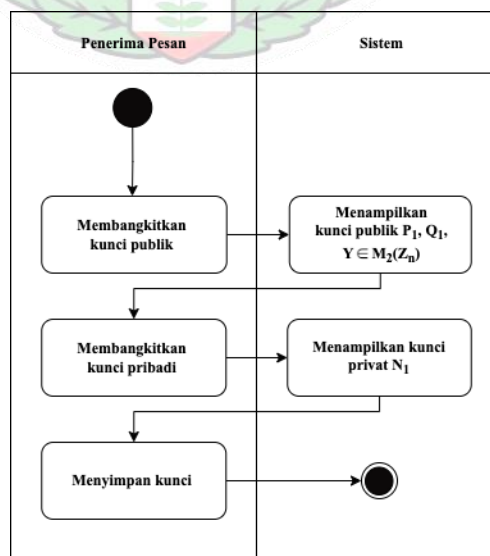
menggambarkan bagaimana aktivitas atau proses berjalan dari awal hingga akhir, termasuk keputusan dan kondisi yang mempengaruhi jalannya proses.

3.2.3.1 Activity Diagram Key Generation



Gambar 3.5 Activity Diagram Key Generation RSA Penerima

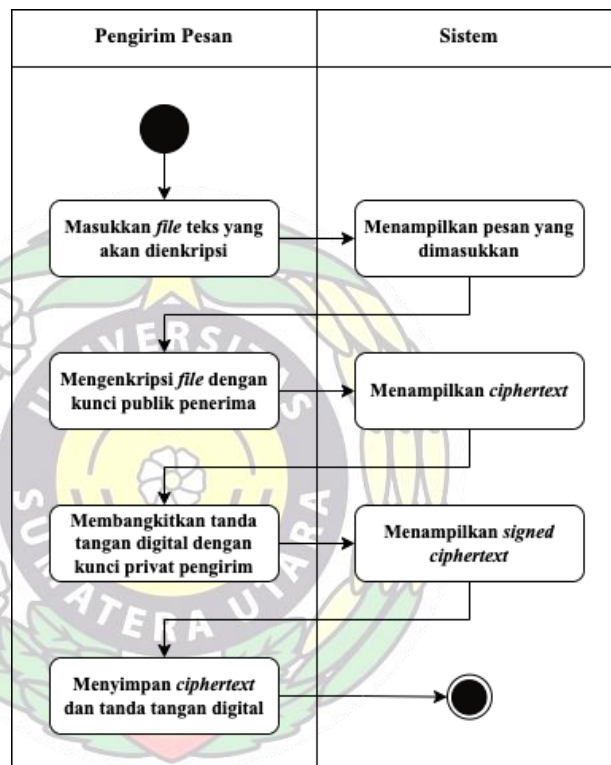
Pada Gambar 3.5 menunjukkan tahapan dalam proses pembangkitan kunci RSA yang digunakan untuk enkripsi dan dekripsi. Proses dimulai dengan membangkitkan kunci publik dan pribadi. Kunci publik terdiri dari dua parameter utama (e, n), sedangkan kunci privat terdiri dari ($d, \phi(n)$). Setelah kedua kunci dibuat, proses diakhiri dengan menyimpannya agar dapat digunakan dalam proses enkripsi dan dekripsi selanjutnya.



Gambar 3.6 Activity Diagram Key Generation IKZA Pengirim

Pada Gambar 3.6 menunjukkan tahapan dalam proses pembangkitan kunci IKZA yang digunakan untuk tanda tangan dan verifikasi. Proses dimulai dengan membangkitkan kunci publik dan pribadi. Kunci publik terdiri dari tiga parameter utama $((P_1, Q_1, Y) \in M_2(Z_N))$, sedangkan kunci privat adalah N_1 . Setelah kedua kunci dibuat, proses diakhiri dengan menyimpannya agar dapat digunakan dalam proses tanda tangan dan verifikasi.

3.2.3.2 Activity Diagram Encrypt dan Sign

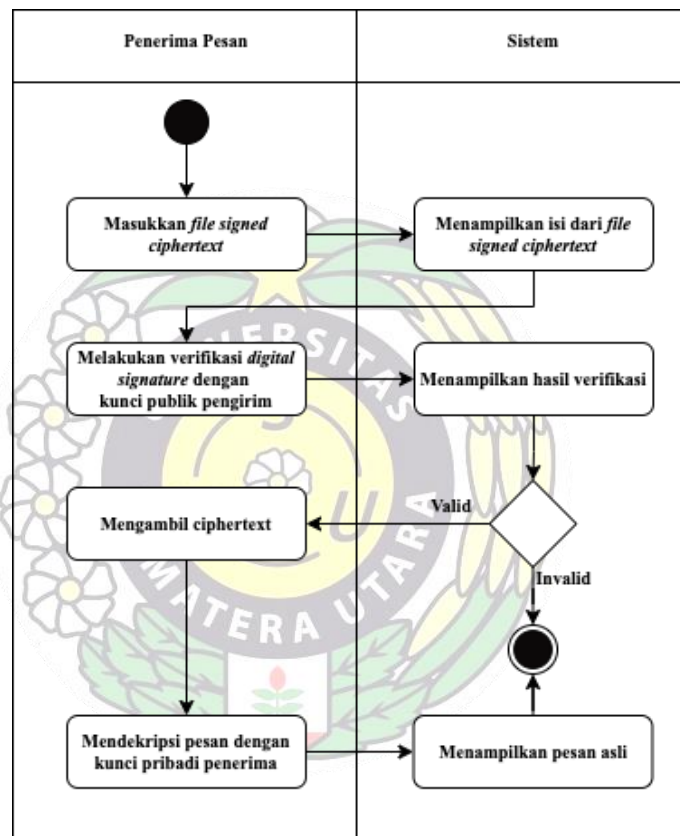


Gambar 3.7 Activity Diagram Encrypt dan Sign

Pada Gambar 3.7 diperlihatkan proses enkripsi pesan dan tanda tangan digital sebelum dikirimkan. Proses dimulai dengan memasukkan *file* teks yang akan dienkripsi. Selanjutnya, pesan dalam bentuk *plaintext* dienkripsi menggunakan kunci publik penerima. Setelah itu, tanda tangan digital dibuat menggunakan kunci privat pengirim. Hasil akhir dari proses ini adalah *signed ciphertext* yang kemudian disimpan dan siap untuk dikirimkan ke penerima.

3.2.3.3 Activity Diagram Verify dan Decrypt

Pada Gambar 3.8 menggambarkan proses verifikasi tanda tangan digital dan dekripsi pesan dalam sistem keamanan data. Alur dimulai dengan memasukkan file *signed ciphertext* (pesan tersandi dan tertandatangan) dan menampilkannya. Kemudian, tanda tangan digital diverifikasi menggunakan kunci publik pengirim. Jika verifikasi berhasil, maka pesan didekripsi menggunakan kunci privat penerima. Setelah itu, pesan asli ditampilkan sebagai *output* akhir



Gambar 3.8 Activity Diagram Verify dan Decrypt

3.3. Flowchart (Diagram Alir)

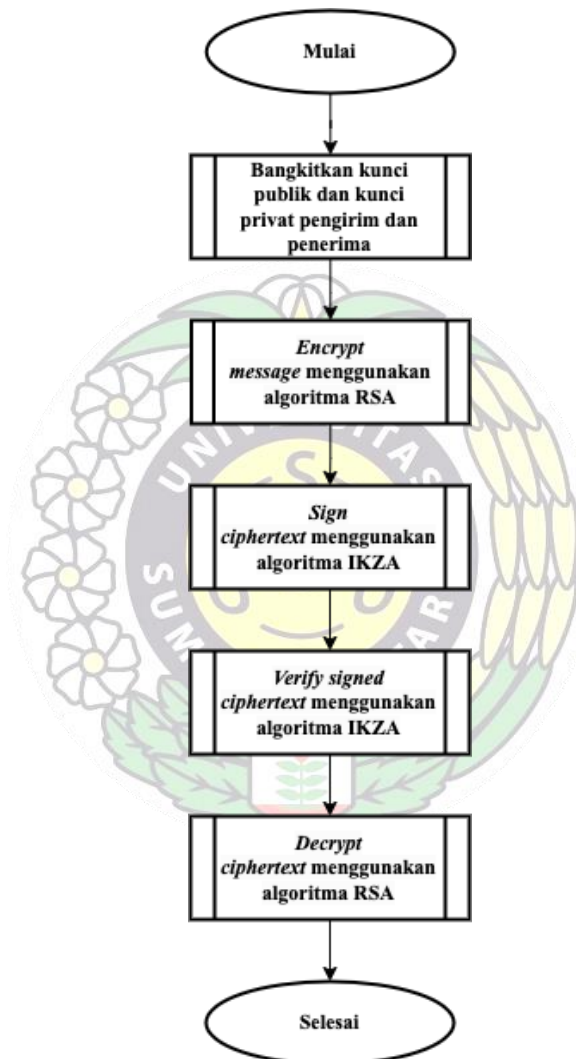
Flowchart adalah representasi grafis dari suatu proses atau algoritma yang menggambarkan langkah-langkah yang harus dilakukan dalam menyelesaikan suatu masalah. *Flowchart* menggunakan simbol-simbol standar untuk merepresentasikan berbagai jenis operasi dan alur proses dalam suatu sistem.

Flowchart sering digunakan dalam perancangan sistem untuk membantu pemahaman terhadap logika dan urutan proses sebelum sistem diimplementasikan dalam bentuk kode atau aplikasi. Dengan menggunakan representasi grafis, *flowchart*

mempermudah analisis dan identifikasi alur kerja, sehingga sistem dapat dirancang dengan lebih efisien dan terstruktur.

3.3.1. Flowchart sistem

Pada gambar 3.9 menunjukkan diagram alir sistem yang akan digunakan pada penelitian ini.

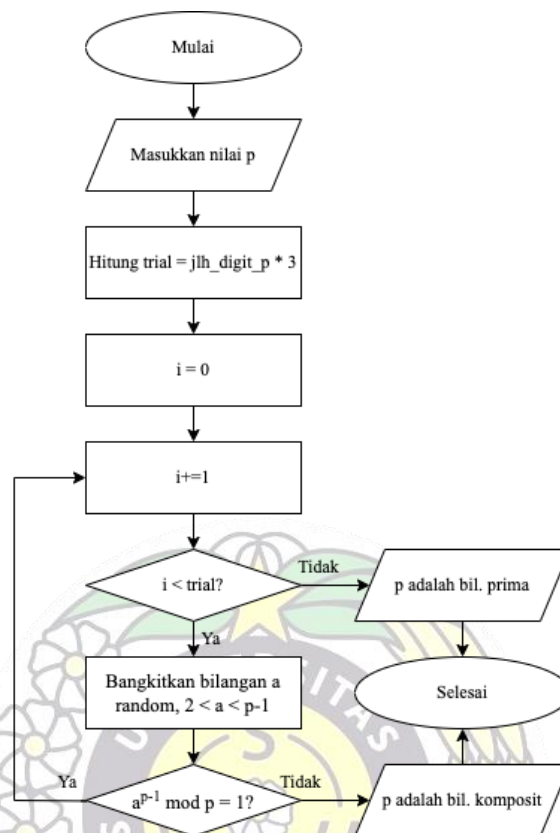


Gambar 3.9 Flowchart Sistem

3.3.2. Flowchart Fermat's Little Theorem

Flowchart pada Gambar 3.10 menggambarkan proses pengujian bilangan prima menggunakan Teorema Fermat. Proses dimulai dengan memasukkan nilai p , kemudian dihitung jumlah percobaan (*trial*) berdasarkan jumlah digit p . Selanjutnya, dilakukan iterasi dengan memilih bilangan acak a dalam rentang $2 <$

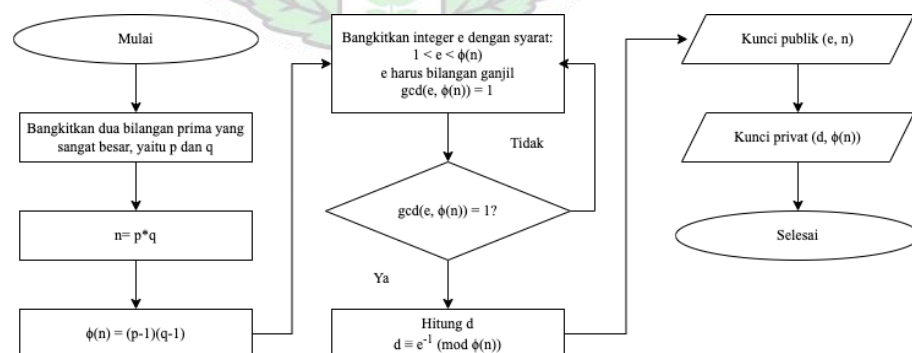
$a < p-1$. Jika $a^{p-1} \bmod p = 1$ sampai $i \geq \text{trial}$ maka nilai p lolos uji Fermat dan dianggap kandidat bilangan prima



Gambar 3.10 Flowchart Fermat's Little Theorem

3.3.3. Flowchart RSA

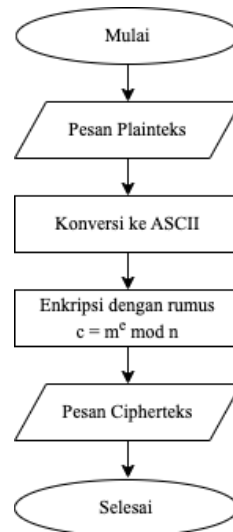
3.3.3.1 Flowchart Key Generation RSA



Gambar 3.11 Flowchart Key Generation RSA

Gambar 3.11 menunjukkan diagram alir proses pembangkitan kunci RSA. Hasil akhirnya adalah pasangan kunci publik (e, n) dan kunci privat $(d, \phi(n))$ yang dapat digunakan untuk enkripsi dan dekripsi dalam sistem RSA

3.3.3.2 Flowchart Encryption RSA



Gambar 3.12 Flowchart Encryption RSA

Gambar 3.12 di atas menunjukkan diagram alir proses enkripsi RSA. Hasil akhirnya adalah *file ciphertext*.

3.3.3.3 Flowchart Decryption RSA



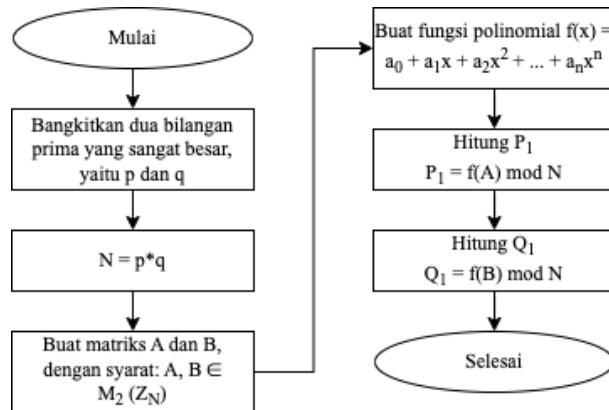
Gambar 3.13 Flowchart Decryption RSA

Gambar 3.13 menunjukkan diagram alir proses enkripsi RSA. Hasil akhirnya adalah *file plaintext*.

3.3.4. Flowchart Inam-Kanwal-Zahid-Abid (IKZA)

3.3.4.1 Flowchart Initialization IKZA

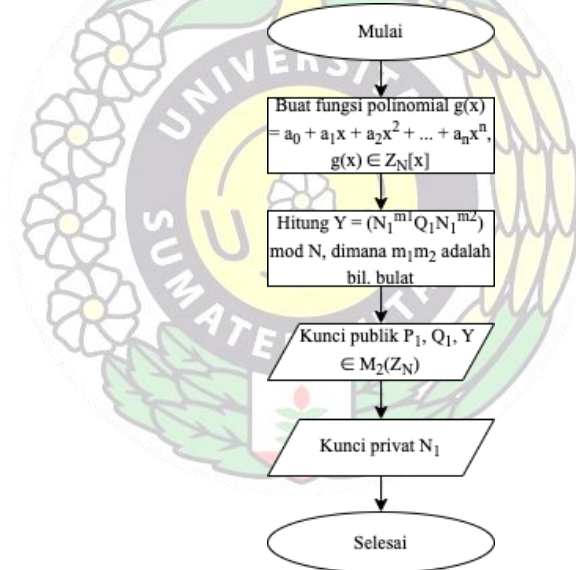
Gambar 3.14 menunjukkan diagram alir proses inisialisasi dalam sistem tanda tangan digital IKZA untuk digunakan nantinya pada proses pembangkitan kunci.



Gambar 3.14 Flowchart Initialization IKZA

3.3.4.2 Flowchart Key Generation IKZA

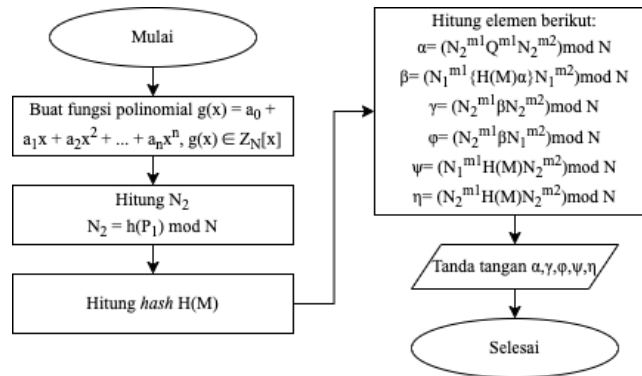
Diagram alir pada Gambar 3.15 menggambarkan proses pembangkitan kunci dalam sistem tanda tangan digital IKZA yang akan digunakan untuk proses *sign* dan *verify*.



Gambar 3.15 Flowchart Key Generation IKZA

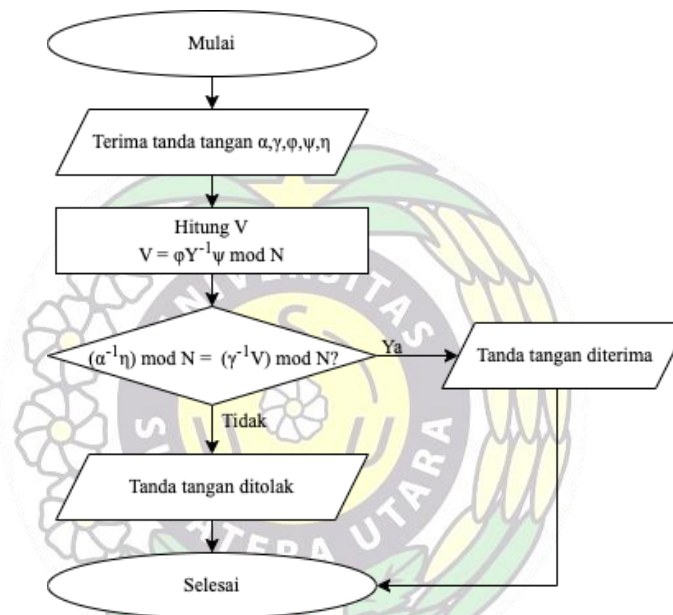
3.3.4.3 Flowchart Signature Generation IKZA

Gambar 3.16 menunjukkan diagram alir proses tanda tangan digital IKZA. Pembuatan tanda tangan dimulai dengan membuat fungsi polynomial $g(x)$, kemudian menghitung N_2 dan nilai *hash* dari pesan. Setelah itu menghitung elemen tanda tangan yang akan dikirimkan ke penerima.



Gambar 3.16 Flowchart Signature Generation IKZA

3.3.4.4 Flowchart Verification IKZA



Gambar 3.17 Flowchart Verification IKZA

Gambar 3.17 menunjukkan diagram alir proses verifikasi IKZA. Verifikasi dimulai dengan menerima tanda tangan digital. Setelah menerima nilai-nilai tersebut, langkah berikutnya adalah menghitung nilai $V = \phi Y^{-1} \psi \bmod N$, kemudian melakukan perbandingan $(\alpha^{-1} \eta) \bmod N = (\gamma^{-1} V) \bmod N$. Jika hasil perbandingan menunjukkan kesamaan antara kedua nilai, maka tanda tangan dianggap valid atau diterima. Sebaliknya, jika hasil perbandingan menunjukkan perbedaan, maka tanda tangan digital dinyatakan tidak valid atau ditolak.

BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1. Implementasi

Dalam penelitian ini, sistem dikembangkan menggunakan *Django* sebagai *back-end* dengan *Django Template Engine* untuk antarmuka. Desain dibuat responsif menggunakan Bootstrap 5.3, dengan struktur berbasis *template inheritance*, di mana *base.html* menjadi template utama yang digunakan oleh semua halaman. Navigasi utama terdiri dari menu *Home*, *Key Generation*, *Encrypt & Sign*, *Verify & Decrypt*, dan *Brute-force* untuk memudahkan akses ke fitur-fitur. Sistem dikembangkan dengan menggunakan *Visual Studio Code* sebagai IDE dan spesifikasi *hardware*: *processor* 1,8 GHz Dual-Core Intel Core i5, RAM 8 GB 1600 MHz DDR3.

4.1.1. Laman Home



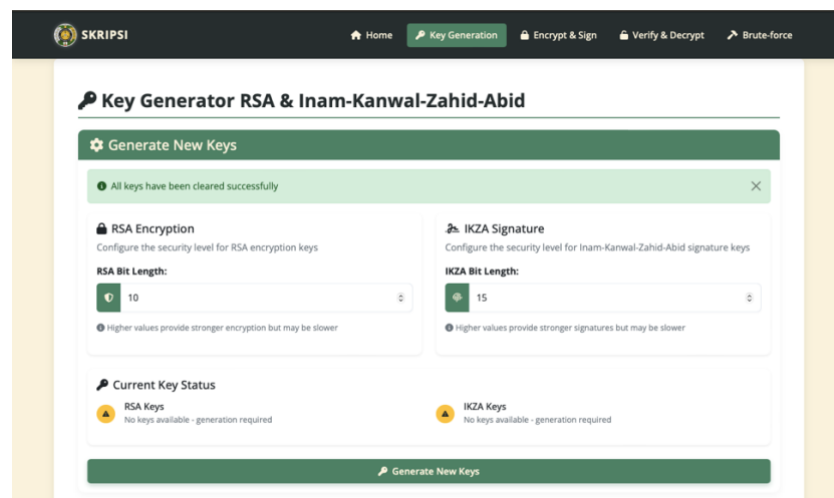
Gambar 4.1 Laman *Home*

Pada Gambar 4.1 dapat dilihat laman *Home*, yaitu halaman utama dari sistem "*SIGNCRYPTION* DENGAN ALGORITMA RSA DAN SKEMA TANDA TANGAN DIGITAL INAM-KANWAL-ZAHID-ABID (IKZA)" yang dikembangkan oleh peneliti. Laman ini dirancang untuk memberikan pemahaman

awal kepada pengguna tentang bagaimana sistem *hybrid signcryption* RSA dan IKZA bekerja, sebelum pengguna menggunakan fitur-fitur yang tersedia pada menu navigasi untuk keperluan keamanan data mereka.

4.1.2. Laman Key Generation

Laman *Key Generation* merupakan laman penting yang berfungsi untuk menghasilkan pasangan kunci kriptografi yang digunakan dalam proses enkripsi dan tanda tangan digital. Laman ini memungkinkan pengguna untuk mengkonfigurasi tingkat keamanan melalui pengaturan panjang bit untuk dua jenis algoritma: RSA untuk enkripsi dan IKZA untuk tanda tangan digital.



Gambar 4.2 Laman *Key Generation*

Pada Gambar 4.2 dapat dilihat bahwa pengguna dapat menentukan nilai bit *length* (dengan nilai default 10 untuk RSA dan 15 untuk IKZA) dengan catatan bahwa nilai yang lebih tinggi akan memberikan keamanan yang lebih kuat namun dengan proses yang lebih lambat. Laman ini juga menampilkan status kunci saat ini dan memberikan tombol "*Generate New Keys*" untuk memulai proses pembangkitan kunci.

4.1.3. Laman *Encrypt & Sign*

Laman *Encrypt & Sign* merupakan antarmuka yang memungkinkan pengguna untuk melakukan enkripsi dokumen dan pemberian tanda tangan digital secara simultan. Pada Gambar 4.3 diperlihatkan bahwa laman ini menyediakan area untuk mengunggah dokumen dengan dukungan format .docx, .doc, .pdf, dan .txt.

The screenshot shows the 'Document Encryption & Digital Signature' page of the SKRIPSI application. The top navigation bar includes links for Home, Key Generation, Encrypt & Sign (active), Verify & Decrypt, and Brute-force. The main content area is divided into three sections:

- Upload Your Document:** Contains a 'Select Document' section with a 'Choose File' button and 'No file chosen' text. Below it, supported formats are listed as .docx, .doc, .pdf, and .txt. The 'Select Output Format' dropdown is set to 'Text File (.txt)'. A green 'Encrypt & Sign Document' button is at the bottom of this section.
- Keys Status:** Displays a warning 'No Keys Found: You must generate keys before uploading a document.' with a 'Keys required' icon. A green 'Generate Keys Now' button is located below the warning.
- How It Works:** A list of five steps explaining the process:
 - This process encrypts your document using RSA encryption and generates a digital signature using the Inam-Kanwal-Zahid-Abid signature scheme.
 - Upload your document in .txt, .docx, .doc, or .pdf format.
 - The system extracts text from the document and converts it into an ASCII matrix.
 - The text is encrypted using RSA with the public key stored in the session.
 - A digital signature is generated using Inam-Kanwal-Zahid-Abid polynomial-based signing.
 - The output format can be selected as .txt, .docx, or .pdf.

Gambar 4.3 Laman *Encrypt and Sign*

Pengguna juga dapat memilih dokumen melalui tombol "*Choose File*" dan menentukan format keluaran yang diinginkan melalui *dropdown* yang tersedia. Sistem ini memeriksa status kunci kriptografi terlebih dahulu, dan jika belum ada kunci yang dibuat, akan menampilkan peringatan "*No Keys Found*" beserta tombol "*Generate Keys Now*" untuk membuat kunci yang diperlukan.

4.1.4. Laman *Verify & Decrypt*

Laman *Verify & Decrypt* merupakan fitur yang berfungsi untuk memvalidasi dan mendekripsi dokumen terenkripsi yang memiliki tanda tangan digital. Pada halaman ini seperti terlihat pada Gambar 4.4, pengguna dapat mengunggah dokumen ciphertext yang sudah ditandatangani dalam format .pdf, .docx, atau .txt melalui tombol "*Choose File*".

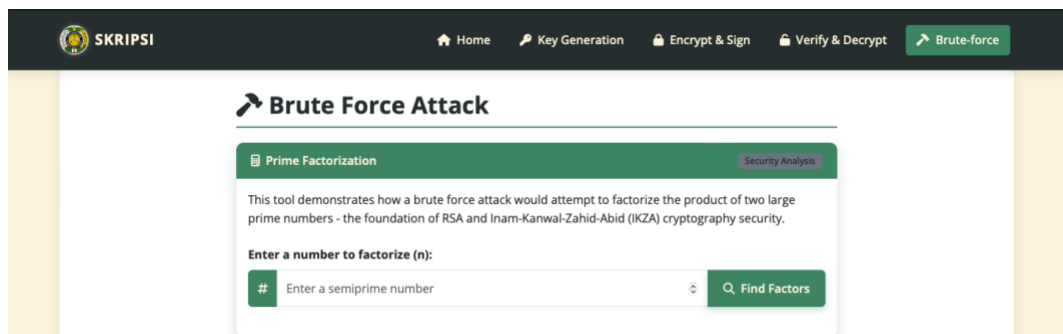
The screenshot shows the 'Verify and Decrypt Document' page of the SKRIPSI application. The top navigation bar includes links for Home, Key Generation, Encrypt & Sign, Verify & Decrypt (active), and Brute-force. The main content area is divided into two sections:

- Upload Signed Ciphertext:** Contains a 'Choose File' button and 'no file selected' text. Below it, supported formats are listed as pdf, docx, and txt. A green 'Verify and Decrypt' button is at the bottom of this section.
- Verification Process:** A green button labeled 'Verification Process' is located at the top right of the main content area.

Gambar 4.4 Laman *Verify and Decrypt*

Proses verifikasi dan dekripsi dilakukan secara berurutan dimana sistem akan mengekstrak teks dari dokumen, memeriksa keabsahan tanda tangan digital menggunakan kunci kriptografi yang tersimpan, dan jika verifikasi berhasil, dokumen akan didekripsi menggunakan kunci privat RSA. Hasil dekripsi akan disimpan sebagai file teks yang dapat diunduh. Fitur ini menjamin keamanan data karena semua pemrosesan dilakukan secara lokal dan tidak disimpan di server manapun.

4.1.5. Laman Brute-force



Gambar 4.5 Laman Brute-force

Laman *Brute-force* mendemonstrasikan bagaimana serangan *brute-force* bekerja terhadap enkripsi RSA dan tanda tangan IKZA dengan mencoba memfaktorkan produk dari dua bilangan prima besar. Pada Gambar 4.5 pengguna dapat memasukkan angka (n) yang merupakan hasil perkalian dua bilangan prima, kemudian mencoba menemukan faktor-faktor prima tersebut.

4.2. Pengujian

Dalam tahap pengujian sistem, pengujian akan dilakukan untuk memastikan bahwa sistem dapat menjalankan proses enkripsi dan dekripsi pesan dengan benar. Sistem harus mampu mengenkripsi pesan menggunakan algoritma RSA serta menghasilkan tanda tangan digital menggunakan algoritma IKZA. Selain itu, sistem juga akan diuji untuk memastikan bahwa pesan yang telah dienkripsi dapat dikembalikan ke bentuk aslinya setelah melalui proses verifikasi tanda tangan dan dekripsi, sehingga validitas serta integritas data tetap terjaga.

4.2.1. Pengujian Key Generation

Gambar 4.6 Pengujian *Key Generation*

Pada Gambar 4.6 di atas dimasukkan panjang bit yang diinginkan untuk proses pembangkitan kunci. Panjang bit default 10 untuk RSA dan 15 untuk IKZA dengan catatan bahwa nilai yang lebih tinggi akan memberikan keamanan yang lebih kuat namun dengan proses yang lebih lambat. Status kunci “*available*” setelah menekan “*Generate New Keys*”, dimana kunci yang baru juga bisa dihapus menggunakan button “*Clear Existing Keys*” yang nantinya akan menyebabkan statusnya menjadi “*no keys available*”.

Pada Gambar 4.7 ditunjukkan hasil dari pembangkitan kunci RSA dan IKZA dimana terdapat *private key* dan *public key* masing-masing algoritma.

Gambar 4.7 Hasil Pengujian *Key Generation*

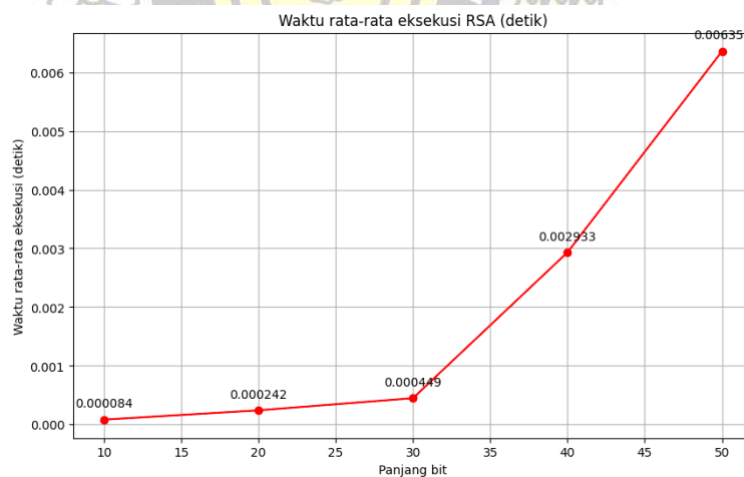
4.2.2. Pengujian Real Running Time Key Generation RSA

Tabel 4.1 menunjukkan waktu eksekusi generasi kunci RSA dengan variasi panjang bit dari 10 hingga 50 bit, menampilkan fluktuasi waktu pada setiap iterasi dan kecenderungan peningkatan waktu rata-rata seiring bertambahnya panjang bit.

Tabel 4.1 Waktu eksekusi *key generation* RSA terhadap panjang bit

Panjang bit	Waktu Eksekusi (s)					
	Iterasi-1	Iterasi-2	Iterasi-3	Iterasi-4	Iterasi-5	Rata-rata
10	0.000121	0.000059	0.000080	0.000068	0.000092	0.000084
20	0.000324	0.000210	0.000256	0.000197	0.000222	0.000242
30	0.000468	0.000386	0.000349	0.000609	0.000433	0.000449
40	0.001933	0.007856	0.001633	0.001740	0.001502	0.002933
50	0.009781	0.002491	0.002374	0.014552	0.002588	0.006357

Gambar 4.8 merupakan grafik yang memvisualisasikan data dari Tabel 4.1 tentang waktu eksekusi generasi kunci RSA, menunjukkan peningkatan eksponensial waktu running time seiring bertambahnya panjang bit dari 10 hingga 50 bit, dengan garis merah yang menandakan bahwa semakin panjang bit kunci yang dihasilkan, semakin signifikan waktu yang dibutuhkan untuk proses generasi kunci RSA.



Gambar 4.8 Grafik Pengujian *Real Running Time Key Generation* RSA

4.2.3. Pengujian Real Running Time Key Generation Inam-Kanwal-Zahid-Abid (IKZA)

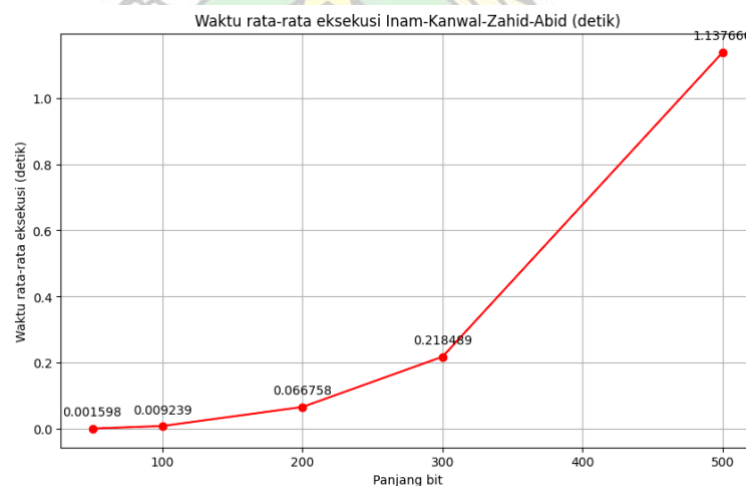
Tabel 4.2 menunjukkan waktu eksekusi untuk proses *key generation* IKZA dengan variasi panjang bit mulai dari 50 hingga 500 bit, di mana setiap iterasi menampilkan

waktu eksekusi yang berbeda dengan rata-rata waktu yang semakin meningkat seiring bertambahnya panjang bit.

Tabel 4.2 Waktu eksekusi *key generation* IKZA terhadap panjang bit

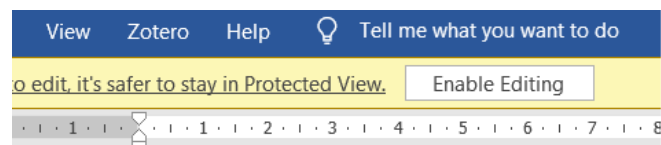
Panjang bit	Waktu Eksekusi (s)					
	Iterasi-1	Iterasi-2	Iterasi-3	Iterasi-4	Iterasi-5	Rata-rata
50	0.001678	0.001459	0.002141	0.001381	0.001330	0.001598
100	0.006837	0.012327	0.007751	0.010004	0.009274	0.009239
200	0.086640	0.060617	0.060097	0.070735	0.055699	0.066758
300	0.251346	0.143128	0.235852	0.217220	0.244896	0.218489
500	1.854857	1.024158	0.966228	1.032122	0.810966	1.137666

Gambar 4.9 memvisualisasikan grafik data dari Tabel 4.2 dengan menampilkan hubungan eksponensial antara panjang bit dan waktu eksekusi, di mana garis merah menunjukkan peningkatan waktu *running time* yang signifikan seiring bertambahnya ukuran bit.



Gambar 4.9 Grafik Pengujian *Real Running Time Key Generation* IKZA

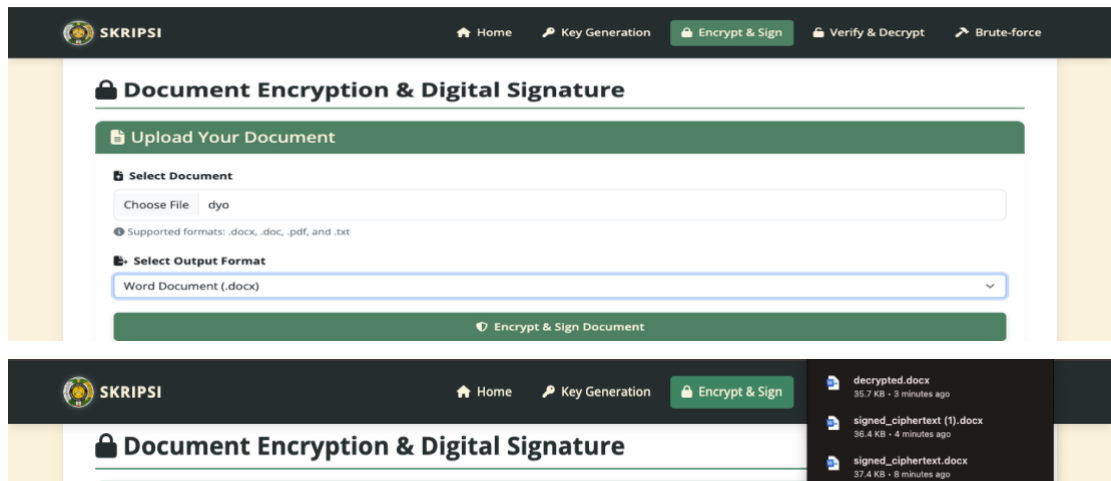
4.2.4. Pengujian *Encrypt-then-Sign*



Saya suka d.o

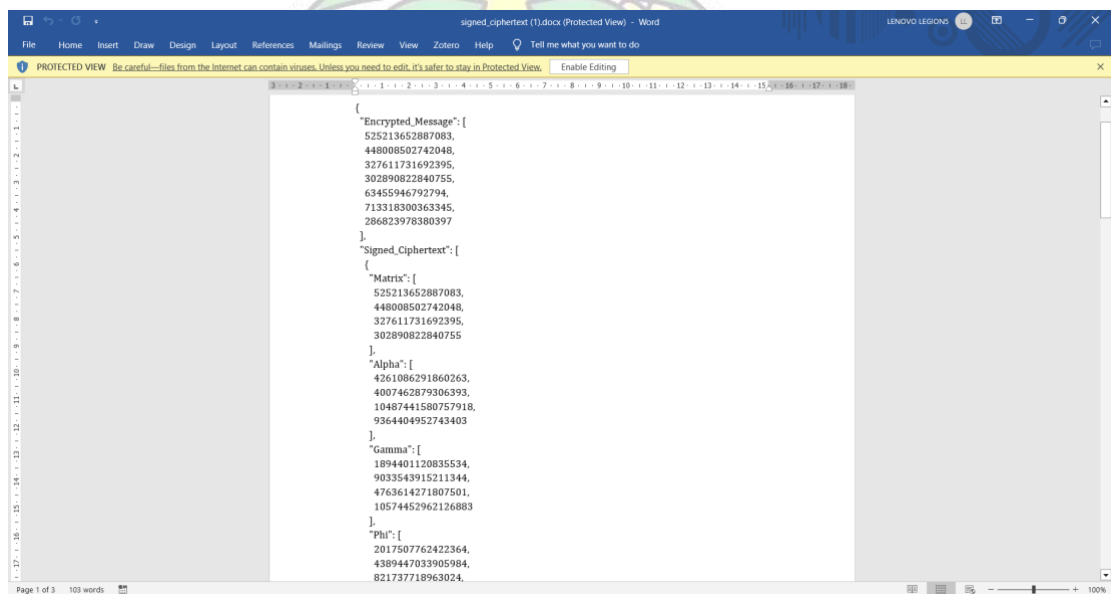
Gambar 4.10 Pesan asli

Gambar 4.10 di atas menunjukkan isi pesan yang akan dimasukkan untuk proses *signcryption*.



Gambar 4.11 Pengujian *Encrypt-then-Sign*

Setelah itu, pesan dimasukkan dalam sistem untuk proses *encrypt and sign*. Setelah menekan button “*Encrypt and Sign Document*”, maka file hasilnya akan otomatis terunduh seperti pada Gambar 4.11.



Gambar 4.12 Hasil Pengujian *Encrypt-then-Sign*

Isi dari file hasil proses *Encrypt and Sign* dapat dilihat pada Gambar 4.12 dimana file diformat dalam bentuk JSON supaya lebih mudah untuk *parsing* pada saat proses *Verify and Decrypt* nantinya. Terdapat pesan yang terenkripsi dan hasil tanda tangan yang terdiri dari komponen *matrix*, *alpha*, *gamma*, *phi*, *psi*, dan *eta*

4.2.5. Pengujian Real Running Time *Encrypt-then-Sign*

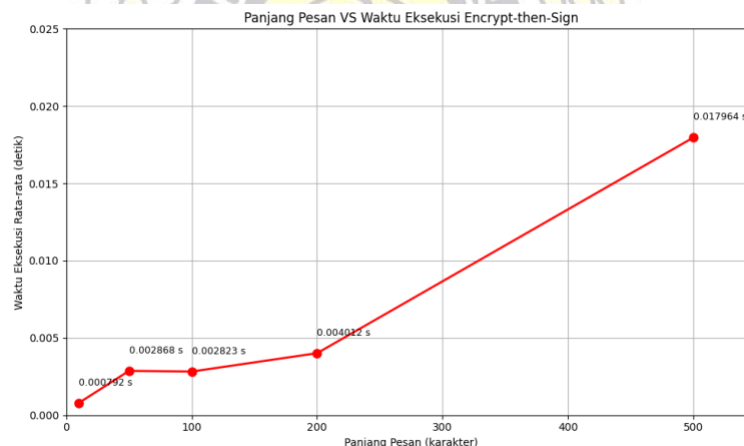
Tabel 4.3 menunjukkan waktu eksekusi proses *encrypt-then-sign* dengan variasi panjang pesan dari 10 karakter hingga 500 karakter, memperlihatkan variasi waktu

eksekusi pada setiap iterasi dengan rata-rata waktu yang bertambah seiring meningkatnya ukuran pesan.

Tabel 4.3 Waktu eksekusi *encrypt RSA then sign* IKZA terhadap panjang pesan

Panjang pesan (karakter)	Waktu Eksekusi (s)					
	Iterasi-1	Iterasi-2	Iterasi-3	Iterasi-4	Iterasi-5	Rata-rata
10	0.0008	0.0009	0.0007	0.0007	0.0008	0.000792
50	0.0012s,	0.0085s,	0.0015s,	0.0012s,	0.0019s	0.002868s
100	0.0019s,	0.0023s,	0.0030s,	0.0048s,	0.0020s	0.002823s
200	0.0031s,	0.0038s,	0.0068s,	0.0031s,	0.0032s	0.004012s
500	0.0238s	0.0122	0.0288	0.0175	0.0076	0.017964

Gambar 4.13 adalah grafik yang menggambarkan hubungan antara ukuran pesan dan waktu eksekusi proses encrypt-then-sign berdasarkan data pada Tabel 4.3, dengan garis biru yang menunjukkan peningkatan bertahap waktu running time seiring bertambahnya panjang karakter pesan dari 10 hingga 500 karakter, mengindikasikan bahwa proses enkripsi dan penandatanganan memerlukan waktu yang semakin lama ketika ukuran pesan semakin besar.



Gambar 4.13 Grafik Pengujian *Real Running Time Encrypt-then-Sign*

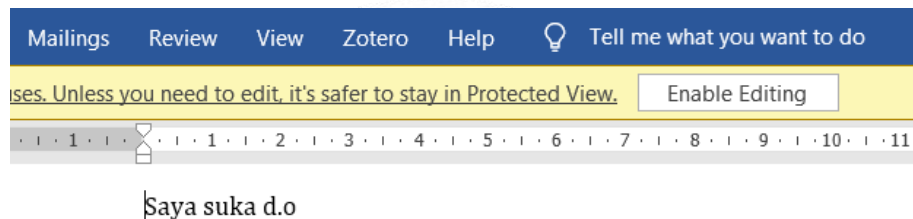
Pengujian Verify-then-Decrypt

Pada laman ini di-upload file hasil proses *Encrypt and Sign* sebelumnya. Setelah menekan button “*Verify and Decrypt*”, maka akan muncul pesan “*Success*” jika validasi berhasil dilakukan.



Gambar 4.14 Pengujian *Verify-then-Decrypt*

Kemudian pengguna bisa memilih format teks dekripsi yang akan diunduh. Setelah mengunduh *file* hasil dekripsi, isi dari *file* tersebut dapat dilihat pada Gambar 4.15 di bawah.



Gambar 4.15 Hasil Pengujian *Verify-then-Decrypt*

File hasil proses *signcryption* terbukti sama dengan *file* yang dimasukkan sebelum proses pada Gambar 4.10.

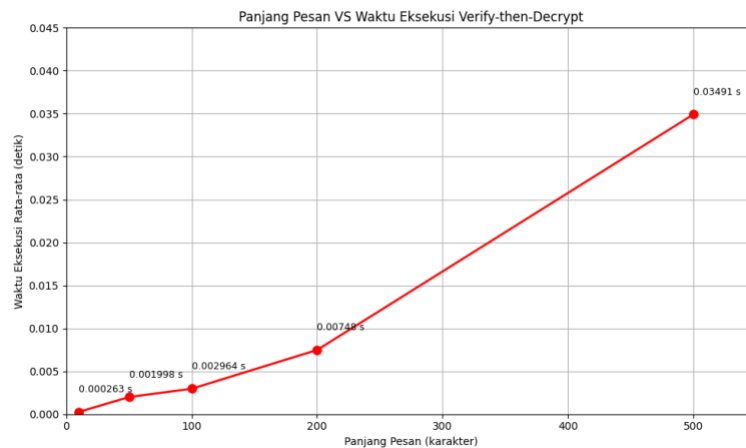
4.2.6. Pengujian Real Running Time *Verify-then-Decrypt*

Tabel 4.4 Waktu eksekusi *verify IKZA then decrypt RSA* terhadap panjang pesan

Panjang pesan (karakter)	Waktu Eksekusi (s)					
	Iterasi-1	Iterasi-2	Iterasi-3	Iterasi-4	Iterasi-5	Rata-rata
10	0.0003	0.0003	0.0003	0.0003	0.0003	0.000263
50	0.0030	0.0028	0.0010	0.0021	0.0011	0.001998
100	0.0048	0.0021	0.0040	0.0020	0.0020	0.002964
200	0.0054	0.0050	0.0095	0.0086	0.0089	0.007480
500	0.0782	0.0277	0.0208	0.0190	0.0288	0.034910

Tabel 4.4 memperlihatkan waktu eksekusi proses verifikasi dan dekripsi dengan variasi panjang pesan dari 5 hingga 500 karakter, menunjukkan fluktuasi waktu eksekusi dalam setiap iterasi dengan rata-rata waktu yang meningkat seiring bertambahnya ukuran pesan.

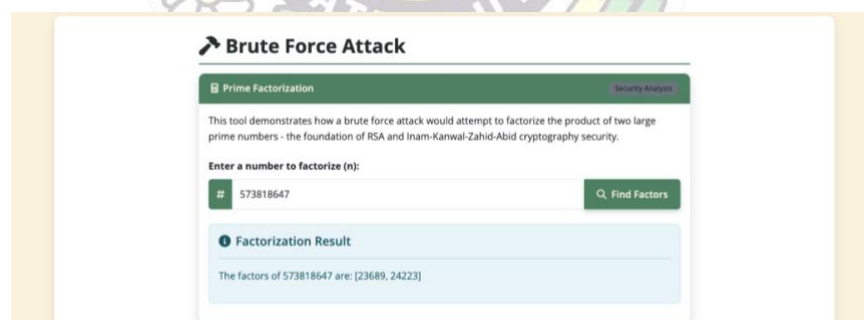
Grafik pada Gambar 4.16 menunjukkan hubungan antara ukuran pesan dan waktu *running time verify-then-decrypt*, dengan garis merah miring ke atas yang mengindikasikan peningkatan waktu eksekusi seiring bertambahnya panjang karakter pesan.



Gambar 4.16 Grafik Pengujian *Real Running Time Verify-then-Decrypt*

Hal ini menunjukkan karakteristik kinerja yang kompleks dengan pertumbuhan superlinear yang lebih signifikan untuk pesan panjang. Ini konsisten dengan operasi kriptografi asimetris yang biasanya memiliki kompleksitas komputasional yang lebih tinggi untuk data yang lebih besar.

4.2.7. Pengujian Brute-Force Attack



Gambar 4.17 Hasil Pengujian *Brute-Force*

Pengguna dapat memasukkan bilangan *semiprime* (n pada RSA dan IKZA) untuk mendapatkan dua bilangan prima. Hasil dari proses *brute-force* dapat dilihat pada Gambar 4.17.

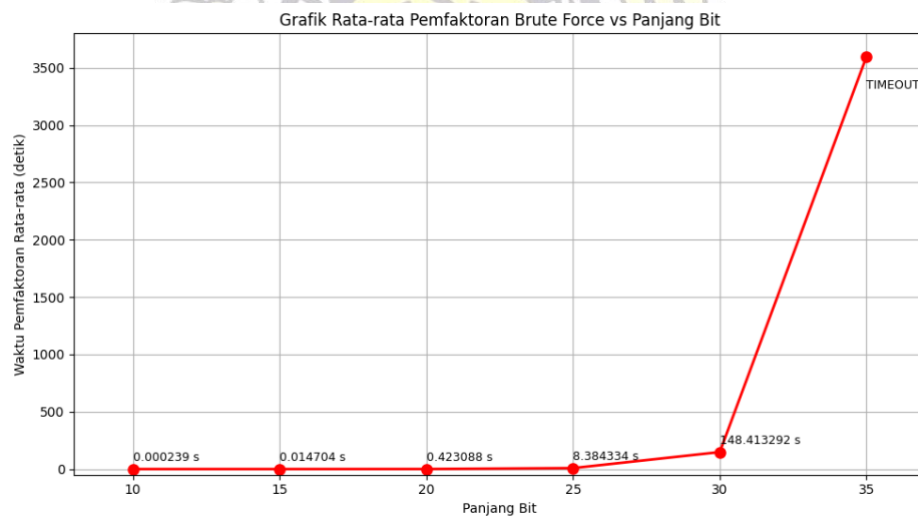
4.2.8. Pengujian Real Running Time Faktorisasi Brute-Force

Tabel 4.5 Waktu eksekusi Faktorisasi Brute-Force

Panjang bit	Waktu Eksekusi (s)					
	Iterasi-1	Iterasi-2	Iterasi-3	Iterasi-4	Iterasi-5	Rata-rata
10	0.000309	0.000233	0.000226	0.000258	0.000169	0.000239
15	0.015930	0.016541	0.011682	0.015447	0.013922	0.014704
20	0.456953	0.469377	0.431190	0.548913	0.209008	0.423088
25	16.046107	10.044576	6.672403	5.360856	3.797727	8.384334
30	164.29531	159.90323	157.41634	129.60380	130.84775	148.41329
35	7	7	3	6	6	2
35	3600	3600	3600	3600	3600	3600

NB: *TIMEOUT* after 3600 seconds

Tabel 4.5 menampilkan waktu eksekusi proses faktorisasi *brute-force* dengan variasi panjang bit dari 10 hingga 35 bit, memperlihatkan peningkatan drastis waktu eksekusi seiring bertambahnya panjang bit, dengan catatan beberapa percobaan mencapai *timeout* setelah 3600 detik.



Gambar 4.18 Grafik Pengujian Real Running Time Brute-Force

Grafik pada Gambar 4.18 adalah representasi grafis dari Tabel 4.5 yang menggunakan skala linear untuk menunjukkan pertumbuhan eksponensial waktu *running time* faktorisasi *brute-force*. Garis merah dengan jelas mendokumentasikan peningkatan waktu eksekusi yang sangat tajam seiring bertambahnya panjang bit. Pada panjang bit 35, algoritma mencapai *timeout* (3600 detik) yang mengindikasikan bahwa metode *brute-force* menjadi sangat tidak efisien untuk kunci dengan panjang bit yang besar.

Peneliti melakukan pengujian ini di laptop sendiri (komputer *stand alone*) tanpa menggunakan sistem terdistribusi atau komputasi cloud. Pengujian dilaksanakan secara mandiri menggunakan spesifikasi hardware yang tersedia pada laptop pribadi peneliti, yang memberikan gambaran nyata tentang kinerja algoritma faktorisasi *brute-force* dalam lingkungan komputasi yang umum digunakan.



BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan pengujian sistem yang telah dilakukan, penelitian ini menyimpulkan bahwa:

1. Sistem yang dibuat berhasil menerapkan *signcryption* menggunakan metode *encrypt-then-sign* pada file teks (.txt), dokumen (.docx), dan PDF (.pdf) tanpa mencakup format lain seperti gambar, video, atau data biner
2. Proses *encrypt-then-sign* menunjukkan performa yang efisien bahkan untuk pesan panjang (500 karakter) dengan waktu rata-rata 0.017964 detik. Proses *verify-then-decrypt* juga menunjukkan kinerja yang baik dengan waktu rata-rata 0.03491 detik untuk pesan 500 karakter.
3. Hasil pengujian menunjukkan bahwa pesan yang telah melalui proses *signcryption* dapat dikembalikan ke bentuk aslinya setelah melalui proses verifikasi tanda tangan dan dekripsi, membuktikan bahwa sistem menjaga integritas data dengan baik.
4. Pembangkitan kunci RSA dan IKZA menunjukkan kinerja yang baik pada panjang bit yang wajar, dengan kompleksitas superlinear yang dapat diterima. RSA membutuhkan waktu rata-rata 0.006357 detik untuk panjang bit 50, sementara IKZA membutuhkan 1.137666 detik untuk panjang bit 500.
5. Hasil pengujian *brute-force* menunjukkan bahwa sistem memiliki ketahanan yang sangat baik terhadap serangan, dengan bilangan 30-bit membutuhkan waktu rata-rata 148.4 detik dan bilangan 35-bit tidak dapat difaktorkan dalam batas waktu 600 detik. Ini memvalidasi keamanan sistem kriptografi yang mengandalkan kesulitan pemfaktoran bilangan besar.

5.2. Saran

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa saran yang dapat dipertimbangkan untuk pengembangan lebih lanjut:

1. Penelitian yang telah dilakukan hanya berfokus pada implementasi kriptografi kunci publik dan tidak memanfaatkan *hybrid cryptosystem*. Untuk pengembangan lebih lanjut dalam hal efisiensi dan keamanan, penelitian berikutnya dapat diarahkan pada perancangan skema *hybrid cryptosystem* yang menggabungkan kelebihan dari kriptografi kunci publik dengan kriptografi kunci simetris.
2. Untuk penelitian selanjutnya, disarankan bahwa seharusnya *ciphertext* tidak langsung di-*sign* tetapi nilai *hash* dari *ciphertext*. Pendekatan ini dapat meningkatkan efisiensi proses dan keamanan sistem secara keseluruhan, karena operasi tanda tangan digital pada nilai *hash* memerlukan komputasi yang lebih ringan dibandingkan dengan penandatanganan seluruh *ciphertext*.
3. Penelitian ini dilakukan pada *computer stand alone*, diharapkan penelitian selanjutnya dapat diimplementasikan langsung ke kondisi *real* pada berbagai *platform* seperti keamanan pesan instan *WhatsApp* untuk memastikan kekuatan enkripsi *end-to-end*, perangkat IoT dengan daya komputasi terbatas untuk mengoptimalkan penggunaan sumber daya, sistem transaksi perbankan online untuk mencegah pemalsuan dan akses tidak sah, serta sistem manajemen kunci pada infrastruktur cloud untuk melindungi data sensitif dari serangan *brute-force* yang dapat mengeksploitasi panjang bit kunci yang tidak memadai.
4. Perluas dukungan format file yang dapat dienkripsi dan ditandatangani, termasuk format binary seperti gambar atau video, untuk memperluas kegunaan sistem.
5. Pertimbangan untuk menetapkan panjang bit minimum yang lebih tinggi untuk implementasi praktis (misalnya minimal 1024-bit untuk RSA dan 2048-bit untuk IKZA) untuk menjamin keamanan jangka panjang mengingat peningkatan kapasitas komputasi.
6. Lakukan analisis dan optimasi penggunaan memori, terutama untuk operasi kriptografi pada data besar, untuk memastikan sistem tetap efisien dan responsif.

DAFTAR PUSTAKA

- Arifin, Z. (2016). Studi Kasus Penggunaan Algoritma RSA Sebagai Algoritma Kriptografi yang Aman. *Informatika Mulawarman: Jurnal Ilmiah Ilmu Komputer*, 4(3), 35-41. <https://doi.org/10.30872/jim.v4i3.43>
- Bahri, R., Budiman, M., & Nasution, B. (2023). Sign-Then-Encrypt Scheme with Cramer-Shoup Cryptosystem and Dissanayake Digital Signature. In *Proceedings of the 3rd International Conference on Advanced Information Scientific Development*, Vol. 1, 131–138. <https://doi.org/10.5220/0012444900003848>
- Basharat, I., Azam, F., & Wahab Muzaffar, A. (2012). Database Security and Encryption: A Survey Study. *International Journal of Computer Applications*, 47(12), 28–34. <https://doi.org/10.5120/7242-0218>
- Basri. (2015). Pendekatan Kriptografi Hybrid pada Keamanan Dokumen Elektronik dan HypertextTransfer Protocol Secure (HTTPS) (Analisis Potensi Implementasi Pada Sistem Keamanan). *Jurnal Ilmiah Ilmu Komputer Fakultas Ilmu Komputer Universitas Al Asyariah Mandar*, 1(2), 15-24. <https://doi.org/10.35329/jiik.v1i2.70>
- Cipta, H. (2020). Completion of Matrix Inversions Using Elementary Matrix Inverse Multiplication Method. *International Journal of Science, Technology & Management*, 1(2), 56-63. <https://doi.org/10.46729/ijstm.v1i2.8>
- Ginting, C. L., Budiman, M. A., & Nasution, S. (2024). Signcryption with Matrix Modification of RSA Digital Signature Scheme and Cayley-Purser Algorithm. *Data Science: Journal of Computing and Applied Informatics*, 8(1), 68-79. <https://doi.org/10.32734/jocai.v8.i1-12226>
- 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), 44-51. <https://doi.org/10.33422/ejest.v3i1.157>
- Jonathan, K. (2015). *Introduction to Modern Cryptography* (2nd ed.). CRC Press.
- Muchlis, B. S. (2015, November 26). *Teknik Pemecahan Kunci Algoritma Rivest Shamir Adleman (RSA) Dengan Metode Kraitichik*. Universitas Indonesia, Vol. 3,

- 112-128. <https://www.semanticscholar.org/paper/Teknik-Pemecahan-Kunci-Algoritma-Rivest-Shamir-Muchlis/a628252a57db4b11530b0b0411255a248d6dad83>
- Munir, R. (2019). *Pengantar Kriptografi* (3rd ed.). Bandung: Informatika Bandung, 215-267.
- Nuraini, R. (2015). Desain Algoritma Operasi Perkalian Matriks Menggunakan Metode Flowchart. *Jurnal Teknik Komputer*, 1(1), 144-149. <https://doi.org/10.31294/jtk.v1i1.245>
- Prasetyo, A., Arief, S. N., & Wakhidah, R. (2021). Optimasi Pemrosesan Enkripsi Dan Dekripsi RSA Pada Single Board Computer (SBC) Dengan Pembagian Beban Komputasi Dalam Sistem Terdistribusi. *Jurnal Informatika Polinema*, 7(4), 89-97. <https://doi.org/10.33795/jip.v7i4.523>
- 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. <https://doi.org/10.1145/359340.359342>
- Smart, N. P. (2016). *Cryptography Made Simple* (1st ed.). Springer International Publishing, 101-156. <https://doi.org/10.1007/978-3-319-21936-3>
- Stallings, W. (2017). *Cryptography and network security: Principles and practice* (7th ed., global edition). Pearson, Vol. 1, 402-445.
- Zheng, Y. (1997). Digital signcryption or how to achieve $\text{cost}(\text{signature} \& \text{encryption}) \ll \text{cost}(\text{signature}) + \text{cost}(\text{encryption})$. In B. S. Kaliski (Ed.), *Advances in Cryptology—CRYPTO '97* (Vol. 1294, pp. 165–179). Springer. <https://doi.org/10.1007/BFb0052234>