

**SIGNCRYPTION CRYPTOSYSTEM SCHEME DENGAN
ALGORITMA XRSA PUBLIC KEY ENCRYPTION DAN
NAGATY DIGITAL SIGNATURE**

SKRIPSI

RODIATUL HUSNA BR SITEPU

211401038



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

2025

**SIGNCRYPTION CRYPTOSYSTEM SCHEME DENGAN
ALGORITMA XRSA PUBLIC KEY ENCRYPTION DAN
NAGATY DIGITAL SIGNATURE**

SKRIPSI

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

RODIATUL HUSNA BR SITEPU

211401038



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

2025

PERSETUJUAN

Judul : SIGNCRYPTION CRYPTOSYSTEM SCHEME
DENGAN ALGORITMA XRSA PUBLIC KEY
ENCRYPTION DAN NAGATY DIGITAL
SIGNATURE

Kategori : SKRIPSI

Nama : RODIATUL HUSNA BR SITEPU

Nomor Induk Mahasiswa : 211401038

Program Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI
INFORMASI UNIVERSITAS SUMATERA UTARA

Komisi Pembimbing :

Pembimbing II

Pembimbing I



Amer Sharif, S.Si, M.Kom.

NIP. 196910212021011001



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

UNIVERSITAS SUMATERA UTARA

PERNYATAAN**SIGNCRYPTION CRYPTOSYSTEM SCHEME DENGAN ALGORITMA XRSA
PUBLIC KEY ENCRYPTION DAN NAGATY DIGITAL SIGNATURE****SKRIPSI**

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

Medan, 25 Maret 2025



Rodiatul Husna Br Sitepu
211401038



PENGHARGAAN

Dengan menyebut nama Allah Yang Maha Pengasih lagi Maha Penyayang, segala puji dan syukur senantiasa dipanjatkan ke hadirat Allah *Subhanahu Wa Ta'ala* atas rahmat dan petunjuk-Nya, yang memungkinkan penulis menyelesaikan penyusunan skripsi ini sebagai bagian dari persyaratan untuk memperoleh gelar Sarjana Komputer di Program Studi S-1 Ilmu Komputer, USU. Shalawat serta salam juga senantiasa tercurah kepada Nabi Muhammad *Shalallaahu 'Alayhi Wasallam*, yang telah membimbing umat manusia dari kegelapan menuju jalan yang terang benderang.

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. Bapak Prof. Dr. Muryanto Amin, S.Sos., M.Si selaku Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lydia, B.Sc., M.Sc. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Ibu Dr. Amalia, S.T., M.T., selaku Ketua Program Studi S1 Ilmu Komputer Universitas Sumatera Utara dan Pembimbing Akademik penulis yang membimbing penulis selama berkuliah di S-1 Ilmu Komputer.
4. 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.
5. Bapak Amer Sharif, S.Si, M.Kom selaku Dosen Pembimbing II penulis yang telah memberikan bimbingan dan arahan dalam menyelesaikan skripsi ini.
6. Ibu Dian Rachmawati, S.Si., M.Kom sebagai Dosen Penguji yang telah memberikan masukan berharga dan saran konstruktif kepada penulis.
7. Ibu Anandhini Medianty Nababan, S.Kom., M.T sebagai Dosen Penguji yang turut memberikan kritik membangun serta arahan yang membantu penulis dalam menyempurnakan karya ini.

8. Seluruh Bapak/Ibu Dosen dan Staf Pegawai Ilmu Komputer yang telah membantu penulis selama melaksanakan perkuliahan.
9. Orangtua penulis yang selalu mendoakan, memberikan semangat, dan dukungan yang berharga kepada penulis.
10. Saudara kandung penulis, Laila Sitepu dan Murina Sitepu yang selalu memberikan dukungan, doa, dan hiburan di saat penulis mengalami kebuntuan selama proses pengerjaan skripsi.
11. Sahabat empat serangkai penulis Yessica Situmorang, A. Tampubolon, dan Johana Sihotang yang selalu memotivasi serta mengajak penulis untuk mengerjakan skripsi agar lulus tepat waktu, serta menyediakan *basecamp* untuk mengerjakan skripsi bersama-sama.
12. Sahabat penulis dalam masa perkuliahan Agatha Sinaga, Sammytha Siagian, dan Elin Br Ginting yang selalu memberikan semangat, dukungan, serta turut mendorong penulis dalam proses penyelesaian skripsi.
13. Seluruh pihak yang telah berkontribusi dalam mendukung dan membantu penulis dalam menyelesaikan skripsi ini, yang tidak dapat disebutkan satu persatu. Semoga segala kebaikan, bantuan, perhatian, dan dukungan yang diberikan mendapat balasan berkah dari Allah *Subhanahu Wa Ta'ala*.

Medan, 25 Maret 2025



Rodiatul Husna Br Sitepu

211401038

ABSTRAK

Keamanan informasi menjadi isu krusial di era digital, terutama dalam pengiriman dan penyimpanan data sensitif. Penelitian ini mengimplementasikan skema *signcryption* dengan pendekatan *sign-then-encrypt* untuk menjamin kerahasiaan, autentikasi, integritas, dan non-repudiation pesan. Skema ini menggunakan algoritma XRSA untuk proses enkripsi dan dekripsi, serta tanda tangan digital Nagaty untuk tanda tangan dan verifikasi. Algoritma XRSA memanfaatkan empat bilangan prima besar dan operasi XOR dalam pembangkitan kunci, sedangkan Nagaty didasarkan pada kompleksitas perhitungan jumlah parsial ke-n dari deret integer tak hingga. Sistem dikembangkan menggunakan Python dan dirancang untuk mengamankan *file* teks berformat *.docx. Pengujian dilakukan pada *file* teks dengan berbagai panjang karakter guna menganalisis performa sistem. Hasil penelitian menunjukkan bahwa waktu pemrosesan meningkat seiring dengan bertambahnya panjang pesan. Sistem juga berhasil melakukan proses *signcryption* dan mengembalikan isi file teks ke bentuk semula dengan verifikasi yang *valid*. Selain itu, sistem terbukti tahan terhadap serangan faktorisasi *brute-force*, di mana kunci berukuran 35-bit membutuhkan waktu lebih dari satu jam untuk dipecahkan.

Kata Kunci: Keamanan informasi, *signcryption*, Algoritma XRSA, Skema Nagaty, Enkripsi, Dekripsi, Tanda Tangan Digital, Verifikasi.

SIGNCRYPTION CRYPTOSYSTEM SCHEME WITH XRSA PUBLIC KEY ENCRYPTION ALGORITHM AND NAGATY DIGITAL SIGNATURE

ABSTRACT

*Information security is a crucial issue in the digital era, especially in the transmission and storage of sensitive data. This research implements a signcryption scheme with a sign-then-encrypt approach to ensure confidentiality, authentication, integrity, and non-repudiation of messages. The scheme uses the XRSA algorithm for encryption and decryption processes, as well as Nagaty digital signatures for signature and verification. The XRSA algorithm utilizes four large primes and the XOR operation in key generation, while Nagaty is based on the complexity of calculating the n th partial sum of an infinite integer sequence. The system was developed using Python and designed to secure text files in *.docx format. Tests were conducted on text files with various character lengths to analyze the performance of the system. The results show that the processing time increases as the message length increases. The system also successfully performs the signcryption process and restores the contents of the text file to its original form with valid verification. In addition, the system proved to be resistant to a brute-force factorization attack, where a 35-bit key took more than an hour to crack.*

Keywords: *Information security, signcryption, XRSA Algorithm, Nagaty Scheme, Encryption, Decryption, Digital Signature, Verification.*

DAFTAR ISI

PERSETUJUAN	i
PERNYATAAN	ii
PENGHARGAAN	iii
ABSTRAK	v
ABSTRACT	vi
DAFTAR ISI	vii
DAFTAR TABEL	x
DAFTAR GAMBAR	xi
BAB I PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	3
1.3. Batasan Masalah	3
1.4. Tujuan Penelitian	4
1.5. Manfaat Penelitian	4
1.6. Metodologi Penelitian	4
1.7. Sistematika Penulisan	5
BAB II LANDASAN TEORI	6
2.1. Kriptografi	6
2.1.1. Terminologi Kriptografi	6
2.1.2. Jenis – Jenis Kriptografi	7
2.2. Dasar Matematika Kriptografi	8
2.2.1. Bilangan Prima	8
2.2.2. Prime Generator	9
2.2.3. Operasi Modulo	9
2.2.4. Greatest Common Divisor (GCD)	10
2.2.5. Invers Modulo	10
2.2.6. Persamaan Diophantine Linear	10
2.2.7. Algoritma Extended Euclidean	11
2.2.8. Modulo Exponensial (Square and Multiply)	12
2.3. Algoritma Digital Signature Kunci Publik	13

2.4.	<i>Metode Signcryption</i>	14
2.5.	<i>Digitized Signature dan Digital Signature</i>	14
2.5.1	<i>Digitized Signature</i>	14
2.5.2	<i>Digital Signature</i>	14
2.6.	<i>Algoritma RSA Berdasarkan Operasi XOR: XRSA</i>	14
2.6.1	<i>Proses Kerja Algoritma XRSA</i>	15
2.6.2	<i>Contoh Perhitungan algoritma XRSA</i>	16
2.7.	<i>Signature Scheme Based on Numerical Series: Nagaty Digital Signature</i>	20
2.7.1	<i>Proses Kerja Nagaty Digital Signature</i>	21
2.7.2	<i>Contoh Perhitungan Nagaty Digital Signature</i>	22
2.8.	<i>Teknik Signcryption dengan Algoritma XRSA dan Nagaty Digital Signature</i>	23
2.9.	<i>Faktorisasi Brute-Force</i>	24
2.10.	<i>Penelitian Relevan</i>	25
BAB III ANALISIS DAN PERANCANGAN SISTEM		26
3.1.	<i>Analisis Sistem</i>	26
3.1.1.	<i>Analisis Masalah</i>	26
3.1.2.	<i>Analisis Kebutuhan</i>	28
3.1.3.	<i>Analisis Proses</i>	29
3.2.	<i>Pemodelan Sistem</i>	29
3.2.1.	<i>Diagram Umum</i>	29
3.2.2.	<i>Use Case Diagram</i>	31
3.2.3.	<i>Activity Diagram</i>	32
3.3.	<i>Flowchart (Diagram Alir)</i>	34
3.3.1.	<i>Flowchart Pembangkitan Bilangan Prima Fermat's Little Theorem</i>	35
3.3.2.	<i>Flowchart Pembangkit Kunci Algoritma XRSA</i>	36
3.3.3.	<i>Flowchart Enkripsi XRSA</i>	36
3.3.4.	<i>Flowchart Dekripsi XRSA</i>	37
3.3.5.	<i>Flowchart Pembangkit Kunci Skema Nagaty</i>	37
3.3.6.	<i>Flowchart Tanda Tangan Nagaty</i>	38
3.3.7.	<i>Flowchart Verifikasi Nagaty</i>	38
3.4.	<i>Perancangan Antarmuka (Interface)</i>	39
3.4.1.	<i>Form Tampilan Awal (Home)</i>	39
3.4.2.	<i>Form Pembangkitan Kunci Nagaty</i>	39
3.4.3.	<i>Form Pembangkitan Kunci XRSA</i>	40

3.4.4.	<i>Form Tanda Tangan dan Enkripsi</i>	41
3.4.5.	<i>Form Dekripsi dan Verifikasi</i>	42
3.4.6.	<i>Form Faktorisasi Brute-Force</i>	42
BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM		44
4.1.	Implementasi Sistem	44
4.1.1.	Laman <i>Home</i>	44
4.1.2.	Laman Pembangkitan Kunci Nagaty	45
4.1.3.	Laman Pembangkitan Kunci XRSA	45
4.1.4.	Laman Tanda Tangan dan Enkripsi	46
4.1.5.	Laman Dekripsi dan Verifikasi	46
4.1.6.	Laman Faktorisasi <i>Brute-Force</i>	47
4.2.	Pengujian Sistem	47
4.2.1.	Pengujian Pembangkitan Kunci	47
4.2.2.	Pengujian Tanda Tangan dan Enkripsi	48
4.2.3.	Pengujian Dekripsi dan Verifikasi	49
4.2.4.	Pengujian Faktorisasi <i>Brute-Force</i>	49
4.2.5.	Pengujian <i>Real Running Time</i> Sistem.....	50
4.2.6.	Analisis Penyerangan Kunci dengan <i>Brute-Force</i>	53
BAB V KESIMPULAN DAN SARAN		56
5.1.	Kesimpulan	56
5.2.	Saran	57
DAFTAR PUSTAKA		58

DAFTAR TABEL

Tabel 2.1 Algoritma <i>Extended Euclidean</i>	11
Tabel 2.2 Perhitungan dengan Algoritma <i>Extended Euclidean</i>	12
Tabel 4.1 Pengujian Waktu Pembangkitan Kunci Nagaty.....	50
Tabel 4.2 Pengujian Waktu Pembangkitan Kunci XRSA.....	51
Tabel 4.3 Pengujian Waktu Tanda Tangan dan Enkripsi	52
Tabel 4.4 Pengujian Waktu Dekripsi dan Verifikasi.....	53
Tabel 4.5 Analisis Faktorisasi <i>Brute Force</i>	54



DAFTAR GAMBAR

Gambar 2.1 Proses Enkripsi dan Dekripsi.....	6
Gambar 2.2 Proses Kriptografi Kunci Simetris.....	7
Gambar 2.3 Proses Kriptografi Kunci Asimetris.....	8
Gambar 2.4 Proses <i>Digital Signature</i> Kunci Publik.....	13
Gambar 3.1 Diagram Umum Pembangkitan Kunci <i>Sender</i>	30
Gambar 3.2 Diagram Umum Pembangkitan Kunci <i>Recipient</i>	30
Gambar 3.3 Diagram Umum <i>Signcryption</i>	30
Gambar 3.4 <i>Use Case Diagram</i>	31
Gambar 3.5 <i>Activity Diagram</i> Pembangkit Kunci XRSA Penerima	32
Gambar 3.6 <i>Activity Diagram</i> Pembangkit Kunci Nagaty Pengirim	33
Gambar 3.7 <i>Activity Diagram</i> Proses Tanda Tangan dan Enkripsi.....	33
Gambar 3.8 <i>Activity Diagram</i> Proses Dekripsi dan Verifikasi	34
Gambar 3.9 <i>Flowchart</i> Pembangkit Bilangan Prima	35
Gambar 3.10 <i>Flowchart</i> Pembangkit Kunci pada Algoritma XRSA	36
Gambar 3.11 <i>Flowchart</i> Enkripsi Algoritma XRSA	36
Gambar 3.12 <i>Flowchart</i> Dekripsi Algoritma XRSA.....	37
Gambar 3.13 <i>Flowchart</i> Pembangkit Kunci pada Skema Nagaty.....	37
Gambar 3.14 <i>Flowchart</i> Tanda Tangan Skema Nagaty.....	38
Gambar 3.15 <i>Flowchart</i> Verifikasi Skema Nagaty.....	38
Gambar 3.16 Rancangan <i>Form</i> Tampilan Awal.....	39
Gambar 3.17 Rancangan <i>Form</i> Pembangkitan Kunci Nagaty	40
Gambar 3.18 Rancangan <i>Form</i> Pembangkitan Kunci XRSA	40
Gambar 3.19 Rancangan <i>Form</i> Tanda Tangan dan Enkripsi.....	41
Gambar 3.20 Rancangan <i>Form</i> Dekripsi dan Verifikasi.....	42
Gambar 3.21 Rancangan <i>Form</i> Faktorisasi Brute-Force.....	43
Gambar 4.1 Laman <i>Home</i>	44
Gambar 4.2 Laman Pembangkitan Kunci Nagaty	45
Gambar 4.3 Laman Pembangkitan Kunci XRSA.....	45

Gambar 4.4 Laman Tanda Tangan dan Enkripsi	46
Gambar 4.5 Laman Dekripsi dan Verifikasi	46
Gambar 4.6 Laman Faktorisasi <i>Brute-Force</i>	47
Gambar 4.7 Pengujian Pembangkitan Kunci Nagaty	47
Gambar 4.8 Pengujian Pembangkitan Kunci XRSA	48
Gambar 4.9 Pengujian Tanda Tangan dan Enkripsi	48
Gambar 4.10 Dokumen Hasil Tanda Tangan dan Enkripsi	48
Gambar 4.11 Pengujian Dekripsi dan Verifikasi	49
Gambar 4.12 Dokumen Hasil Dekripsi dan Verifikasi	49
Gambar 4.13 Pengujian Faktorisasi <i>Brute-Force</i>	50
Gambar 4.14 Grafik Pengujian Pembangkitan Kunci Nagaty	51
Gambar 4.15 Grafik Pengujian Pembangkitan Kunci XRSA	51
Gambar 4.16 Grafik Pengujian Waktu Tanda Tangan dan Enkripsi	52
Gambar 4.17 Grafik Pengujian Waktu Dekripsi dan Verifikasi	53
Gambar 4.18 Grafik Faktorisasi <i>Brute-Force</i>	54



BAB I

PENDAHULUAN

1.1. Latar Belakang

Keamanan informasi merupakan salah satu isu yang paling krusial saat ini. Semakin banyak data sensitif yang disimpan dan dikirim secara digital, sehingga ancaman terhadap keamanan informasi juga semakin kompleks dan canggih. Data pribadi, informasi bisnis, dan aset digital lainnya menjadi target serangan siber yang dapat menimbulkan kerugian finansial yang besar dan merusak reputasi.

Kerahasiaan data sering kali dijamin melalui penggunaan enkripsi, yang mengubah pesan menjadi format yang tidak dapat dibaca oleh pihak yang tidak berwenang. Sementara itu, tanda tangan digital digunakan untuk memastikan bahwa pesan yang diterima berasal dari sumber yang sah dan tidak mengalami perubahan selama pengiriman. Kedua proses ini, yaitu enkripsi dan penandatanganan digital, biasanya dilakukan secara terpisah dalam metode konvensional. Namun, pendekatan ini memiliki kelemahan dalam hal efisiensi, baik dari segi komputasi maupun komunikasi (Zhang et al., 2022).

Untuk mengatasi ancaman keamanan informasi, berbagai teknik kriptografi telah dikembangkan. Salah satunya adalah teknik *signcryption*. Istilah *signcryption* pertama kali dikenalkan oleh Zheng pada tahun 1997 dan disinggung kembali oleh Jee Hea An, Yevgeniy Dodis, dan Tal Rabin pada tahun 2002 dengan tujuan meningkatkan efisiensi *signcryption*. Penerapan enkripsi dan tanda tangan digital secara terpisah memerlukan operasi kriptografi yang terpisah, sehingga besar peluang terjadinya serangan diantara tahap enkripsi dan tanda tangan digital. Oleh karena itu, penelitian ini mengandalkan metode *signcryption* dimana proses enkripsi dan tanda tangan digital digabungkan menjadi satu langkah, dengan tujuan menjaga kerahasiaan dan autentikasi dalam konfigurasi kunci publik (Ginting et al., 2024).

Algoritma RSA adalah salah satu skema kriptografi yang populer hingga saat ini, dengan penggunaan dua bilangan prima besar dalam prosesnya. Namun algoritma RSA relatif lambat dalam pembangkitan kunci serta rentan terhadap serangan faktorisasi, dengan kemajuan teknologi komputasi saat ini, termasuk komputasi kuantum, sehingga faktorisasi bilangan prima besar yang digunakan dalam algoritma RSA menjadi lebih mudah dipecahkan/diserang (Imam et al., 2022).

Pada penelitian tentang algoritma RSA yang dilakukan oleh (Imam et al., 2022) dengan judul “*An Effective and enhanced RSA based Public Key Encryption Scheme (XRSA)*” dimana penelitian ini membahas upaya peningkatan keamanan dari algoritma RSA. Penelitian ini menggunakan empat bilangan prima besar pada proses pembangkitan *public key* dan *private key* serta menerapkan operasi XOR dalam proses pembangkitan kunci, enkripsi, dan dekripsi untuk mencapai kompleksitas algoritma yang lebih tinggi. Berdasarkan penelitian tersebut terbukti algoritma XRSA lebih tahan terhadap serangan *brute-force*, karena memerlukan waktu dan daya komputasi yang jauh lebih besar untuk memecahkan enkripsi. Namun, XRSA hanya berfokus pada enkripsi dan dekripsi tanpa fitur otentikasi pengirim, sehingga pihak ketiga dapat memalsukan data terenkripsi dan penerima tidak memiliki cara untuk memverifikasi keaslian pengirim.

Pada penelitian tentang Nagaty digital signature yang dilakukan oleh (Khaled A. Nagaty, 2020) dengan judul “*A Public Key Cryptosystem and Signature Scheme based on Numerical Series*” mengusulkan skema tanda tangan digital yang mengandalkan kesulitan dalam menghitung jumlah parsial ke- n (S_n) dari deret integer tak hingga untuk membangkitkan *public key* dan *private key*. Nilai n diambil secara acak dan bernilai besar, sehingga suku terakhir n akan selalu jauh lebih besar dari suku pertama. Penelitian ini mengusulkan proses *signing* dan *verification sign* yang tidak memerlukan perhitungan eksponensial, namun memberikan tingkat keamanan yang lebih tinggi. Berbeda dari kriptografi RSA maupun *Elliptic Curve* yang terlalu lambat secara komputasi, skema ini lebih mudah diimplementasikan dan lebih cepat secara komputasi.

Namun, skema tanda tangan digital Nagaty merupakan pendekatan baru, yang berarti belum diuji secara luas dalam berbagai skenario serangan kriptografi seperti serangan adaptif terhadap skema tanda tangan, serangan pemalsuan tanda tangan

(*forgery attack*), dan serangan berbasis kuantum terhadap skema numerical series, sehingga dibutuhkan penggabungan dengan algoritma yang memiliki tingkat keamanan tinggi dan sudah teruji untuk memastikan perlindungan pesan yang lebih optimal.

Penelitian ini bertujuan untuk merancang sistem perlindungan file teks dari akses pihak yang tidak berwenang menggunakan teknik *signcryption*, dengan memanfaatkan algoritma XRSA dalam proses enkripsi dan dekripsi serta skema tanda tangan digital Nagaty dalam proses tanda tangan dan verifikasi. Penggabungan kedua algoritma ini dilakukan untuk mengatasi kelemahan masing-masing, di mana XRSA tidak memiliki mekanisme otentikasi pengirim, sedangkan skema tanda tangan digital Nagaty merupakan pendekatan baru yang belum diuji secara luas dalam berbagai skenario serangan kriptografi. Dengan demikian, penelitian ini diharapkan dapat menghasilkan sistem *signcryption* yang lebih aman dan lebih terpercaya, serta berkontribusi dalam pengembangan metode perlindungan data di masa mendatang.

1.2. Rumusan Masalah

Berdasarkan latar belakang di atas, maka rumusan masalah dalam penelitian ini adalah dengan algoritma XRSA untuk enkripsi kunci publik dan skema Nagaty *digital signature* dapat menjaga kerahasiaan dan keaslian informasi berbentuk pesan teks, sehingga pesan tidak dapat dimanipulasi dan dibaca oleh pihak yang tidak sah.

1.3. Batasan Masalah

Beberapa batasan masalah dalam penelitian ini sebagai berikut.

1. Karakter dalam *plaintext* dan *ciphertext* didasarkan pada kode ASCII yang berupa 256 karakter.
2. Algoritma yang diterapkan pada penelitian ini adalah algoritma XRSA untuk proses enkripsi dan dekripsi serta algoritma Nagaty untuk proses tanda tangan dan verifikasi.
3. Metode *signcryption* yang digunakan adalah *sign-then-encrypt*.
4. Penelitian ini tidak menggunakan fungsi *hash* dalam proses tanda tangan.
5. Program dirancang dengan menggunakan bahasa python.
6. Jenis data yang digunakan dalam penelitian ini adalah format data teks (*.docx) dan tidak mencakup data selain teks, seperti gambar dan tabel.

7. Pembangkit bilangan prima dalam penelitian ini menggunakan uji primalitas Fermat.
8. Penelitian ini melakukan pengujian *brute-force* terhadap kunci publik XRSA, yaitu nilai n .

1.4. Tujuan Penelitian

Penelitian ini bertujuan untuk merancang sebuah sistem perlindungan *file* teks dari akses pihak yang tidak berwenang menggunakan teknik *signcryption* yang memanfaatkan algoritma XRSA dalam proses enkripsi dan dekripsi serta skema tanda tangan digital Nagaty dalam proses tanda tangan dan verifikasi.

1.5. Manfaat Penelitian

Manfaat dari penelitian ini adalah merancang sistem yang mampu mengamankan *file* teks menggunakan teknik *signcryption* yang memanfaatkan algoritma XRSA dan skema tanda tangan digital Nagaty. Penelitian ini diharapkan dapat menjadi landasan bagi penelitian lebih lanjut mengenai *signcryption*, khususnya dalam penerapan algoritma XRSA dan skema Nagaty, sehingga membuka peluang pengembangan skema kriptografi yang lebih canggih dan adaptif.

1.6. Metodologi 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 Algoritma XRSA dan Nagaty *Digital Signature*.
2. Perancangan Sistem: Berdasarkan ruang lingkup penelitian, penulis melakukan analisis masalah menggunakan metode *5-Whys* untuk mengidentifikasi kebutuhan penelitian. Hasil analisis ini digunakan sebagai dasar dalam merancang *use case*, *activity diagram*, diagram umum, dan *flowchart*.
3. Implementasi Sistem: Pada tahap ini, membuat sebuah sistem dengan menggunakan bahasa pemrograman Python sesuai dengan diagram alir yang telah dirancang.

4. Pengujian Sistem: Pada tahap ini, sistem yang telah dirancang dilakukan uji coba untuk melakukan *Signcryption* dengan menggunakan Algoritma XRSA *Public Key Encryption* dan Nagaty *Digital Signature*.
5. Dokumentasi: Pada tahap ini, penelitian yang telah dilakukan, didokumentasikan mulai dari tahap analisa sampai kepada pengujian dalam bentuk skripsi.

1.7. Sistematika Penulisan

Struktur skripsi yang diterapkan dalam penulisan penelitian ini terdiri dari lima bab utama, yaitu:

BAB 1 PENDAHULUAN

Bab ini membahas aspek-aspek utama penelitian, termasuk latar belakang, rumusan masalah, batasan penelitian, tujuan, manfaat, serta penelitian terkait yang dijadikan rujukan. Selain itu, dijelaskan juga metodologi yang digunakan serta struktur penulisan skripsi yang akan diterapkan dalam laporan penelitian ini.

BAB 2 LANDASAN TEORI

Bab ini membahas kriptografi beserta komponennya, serta algoritma yang digunakan dalam teknik *signcryption*, yaitu Algoritma XRSA dan skema tanda tangan digital Nagaty.

BAB 3 ANALISIS DAN PERANCANGAN

Bab ini menjelaskan alur dan proses kerja teknik *signcryption* menggunakan Algoritma XRSA dan skema tanda tangan digital Nagaty.

BAB 4 IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini meliputi penerapan sistem yang telah dibangun serta hasil uji coba sistem menggunakan data teks yang dimasukkan ke dalam sistem.

BAB 5 KESIMPULAN DAN SARAN

Bab ini memuat kesimpulan yang diambil dari pembahasan tiap bab, serta rekomendasi yang diberikan oleh peneliti sebagai kontribusi untuk penelitian di masa mendatang.

BAB II

LANDASAN TEORI

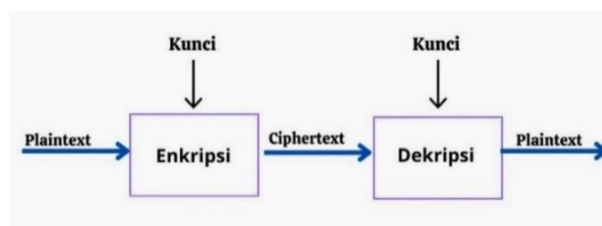
2.1. Kriptografi

Kriptografi adalah studi tentang strategi komunikasi aman yang hanya memungkinkan pesan dapat dilihat oleh pihak pengirim dan penerima yang sah. Teknik kriptografi digunakan untuk menjaga kerahasiaan, autentikasi dan integritas pesan dari pihak yang tidak berwenang (Imam *et al.*, 2022).

2.1.1. Terminologi Kriptografi

Beberapa terminologi dalam kriptografi adalah sebagai berikut:

1. *Plaintext* adalah pesan yang dapat dibaca dan memiliki makna.
2. *Ciphertext* adalah pesan yang telah dienkripsi, sehingga tidak dapat dibaca dan tidak memiliki makna.
3. *Sender* adalah pihak yang mengirimkan pesan.
4. *Receiver* adalah pihak yang menerima pesan.
5. Enkripsi adalah proses mengubah pesan yang memiliki makna (*plaintext*) menjadi pesan yang tidak memiliki makna (*ciphertext*).
6. Dekripsi adalah proses mengubah pesan yang tidak memiliki makna (*ciphertext*) menjadi pesan yang memiliki makna (*plaintext*).
7. Kunci adalah parameter yang digunakan dalam proses enkripsi dan dekripsi.



Gambar 2.1 Proses Enkripsi dan Dekripsi

Gambar 2.1 memperlihatkan *plaintext* yang melewati proses enkripsi dengan menggunakan sebuah kunci akan menghasilkan *ciphertext*, lalu *ciphertext* yang

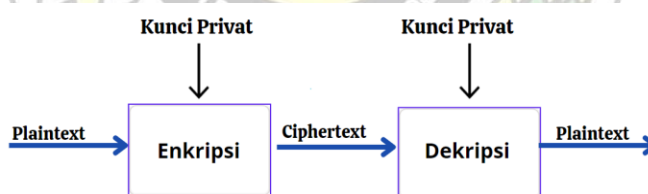
melewati proses dekripsi menggunakan sebuah kunci akan menghasilkan *plaintext* yang semula.

2.1.2. Jenis – Jenis Kriptografi

Jenis – jenis kriptografi terbagi menjadi dua berdasarkan kunci yang digunakan, yaitu:

1. Kriptografi Kunci Simetris

Cara kerja kriptografi kunci simetris adalah penggunaan kunci yang sama dalam proses enkripsi dan dekripsi pesan. Keunggulan dari kriptografi kunci simetris adalah kecepatan waktunya dalam melakukan proses enkripsi terhadap *plaintext* dan proses dekripsi terhadap *ciphertext* (M. Budiman *et al.*, 2020). Kelemahan dari kriptografi kunci simetris sendiri terletak pada penggunaan kunci yang sama dalam proses enkripsi dan dekripsi, dikarenakan kunci yang sama tersebut harus dikirimkan kepada penerima pesan agar penerima pesan dapat melakukan proses dekripsi. Hal tersebut menjadi masalah ketika sulitnya terbangun jalur pengiriman yang aman (M. A. Budiman & Rachmawati, 2020). Sehingga, jika kunci tersebar kepada pihak yang tidak sah maka mereka dapat melakukan proses dekripsi. Contoh algoritma kriptografi kunci simetris adalah *Advanced Encryption Standard* (AES), *Data Encryption Standard* (DES), dan *Triple DES* (3DES).



Gambar 2.2 Proses Kriptografi Kunci Simetris

2. Kriptografi Kunci Publik

Cara kerja kriptografi kunci publik adalah penggunaan kunci yang berbeda pada proses enkripsi dan dekripsinya. Seorang penerima pesan membangkitkan dua kunci, yaitu kunci pribadi dan kunci publik. Kunci pribadi akan disimpan oleh penerima pesan, sedangkan kunci publik dibagikan agar dapat diketahui oleh semua pihak. Menggunakan kunci publik pengirim pesan melakukan proses enkripsi (M. A. Budiman & Rachmawati, 2020). Penerima pesan melakukan proses dekripsi menggunakan kunci pribadi miliknya.

Kriptografi kunci publik memiliki tingkat keamanan yang tinggi. Namun, metode ini kurang efisien untuk mengenkripsi data dalam jumlah besar karena bergantung pada bilangan prima yang besar. Hal ini membuat proses enkripsi dan dekripsi lebih lambat serta memerlukan perhitungan yang kompleks dan memakan waktu. Oleh karena itu, penggunaannya lebih sering terbatas pada pengamanan informasi singkat, seperti kunci sesi (Smart, 2016).

Untuk mengatasi keterbatasan tersebut, kriptografi kunci publik sering dikombinasikan dengan kriptografi kunci simetris dalam pendekatan yang dikenal sebagai *hybrid cryptosystem*. Dalam skema ini, enkripsi kunci publik digunakan untuk mendistribusikan kunci sesi secara aman, sementara data utama dienkripsi menggunakan kriptografi kunci simetris yang lebih cepat dan efisien (Smart, 2016).

Beberapa algoritma kriptografi kunci publik yang umum digunakan antara lain RSA, ElGamal, dan *Elliptic Curve Cryptography* (ECC).



Gambar 2.3 Proses Kriptografi Kunci Asimetris

Pada Gambar 2.3 dapat dilihat bahwa *plaintext* di enkripsi menggunakan kunci publik menjadi *ciphertext*, lalu *ciphertext* di dekripsi menggunakan kunci privat untuk menghasilkan *plaintext* yang semula.

2.2. Dasar Matematika Kriptografi

Berikut beberapa dasar – dasar matematika yang digunakan dalam proses pengerjaan algoritma XRSA dan Nagaty.

2.2.1. Bilangan Prima

Bilangan prima adalah bilangan bulat positif yang tepat hanya memiliki dua faktor pembagi yaitu bilangan satu dan bilangan itu sendiri. Contoh dari bilangan prima adalah 2, 3, 5, 7, 11, 13, 17 dan seterusnya.

2.2.2. Prime Generator

Prime generator adalah sebuah metode untuk membuktikan apakah suatu bilangan yang diinputkan merupakan bilangan prima atau tidak. *Fermat's Little Theorem* adalah salah satu teori yang mendasari berbagai macam teorema lain dalam teori bilangan. Dalam teorema ini menyatakan, bila terdapat bilangan bulat positif p dan berlaku $a^{p-1} \equiv 1 \pmod{p}$ dengan syarat $1 < a < p$ maka bilangan p merupakan bilangan prima (Daep & Gorkin, 2011).

Namun, dalam penerapan teorema Fermat, sering terjadi *Fermat's liar* (*false prime*), yaitu kasus di mana bilangan komposit keliru teridentifikasi sebagai bilangan prima. Untuk mengurangi kemungkinan kesalahan ini, penelitian ini menerapkan uji Fermat sebanyak jumlah digit p dikali 3 sebagai langkah verifikasi tambahan.

Contoh: Apakah 17 prima?

$p = 17$, untuk $1 < a < 17$

- | | |
|---|---|
| - $a = 2$, cek $2^{17-1} \bmod 17 = 1$ | - $a = 10$, cek $10^{17-1} \bmod 17 = 1$ |
| - $a = 3$, cek $3^{17-1} \bmod 17 = 1$ | - $a = 11$, cek $11^{17-1} \bmod 17 = 1$ |
| - $a = 4$, cek $4^{17-1} \bmod 17 = 1$ | - $a = 12$, cek $12^{17-1} \bmod 17 = 1$ |
| - $a = 5$, cek $5^{17-1} \bmod 17 = 1$ | - $a = 13$, cek $13^{17-1} \bmod 17 = 1$ |
| - $a = 6$, cek $6^{17-1} \bmod 17 = 1$ | - $a = 14$, cek $14^{17-1} \bmod 17 = 1$ |
| - $a = 7$, cek $7^{17-1} \bmod 17 = 1$ | - $a = 15$, cek $15^{17-1} \bmod 17 = 1$ |
| - $a = 8$, cek $8^{17-1} \bmod 17 = 1$ | - $a = 16$, cek $16^{17-1} \bmod 17 = 1$ |
| - $a = 9$, cek $9^{17-1} \bmod 17 = 1$ | |

Berdasarkan iterasi disimpulkan bahwa 17 adalah prima.

2.2.3. Operasi Modulo

Operasi modulo merupakan operasi matematika yang menghasilkan sisa pembagian dari suatu bilangan terhadap bilangan lainnya. Modulo biasanya disingkat menjadi mod.

Contoh: $-7 \bmod 3 = 1$

- $8 \bmod 3 = 2$

- $9 \bmod 3 = 0$

Keterangan: $-7 \bmod 3$ hasilnya 1, karena hasil dari 7 bagi 3 adalah 2 dan sisanya 1.

- $8 \bmod 3$ hasilnya 2, karena hasil dari 8 bagi 3 adalah 2 dan sisanya 2.

- $9 \bmod 3$ hasilnya 0, karena hasil dari 9 bagi 3 adalah 3 dan sisanya 0.

Sifat modulo

- Jika $m < n$, dimana m dan n adalah komponen dari bilangan bulat positif kecuali nol, maka $m \bmod n = m$.
- Jika $m \bmod n = r$, maka $(m \pm kn) \bmod n = r$.

2.2.4. Greatest Common Divisor (GCD)

Greatest Common Divisor (GCD) adalah faktor persekutuan terbesar, GCD dari dua bilangan bulat adalah bilangan bulat terbesar yang sama – sama membagi habis kedua bilangan tersebut.

Algoritma *Euclidean* adalah algoritma yang digunakan untuk mencari GCD dari dua bilangan bulat. Algoritma *Euclidean* pertama kali ditemukan oleh seorang matematikawan Yunani, yaitu Euclid yang menuliskannya dalam buku berjudul *Element* (Mahmudah & Triyana, 2018). Pencarian GCD dengan algoritma *Euclidean* menggunakan operasi modulo, berikut contoh dari perhitungan GCD: Berapakah GCD dari 72 dan 28?

$$72 \bmod 28 = 16$$

$$28 \bmod 16 = 12$$

$$16 \bmod 12 = 4$$

$$12 \bmod 4 = 0, \text{ Maka GCD dari 72 dan 28 adalah 4}$$

2.2.5. Invers Modulo

Invers modulo adalah sebuah konsep matematika untuk mencari angka yang jika dikalikan dengan angka lain dan dibagi modulo nilai tertentu, hasilnya 1. Contoh notasinya, m^{-1} dikatakan *invers* dari $m \pmod{n}$ apabila $m^{-1} \cdot m \equiv 1 \pmod{n}$. Syarat untuk $m \pmod{n}$ punya *invers* adalah $\text{GCD}(m, n) = 1$, dimana $n > 1$.

2.2.6. Persamaan Diophantine Linear

Bentuk umum dari persamaan diophantine linear adalah $ax + by = c$, dimana x, y, a, b, c elemen dari bilangan bulat. Nilai x dan y adalah peubah yang ingin dicari.

2.2.7. Algoritma Extended Euclidean

Algoritma *Extended Euclidean* digunakan untuk mencari solusi dari persamaan diophantine linear dengan identitas Bezout. Dimana identitas Bezout digunakan untuk mengetahui x dan y adalah bilangan bulat, dengan rumus:

$$c \equiv \text{GCD}(a, b), \text{ maka } ax + by = c$$

Berikut algoritma *Extended Euclidean* (Arteta et al., 2012):

1. Tetapkan nilai awal, dengan syarat:
 - Jika $a > b$ maka $x_0 = 1$ dan $y_0 = 0$
 - Jika $a < b$ maka $x_0 = 0$ dan $y_0 = 1$
2. Lakukan iterasi, $i = 1$, dengan syarat:
 - Jika $d_i = 0$ maka kembali ke nilai d_{i-1} , selesai.
 - Jika $d_i > 0$ maka bagi d_{i-1} dengan d_i , sehingga didapatkan k_i .
3. Hitung nilai d_{i+1} , x_{i+1} , y_{i+1} , dengan rumus:
 - $d_{i+1} = d_{i-1} - k_i d_i$
 - $x_{i+1} = x_{i-1} - k_i x_i$
 - $y_{i+1} = y_{i-1} - k_i y_i$
4. Lakukan iterasi $i + 1$ dan kembali ke langkah kedua.
5. Nilai d_n akan dapat sebelum hasilnya nol terakhir.

$$\text{GCD}(a, b) = d_n = ax_n + by_n$$

Dari algoritma di atas, didapatkan tabel sebagai berikut:

Tabel 2.1 Algoritma *Extended Euclidean*

$x_0 = 1$	$y_0 = 0$	$d_0 = a$	
$x_1 = 0$	$y_1 = 1$	$d_1 = b$	k_1
\vdots	\vdots	\vdots	\vdots
x_{n-1}	y_{n-1}	d_{n-1}	k_{n-1}
x_n	y_n	d_n	k_n
		$d_{n+1} = 0$	

Contoh: Carilah nilai *invers* 2 (mod 9) menggunakan persamaan diophantine linear!

Penyelesaian:

1. Ubah 2 (mod 9) menjadi persamaan diophantine, $2x + 9y = c$.

$$c = \text{GCD}(2, 9) = 1, \text{ jadi}$$

$$2x + 9y = 1$$

2. Diketahui $a > b$, maka $x_0 = 1$ dan $y_0 = 0$

Tabel 2.2 Perhitungan dengan Algoritma *Extended Euclidean*

X	y	d	k
0	1	9	
1	0	2	4
-4	1	1	

Berdasarkan tabel 2.2, dapat dilihat bahwa iterasi berhenti karena nilai $d=1$ dan terlihat hasil akhir adalah $x = -4$ dan $y = 1$, maka:

$$x^{-1} = -4 \bmod 9$$

$$x^{-1} = 5$$

Maka nilai *invers* 2 (mod 9) adalah 5.

2.2.8. Modulo Exponensial (*Square and Multiply*)

Modulo eksponensial merupakan operasi matematika yang melibatkan pangkat dan sisa pembagian, dengan bentuk umum: $x^y \bmod n$

Keterangan: x adalah basis
 y adalah eksponen
 n adalah modular

Dalam menyelesaikan modulo eksponensial dapat digunakan algoritma *Square and Multiply*, dimana pada algoritma ini eksponen akan diubah menjadi biner untuk mempermudah perhitungan ketika nilai eksponennya besar (Negre & Plantard, 2017). Berikut ketentuan dari algoritma *Square and Multiply*:

1. Inisialisasi nilai x , y , dan n .
2. Inisiasi nilai $z = 1$.
3. Ubah eksponen menjadi bilangan biner.
4. Telusuri nilai eksponen yang dalam bentuk biner, dari kiri ke kanan.
5. Jika bertemu bit nol (0) maka update nilai z menjadi, $z = z^2 \bmod n$.
6. Jika bertemu bit satu (1) maka update nilai z menjadi, $z = x.z^2 \bmod n$.
7. Nilai z terakhir adalah hasil dari $x^y \bmod n$.

Contoh: Berapa hasil dari $3^{91} \bmod 13$

$x = 3$, $n = 13$, $z = 1$, $y = 91$ **biner:** 1011011

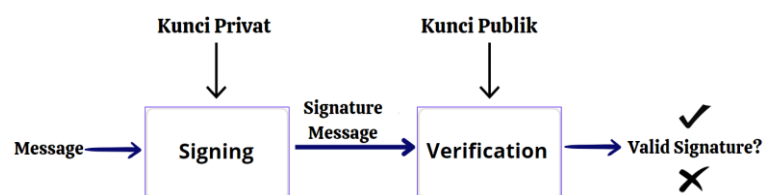
Iterasi

- | | |
|---|---|
| <p>1. Bertemu bit 1</p> $z = x * z^2 \bmod n$ $z = 3 * 1 \bmod 13 = 3$ <p>2. Bertemu bit 0</p> $z = z^2 \bmod n$ $z = 3^2 \bmod 13 = 9$ <p>3. Bertemu bit 1</p> $z = x * z^2 \bmod n$ $z = 3 * 9^2 \bmod 13 = 9$ <p>4. Bertemu bit 1</p> $z = x * z^2 \bmod n$ $z = 3 * 9^2 \bmod 13 = 9$ | <p>5. Bertemu bit 0</p> $z = z^2 \bmod n$ $z = 9^2 \bmod 13 = 3$ <p>6. Bertemu bit 1</p> $z = x * z^2 \bmod n$ $z = 3 * 3^2 \bmod 13 = 1$ <p>7. Bertemu bit 1</p> $z = x * z^2 \bmod n$ $z = 3 * 1^2 \bmod 13 = 3$ <p>Maka, hasil dari $3^{91} \bmod 13$ adalah 3</p> |
|---|---|

2.3. Algoritma *Digital Signature* Kunci Publik

Digital signature atau tanda tangan digital dibuat menggunakan bantuan metode kriptografi, yang bertujuan dapat menjaga keaslian dari pesan maupun informasi. Tanda tangan digital mengamankan dokumen elektronik dari perubahan dan pemalsuan. Tanda tangan digital memastikan bahwa pengirim dokumen tidak dapat menyangkal telah membuat dokumen tersebut, serta mencegah penerima maupun orang lain mengubah isi dari dokumen tersebut (Pooja & Yadav, 2018).

Proses *digital signature* kunci publik dimulai dari pembangkitan dua kunci oleh pengirim (*sender*), dimana kunci publik akan dipublikasikan dan kunci privat akan disimpan oleh pengirim (*sender*). Selanjutnya pengirim (*sender*) melakukan proses tanda tangan menggunakan kunci privatnya, lalu proses verifikasi dilakukan oleh penerima (*receiver*) menggunakan kunci publik yang sudah dipublikasikan oleh pengirim (*sender*) (Nagaty, 2020). Proses *digital signature* dapat dilihat pada Gambar 2.4.



Gambar 2.4 Proses *Digital Signature* Kunci Publik

2.4. Metode *Signcryption*

Metode *signcryption* adalah kriptografi kunci publik dengan penggabungan tanda tangan digital dan enkripsi, sehingga biaya komputasinya relatif lebih rendah daripada metode konvensional. Metode *signcryption* juga memiliki tingkat keamanan yang lebih tinggi dibandingkan metode konvensional. Penerapan enkripsi dan tanda tangan digital secara terpisah (konvensional) memerlukan operasi kriptografi yang terpisah, sehingga besar peluang terjadinya serangan diantara tahap enkripsi dan tanda tangan digital (Ginting et al., 2024).

2.5. *Digitized Signature* dan *Digital Signature*

2.5.1 *Digitized Signature*

Digitized signature adalah tanda tangan asli seseorang yang di-scan dan didigitalkan tanpa keamanan tambahan. Tanda tangan yang didigitalkan dapat dengan mudah diubah dan dipalsukan oleh orang lain karena tidak ada mekanisme untuk verifikasi keaslian dan integritasnya.

2.5.2 *Digital Signature*

Digital signature adalah metode autentikasi yang digunakan untuk memverifikasi bahwa dokumen yang diterima berasal dari pengirim yang sah dan isinya tidak diubah oleh pihak lain. *Digital signature* dibuat menggunakan metode kriptografi, dimana kunci privat pengirim digunakan untuk menciptakan tanda tangan digital pada dokumen. Fungsi utama *digital signature* meliputi autentikasi dan verifikasi integritas dokumen (Pooja & Yadav, 2018).

Digital signature juga memiliki sifat *non-repudiation*, yang berarti bahwa pengirim tidak dapat menyangkal telah membuat dan menandatangani dokumen tersebut, dan pesan yang ditandatangani secara digital tidak dapat diubah tanpa membatalkan tanda tangannya.

2.6. Algoritma RSA Berdasarkan Operasi XOR: XRSA

Algoritma XRSA diperkenalkan oleh R. Imam pada tahun 2022. Algoritma XRSA adalah algoritma pembangkitan kunci yang dimodifikasi dari algoritma RSA standar. Algoritma XRSA menggunakan empat bilangan prima acak besar untuk meningkatkan

keamanan dibandingkan dengan RSA standar yang menggunakan dua bilangan prima acak besar. Perhitungan pada eksponen kunci publik (E) juga tidak langsung, melainkan melibatkan operasi XOR dan variabel perantara yang meningkatkan kompleksitas dan keamanannya (Imam et al., 2022).

Pada algoritma XRSA juga diperlukan nilai dari E1 dan E2 yang digunakan untuk menentukan nilai E' dan D' dimana nilai tersebut akan digunakan untuk menentukan kunci publik dan kunci privat, sehingga meningkatkan waktu yang dibutuhkan untuk menyerang *ciphertext*. Keterlibatan empat bilangan prima besar dalam pembuatan kunci membuat musuh sulit untuk mengetahui nilai dari masing – masing bilangan tersebut. Serta musuh yang memiliki kunci publik juga tidak dapat menyimpulkan nilai kunci privat (D), karena kunci privat (D) dan kunci publik (E) tidak digunakan langsung dalam proses enkripsi dan dekripsi. Namun melibatkan operasi XOR dalam tahapannya, sehingga keamanannya meningkat.

2.6.1 Proses Kerja Algoritma XRSA

Proses kerja algoritma XRSA adalah sebagai berikut (Imam et al., 2022):

- **Pembangkitan Kunci**

1. Bangkitkan 4 bilangan prima yang besar secara acak, inisiasi sebagai p1, p2, p3, dan p4.
2. Hitung nilai $N = x * y$, $x = p1 * p2$ dan $y = p3 * p4$.
3. Hitung nilai *Euler-totient*
 - $\varphi(x) = (p1 - 1)(p2 - 1)$,
 - $\varphi(y) = (p3 - 1)(p4 - 1)$,
 - $\varphi(N) = \varphi(x) * \varphi(y)$
4. Ambil 2 bilangan ganjil dan inisiasikan sebagai E1 dan E2, dengan syarat:
 - $1 < E1 < \varphi(x)$, $1 < E2 < \varphi(y)$
 - Dan GCD dari $(E1 * E2, \varphi(N)) = 1$
5. Tentukan nilai E' dengan cara: $E' = (E1 * E2) \bmod N$
6. Hitung nilai D' dengan cara: $E' * D' \equiv 1 \bmod (\varphi(N))$
7. Hitung nilai E dengan cara: $E = E' \text{ XOR } N$
8. Hitung nilai D dengan cara: $D = D' \text{ XOR } N$
9. Berhasil membangkitkan kunci:
 - Kunci publik (E, N)
 - Kunci privat (p1, p2, p3, p4, dan $\varphi(N)$,D)

- **Proses Enkripsi**

Dapatkan kunci publik milik *recipient*, lalu hitung dengan cara:

$$E'' = E \text{ XOR } N$$

$$C = M^{E''} \bmod N$$

M: pesan (*plaintext*)

C: *ciphertext*

- **Proses Dekripsi**

Menggunakan kunci privat, lakukan dekripsi dengan cara:

$$D'' = D \text{ XOR } N$$

$$M = C^{D''} \bmod N$$

2.6.2 Contoh Perhitungan algoritma XRSA

- **Pembangkitan Kunci**

1. Bangkitkan 4 bilangan prima yang besar secara acak,

$$p_1 = 17 \quad p_3 = 11$$

$$p_2 = 13 \quad p_4 = 19$$

2. Hitung nilai $N = x * y$, $x = p_1 * p_2$ dan $y = p_3 * p_4$.

$$x = p_1 * p_2 = 17 * 13 = 221$$

$$y = p_3 * p_4 = 11 * 19 = 209$$

$$N = x * y = 221 * 209 = 46189$$

3. Hitung nilai *Euler-totient*

$$\begin{aligned} - \varphi(x) &= (p_1 - 1)(p_2 - 1) \\ &= (17 - 1)(13 - 1) = 192 \end{aligned}$$

$$\begin{aligned} - \varphi(y) &= (p_3 - 1)(p_4 - 1) \\ &= (11 - 1)(19 - 1) = 180 \end{aligned}$$

$$\begin{aligned} - \varphi(N) &= \varphi(x) * \varphi(y) \\ &= (192)(180) = 34560 \end{aligned}$$

4. Ambil 2 bilangan ganjil dan inisiasikan sebagai E1 dan E2, dengan syarat:

$$- 1 < E_1 < \varphi(x), \quad 1 < E_2 < \varphi(y)$$

$$- \text{Dan GCD dari } (E_1 * E_2, \varphi(N)) = 1$$

$$E_1 = 37 \quad \text{GCD dari } (37 * 13, 34560) = 1$$

$$E_2 = 13 \quad \text{GCD dari } (481, 34560) = 1 \text{ (TERBUKTI)}$$

Pembuktian:

$$\begin{aligned}
 & - 34560 \bmod 481 = 409 & - 49 \bmod 23 = 3 \\
 & - 481 \bmod 409 = 72 & - 23 \bmod 3 = 2 \\
 & - 409 \bmod 72 = 49 & - 3 \bmod 2 = 1 \\
 & - 72 \bmod 49 = 23 & - 2 \bmod 1 = 0
 \end{aligned}$$

5. Tentukan nilai E' dengan cara:
- $E' = (E_1 * E_2) \bmod N$
 - $E' = (37 * 13) \bmod 46189$
 - $E' = 481 \bmod 46189 = 481$
6. Hitung nilai D' dengan cara:
- $E' * D' \equiv 1 \bmod (\varphi(N))$
 - $481 * D' \equiv 1 \bmod (34560)$

Jadikan Persamaan Diophantine: $481x + 34560y = 1$

Selesaikan dengan algoritma *Extended Euclidean*

x	y	r	k
0	1	34560	
1	0	481	71
-71	1	409	1
72	-1	72	5
-431	6	49	1
503	-7	23	2
-1437	20	3	7
10552	-147	2	1
-11999	167	1	

Nilai $x = -11999$, maka: $D' \equiv -11999 \bmod 34560$

$$D' \equiv 22561 \bmod 34560$$

7. Hitung nilai E dengan cara

$$E = E' \text{ XOR } N$$

$$E = 481 \text{ XOR } 46189$$

$$\text{Biner } 481 = 0000000111100001$$

$$\text{Biner } 46189 = 1011010001101101$$

$$481 \text{ XOR } 46189 = 1011010110001100$$

$$E = 46476$$

8. Hitung nilai D dengan cara

$$D = D' \text{ XOR } N$$

$$D = 22561 \text{ XOR } 46189$$

$$\begin{aligned}
 \text{Biner } 22561 &= 0101100000100001 \\
 \text{Biner } 46189 &= 1011010001101101 \\
 15571 \text{ XOR } 46189 &= 1110110001001100 \\
 D &= 60492
 \end{aligned}$$

9. Berhasil membangkitkan kunci

- Kunci publik (E, N)
- Kunci privat (D, N)

• **Proses Enkripsi**

Dapatkan kunci publik milik *recipient*, lalu hitung dengan cara:

$$E'' = E \text{ XOR } N$$

$$C = M^{E''} \bmod N$$

M: pesan (*plaintext*)

C: *ciphertext*

Misal M = "B" – ASCII – 66

$$E'' = E \text{ XOR } N$$

$$E'' = 46476 \text{ XOR } 46189$$

$$\text{Biner } 46476 = 1011010110001100$$

$$\text{Biner } 46189 = 1011010001101101$$

$$46476 \text{ XOR } 46189 = 0000000111100001$$

$$E'' = 481$$

$$C = M^{E''} \bmod N, \quad C = 66^{481} \bmod 46189$$

Algoritma Square and Multiply

$$x = 66, n = 46189, z = 1, y = 481 \quad \text{biner: } 111100001$$

Iterasi

- | | |
|--|------------------------------------|
| 1. Bertemu bit 1 ($x * z^2 \bmod n$) | 4. Bertemu bit 1 |
| $z = 66 * 1^2 \bmod 46189$ | $z = 66 * 33957^2 \bmod 46189$ |
| $z = 66$ | $z = 16940$ |
| 2. Bertemu bit 1 | 5. Bertemu bit 0 ($z^2 \bmod n$) |
| $z = 66 * 66^2 \bmod 46189$ | $z = 16940^2 \bmod 46189$ |
| $z = 10362$ | $z = 37532$ |
| 3. Bertemu bit 1 | 6. Bertemu bit 0 |
| $z = 66 * 10362^2 \bmod 46189$ | $z = 37532^2 \bmod 46189$ |
| $z = 33957$ | $z = 25091$ |

- | | |
|---|--|
| <p>7. Bertemu bit 1
 $z = 25091^2 \bmod 46189$
 $z = 2211$</p> <p>8. Bertemu bit 0
 $z = 2211^2 \bmod 46189$
 $z = 38676$</p> | <p>9. Bertemu bit 0
 $z = 66 * 38676^2 \bmod 46189$
 $z = 7359$</p> <p>Maka hasil dari $66^{481} \bmod 46189$
 adalah 7359, C = 7359</p> |
|---|--|

• **Proses Dekripsi**

Menggunakan kunci privat, lakukan dekripsi dengan cara:

$$D'' = D \text{ XOR } N$$

$$D'' = 60492 \text{ XOR } 46189$$

$$\text{Biner } 60492 = 1110110001001100$$

$$\text{Biner } 46189 = 1011010001101101$$

$$60492 \text{ XOR } 46189 = 0101100000100001$$

$$D'' = 22561$$

$$M = C^{D''} \bmod N = 7359^{22561} \bmod 46189$$

Algoritma Square and Multiply

$$x = 7359, n = 46189, z = 1, y = 22561 \text{ biner: } 101100000100001$$

Iterasi

- | | |
|--|--|
| <p>1. Bertemu bit 1
 $z = 7359 * 1^2 \bmod 46189$
 $z = 7359$</p> <p>2. Bertemu bit 0
 $z = 7359^2 \bmod 46189$
 $z = 21373$</p> <p>3. Bertemu bit 1
 $z = 7359 * 21373^2 \bmod 46189$
 $z = 36960$</p> <p>4. Bertemu bit 1
 $z = 7359 * 36960^2 \bmod 46189$
 $z = 32241$</p> <p>5. Bertemu bit 0
 $z = 32241^2 \bmod 46189$
 $z = 44825$</p> | <p>6. Bertemu bit 0
 $z = 44825^2 \bmod 46189$
 $z = 12936$</p> <p>7. Bertemu bit 0
 $z = 12936^2 \bmod 46189$
 $z = 43538$</p> <p>8. Bertemu bit 0
 $z = 43538^2 \bmod 46189$
 $z = 7073$</p> <p>9. Bertemu bit 0
 $z = 7073^2 \bmod 46189$
 $z = 4642$</p> <p>10. Bertemu bit 1
 $z = 7359 * 4642^2 \bmod 46189$
 $z = 4928$</p> |
|--|--|

- | | |
|---------------------------|--|
| 11. Bertemu bit 0 | 14. Bertemu bit 0 |
| $z = 4928^2 \bmod 46189$ | $z = 38676^2 \bmod 46189$ |
| $z = 35959$ | $z = 2211$ |
| 12. Bertemu bit 0 | 14. Bertemu bit 1 |
| $z = 35959^2 \bmod 46189$ | $z = 7359 * 2211^2 \bmod 46189$ |
| $z = 34815$ | $z = 66$ |
| 13. Bertemu bit 0 | Maka, hasil dari $7359^{22561} \bmod$ |
| $z = 38676^2 \bmod 46189$ | 46189 adalah 66 |
| $z = 38676$ | $M = 66 - \text{ASCII} - \text{"B"} \text{ (Hasilnya sesuai dengan M semula)}$ |

2.7. *Signature Scheme Based on Numerical Series: Nagaty Digital Signature*

Nagaty *digital signature* diperkenalkan oleh K. A. Nagaty pada tahun 2020. Berbeda dari algoritma lain seperti *Elliptic Curve Cryptography* (ECC) yang menggunakan kurva elips untuk pembangkitan kunci, algoritma RSA dan Diffie-Hellman yang menggunakan eksponensial sehingga membuatnya terlalu lambat dalam komputasi. Algoritma ini bergantung pada kesulitan menghitung jumlah parsial ke-n dari deret integer tak hingga, serta tanpa menggunakan perhitungan eksponensial. Hal tersebut membuat algoritma ini menjadi lebih sederhana dan lebih cepat secara komputasi (Nagaty, 2020).

Dalam hal deret integer tak hingga, suku pertama yang disebut a dan sejumlah besar suku n pada deret dirahasiakan. Secara komputasi diyakini bahwa musuh akan sulit untuk menghitung jumlah parsial ke-n dari deret integer tak hingga dalam algoritma ini.

Konsep dari algoritma Nagaty adalah misalkan pengirim A ingin membangkitkan kunci publik dan kunci privat menggunakan dua deret integer tak hingga yang berbeda, t_1 dan t_2 . Dalam pembangkitan kunci privat digunakan deret t_1 dan pembangkitan kunci publik digunakan deret t_2 . Suku pertama (a) dan sejumlah besar suku n dipilih secara acak dan bernilai besar oleh pengirim A, sehingga suku terakhir n pada deret akan selalu jauh lebih besar dibandingkan suku pertama (a).

2.7.1 Proses Kerja Nagaty *Digital Signature*

Proses kerja Nagaty *digital signature* adalah sebagai berikut (Nagaty, 2020):

- **Pembangkitan Kunci**

1. Bangkitkan bilangan prima besar, dengan syarat: $0 \leq m \leq p - 1$

Dimana: - p adalah bilangan prima

- m adalah pesan asli yang ingin di enkripsi

2. Bangkitkan dua deret tak hingga dengan notasi:

- t1, deret pertama

- a, suku pertama dari deret t1

- r, sejumlah besar suku r pada deret t1

- t2, deret kedua

- b, suku pertama dari deret integer t2

- k, sejumlah besar suku k pada deret integer t2

3. Hitung y_r sebagai kunci privat dengan rumus:

$$y_r = S_r \bmod p$$

Dimana: S_r adalah jumlah parsial r dari deret integer t1

4. Hitung y_k sebagai kunci publik dengan rumus:

$$y_k = S_k \bmod p$$

Dimana: S_k adalah jumlah parsial k dari deret integer t2

- **Proses Tanda Tangan**

1. Lakukan proses tanda tangan dengan kunci privat (y_r), dengan rumus:

$$m_r = m * y_r$$

2. Lakukan proses tanda tangan dengan kunci publik (y_k), dengan rumus:

$$m_k = m * y_k$$

3. Hitung nilai m_{rk} , dengan rumus:

$$m_{rk} = m_r * m_k$$

4. Publikasikan kunci publik (y_k) serta pesan yang sudah ditandatangani oleh dua kunci (m_{rk}) dan pesan yang ditandatangani dengan kunci privat (m_r).

- **Proses Verifikasi**

1. Diketahui nilai y_k , m_{rk} , dan m_r .

2. Hitung nilai m'_k , dengan rumus: $m'_k = m * y_k$

3. Hitung nilai m'_{rk} , dengan rumus: $m'_{rk} = m'_k * m_r$

Cek apakah nilai $m_{rk} = m'_{rk}$? Jika nilai sama, maka pesan dapat dipastikan tidak berubah. Jika nilai berbeda, maka pesan sudah diubah oleh musuh.

2.7.2 Contoh Perhitungan Nagaty *Digital Signature*

- **Pembangkitan Kunci**

1. Bangkitkan bilangan prima besar, dengan syarat: $0 \leq m \leq p - 1$

Misal: $p = 7$ dan $m = 3$

2. Bangkitkan dua deret tak hingga dengan notasi:

- t_1 , deret pertama
- a , suku pertama dari deret t_1
- r , sejumlah besar suku ke- r pada deret t_1
- t_2 , deret kedua
- b , suku pertama dari deret integer t_2
- k , sejumlah besar suku ke- k pada deret integer t_2

- $t_1 = 2, 4, 6, 8, 10, 12, 14, 16, \dots$

$a = 2 \quad r = 5$

- $t_2 = 1, 4, 7, 10, 13, 16, 19, \dots$

$b = 1 \quad k = 6$

3. Hitung y_r sebagai kunci privat dengan rumus:

$$y_r = S_r \bmod p$$

$$y_r = r/2 (a + a_r) \bmod p$$

$$y_r = 5/2 (2 + 10) \bmod 7$$

$$y_r = 30 \bmod 7 = 5$$

4. Hitung y_k sebagai kunci publik dengan rumus:

$$y_k = S_k \bmod p$$

$$y_k = k/2 (a + a_k) \bmod p$$

$$y_k = 6/2 (1 + 16) \bmod 7$$

$$y_k = 51 \bmod 7 = 2$$

- **Proses Tanda Tangan**

- Lakukan proses tanda tangan dengan kunci privat (y_r), dengan rumus:

$$m_r = m * y_r$$

$$m_r = 3 * 5 = 15$$

- Lakukan proses tanda tangan dengan kunci publik (y_k), dengan rumus:

$$m_k = m * y_k$$

$$m_k = 3 * 2 = 6$$

- Hitung nilai m_{rk} , dengan rumus:

$$m_{rk} = m_r * m_k$$

$$m_{rk} = 15 * 6 = 90$$

- Publikasikan kunci publik (y_k) serta pesan yang sudah ditandatangani oleh dua kunci (m_{rk}) dan pesan yang ditandatangani dengan kunci privat (m_r).

- **Proses Verify**

- Diketahui nilai y_k , m_{rk} , dan m_r .
- Asumsikan penerima pesan ingin melakukan verifikasi bahwa pesan tidak berubah.

$$m = 3$$

- Hitung nilai m'_k

$$\text{Dengan rumus: } m'_k = m * y_k$$

$$m'_k = 3 * 2 = 6$$

- Cek apakah nilai $m_{rk} = m'_{rk}$?

$$m_{rk} = m'_{rk}$$

$$90 = 90 \text{ (Terbukti), Pesan berhasil diverifikasi / pesan tidak berubah.}$$

- Hitung nilai m'_{rk}

$$\text{Dengan rumus: } m'_{rk} = m'_k * m_r$$

$$m'_{rk} = 6 * 15 = 90$$

2.8. Teknik *Signcryption* dengan Algoritma XRSA dan Nagaty *Digital Signature*

Teknik *signcryption* yang digunakan dalam penelitian ini adalah metode *sign-then-encrypt*. Dimana pesan yang dikirim akan melalui proses *signing* terlebih dahulu menggunakan algoritma Nagaty *digital signature* dan berubah menjadi *signature message*. Proses *signing* dilakukan untuk memberikan keabsahan pengirim dan keaslian dari pesan.

Lalu *signature message* akan dilakukan proses *encryption* menggunakan algoritma XRSA dan berubah menjadi *signature ciphertext*. Proses enkripsi dilakukan agar pesan tidak dapat dibaca oleh pihak yang tidak sah, sehingga kerahasiaan dari pesan dapat terjaga. Setelah proses *sign* dan *encrypt*, *signature ciphertext* akan dikirim kepada penerima pesan.

Penerima akan melakukan proses dekripsi terlebih dahulu untuk menghasilkan *signature message*. Lalu penerima melakukan verifikasi *signature message* untuk

mengetahui bahwa pesan tidak berubah dan berasal dari pengirim yang sah. Beberapa karakteristik dari *digital signature* adalah (Schneier, 2015):

- Tanda tangan digital sulit untuk dipalsukan karena hanya pengirim yang memiliki kunci pribadi untuk melakukan proses *sign*.
- Tanda tangan digital sulit untuk diubah karena tanda tangan digital akan dibangkitkan berdasarkan pesan yang ingin dikirim. Jika pesan atau tanda tangan digital mengalami perubahan pada saat pengiriman maka tanda tangan digital tidak dapat diverifikasi.

Tanda tangan digital tidak dapat digunakan kembali karena setiap pesan akan membangkitkan tanda tangan digital yang berbeda.

2.9. Faktorisasi *Brute-Force*

Faktorisasi *brute-force* adalah metode pencarian faktor dari suatu bilangan dengan mencoba semua kemungkinan faktor secara sistematis hingga menemukan pasangan faktor yang benar. Dalam konteks faktorisasi bilangan pada RSA, metode ini mencoba semua bilangan prima dari 2 hingga akar kuadrat dari bilangan tersebut untuk menemukan dua bilangan prima yang jika dikalikan menghasilkan n (kunci publik) dalam algoritma RSA (M. A. Budiman & Rachmawati, 2018). Sementara dalam algoritma XRSA, yang menggunakan empat bilangan prima, metode ini mencoba semua bilangan prima dari 2 hingga akar pangkat empat dari bilangan tersebut untuk menemukan keempat faktor primanya.

Brute force dikenal sebagai pendekatan *exhaustive search* atau *generate and test*, yang berarti algoritma ini menjelajahi semua kemungkinan solusi dan menguji setiap kandidat hingga menemukan hasil yang sesuai (M. A. Budiman & Rachmawati, 2018).

Contohnya: Misalkan nilai $n = 3003$

Mencari batas atas pengecekan $\sqrt[4]{3003} = 13.16$ (maka percobaan dilakukan pada bilangan prima dari 2 hingga 13)

- $3003 : 2 = 1501.5$ (tidak habis dibagi 2)
- $3003 : 3 = 1001$ (habis dan ambil 3 sebagai faktor pertamanya)
- $1001 : 5 = 200.2$ (tidak habis dibagi 5)
- $1001 : 7 = 143$ (habis dan ambil 7 sebagai faktor keduanya)
- $143 : 11 = 13$ (habis dan ambil 11 sebagai faktor ketiganya)
- 13 (adalah bilangan prima terakhir)

Maka faktor dari $n = 3003$ adalah $p_1 = 3$, $p_2 = 7$, $p_3 = 11$, dan $p_4 = 13$

2.10. Penelitian Relevan

Berikut adalah beberapa penelitian terdahulu yang berkaitan dengan penelitian ini.

1. Dalam penelitian yang berjudul “*An Effective and Enhanced RSA based Public Key Encryption Scheme (XRSA)*”, menyimpulkan bahwa dibandingkan dengan varian RSA lainnya, XRSA terbukti lebih efisien dalam hal waktu enkripsi dan dekripsi. Algoritma XRSA juga menawarkan peningkatan keamanan signifikan dalam menghadapi *brute-force* melalui penambahan 2 bilangan prima acak pada proses pembangkitan kunci, serta penggunaan operasi XOR untuk meningkatkan kompleksitas dan membuat sistem sulit diserang (Imam, *et al.*, 2022).
2. Berdasarkan penelitian yang berjudul “*A Public Key Cryptosystem and Signature Scheme based on Numerical Series*”, memperkenalkan skema tanda tangan digital yang tidak memerlukan perhitungan eksponensial yaitu skema Nagaty. Skema Nagaty didasarkan pada kesulitan menghitung jumlah parsial ke- n dari deret integer tak hingga, menawarkan efisiensi yang lebih tinggi dalam proses *sign* dan *verify*. Berdasarkan hasil penelitian, metode ini terbukti lebih cepat dan praktis dibandingkan dengan kriptosistem yang bergantung pada logaritma diskrit atau faktorisasi bilangan prima besar.
3. Dalam penelitian Pooja dan Yadav (2018), dijelaskan bahwa tanda tangan digital merupakan solusi penting untuk autentikasi dan keamanan dokumen elektronik. Tanda tangan digital berperan dalam mencegah perubahan isi dokumen serta menegakkan prinsip *non-repudiation*, yang memastikan bahwa penandatanganan tidak dapat menyangkal dokumen yang telah mereka tandatangani. Penelitian ini menggunakan kriptografi kunci publik dalam proses tanda tangan digital dan membahas berbagai skema tanda tangan, termasuk *Schemes with Increased Efficiency*, *Schemes with Increased Security*, *Schemes with Anonymity Services*, serta *Schemes with Enhanced Signing and Verification*.
4. Penelitian yang dilakukan oleh Ginting *et al.* (2024) dalam *paper* yang berjudul “*Signcryption with Matrix Modification of RSA Digital Signature Scheme and Cayley-Purser Algorithm*”, membahas keunggulan skema *signcryption* yang melakukan enkripsi dan tanda tangan digital dalam satu langkah. Sehingga dapat menghemat waktu dan sumber daya komputasi.

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1. Analisis Sistem

Analisis sistem merupakan tahapan penting dalam proses pengembangan sistem yang berfungsi untuk mengidentifikasi kebutuhan, memahami permasalahan, dan memastikan sistem berjalan secara optimal sesuai dengan tujuan. Tahapan ini terdiri dari tiga langkah utama yaitu, analisis masalah yang berfokus pada identifikasi masalah, analisis kebutuhan untuk mendeskripsikan fungsi-fungsi yang diperlukan, serta analisis proses yang bertujuan memodelkan kinerja sistem. Sebagai langkah awal sebelum masuk ke tahap perancangan dan pengembangan, analisis sistem membantu memecah kebutuhan menjadi bagian-bagian kecil yang lebih mudah dipahami sehingga pengembangan dapat dilakukan secara terstruktur dan sesuai dengan sasaran.

3.1.1. Analisis Masalah

Penelitian ini berfokus pada perlindungan dokumen digital, khususnya dalam menjaga keamanannya agar dokumen tidak dapat diubah oleh pihak yang tidak berwenang. Dokumen digital adalah jenis dokumen yang sering digunakan untuk berbagai keperluan, mulai dari pembelajaran, berbagi informasi, hingga hiburan. Dalam beberapa situasi, dokumen ini bersifat penting dan rahasia.

Pada tahap identifikasi masalah, penelitian ini menggunakan metode 5-Whys sebagai alat bantu dalam menganalisis permasalahan. Teknik ini berfokus pada penggalian akar penyebab dengan mengajukan pertanyaan "mengapa" secara berulang hingga ditemukan inti permasalahan. Dengan pendekatan ini, hubungan sebab-akibat dapat diuraikan secara sistematis, seperti dalam analisis berikut:

- a. Mengapa keamanan informasi sangat penting dalam era digital?
Dalam era digital, sebagian besar aktivitas dilakukan secara online, yang menghasilkan banyak data sensitif. Keamanan menjadi elemen utama untuk

memastikan data terlindungi dari ancaman seperti pencurian atau modifikasi oleh pihak tidak berwenang

- b. Mengapa data sensitif perlu dilindungi dengan sistem kriptografi? Dokumen digital sering kali berisi informasi penting yang harus dijaga dari akses atau perubahan oleh pihak yang tidak berwenang. Untuk memastikan keamanannya, diperlukan sistem kriptografi, dimana enkripsi berfungsi untuk mengubah data menjadi format yang tidak dapat dibaca oleh pihak yang tidak berwenang, sedangkan dekripsi memungkinkan penerima yang sah untuk mengembalikannya ke bentuk semula. Dengan demikian, sistem kriptografi membantu menjaga keamanan serta kerahasiaan informasi.

- c. Mengapa penting untuk memverifikasi pengirim dan memastikan pesan tidak diubah?

Keaslian dokumen digital adalah hal penting dalam memastikan bahwa dokumen tidak dipalsukan. Skema tanda tangan digital menyediakan jaminan terhadap integritas dokumen serta membuktikan kepemilikannya. Dengan menggunakan tanda tangan digital, penerima dokumen dapat memastikan bahwa dokumen berasal dari sumber yang valid dan tetap utuh selama proses pengiriman.

- d. Mengapa teknik *signcryption* dibutuhkan dalam pengamanan data? Teknik *signcryption* adalah penggabungan enkripsi dan tanda tangan digital dalam satu proses, sehingga lebih efisien dibandingkan metode konvensional yang memisahkan keduanya. Teknik ini memastikan bahwa data tetap rahasia, terverifikasi, dan tidak dapat diubah oleh pihak yang tidak berwenang.

- e. Mengapa menggunakan algoritma XRSA kriptografi kunci publik dan skema Nagaty *digital signature* dalam teknik *signcryption*?

Algoritma XRSA merupakan algoritma yang dikembangkan dari RSA dengan meningkatkan keamanan melalui penggunaan empat bilangan prima acak besar serta operasi XOR dalam pembangkitan kunci. Metode ini membuat kunci privat lebih sulit diketahui, bahkan jika kunci publik telah diketahui oleh musuh. Sementara itu, algoritma Nagaty *Digital Signature* diperkenalkan pada 2020, menggunakan deret integer tak hingga untuk tanda tangan digital tanpa perhitungan eksponensial, sehingga lebih sederhana dan cepat secara komputasi.

Kesulitan dalam menghitung jumlah parsial ke-n dari deret ini menjadi faktor utama dalam keamanannya.

3.1.2. Analisis Kebutuhan

Analisis kebutuhan merupakan proses terstruktur yang bertujuan untuk mengidentifikasi, memahami, dan mencatat kebutuhan atau persyaratan dari suatu sistem. Tujuannya adalah untuk memastikan bahwa solusi yang dihasilkan dapat secara optimal memenuhi kebutuhan pengguna atau pihak terkait. Proses ini mencakup pengkajian berbagai aspek, seperti kebutuhan fungsional, non-fungsional yang dilakukan melalui berbagai metode, seperti wawancara, survei, observasi, atau analisis dokumen. Hasil dari analisis ini digunakan sebagai landasan dalam merancang, mengembangkan, dan melaksanakan solusi yang sesuai dengan tujuan yang telah ditetapkan.

1. Kebutuhan Fungsional

Kebutuhan fungsional adalah deskripsi mengenai fungsi atau kemampuan yang harus dimiliki oleh suatu sistem untuk memenuhi tujuan yang diinginkan. Berikut merupakan beberapa kebutuhan fungsional yang diperlukan dalam perancangan sistem ini:

- a. Sistem dapat menerima masukan berupa *file* berekstensi docx.
- b. Sistem dapat membangkitkan kunci publik dan kunci privat dari algoritma XRSA yang digunakan dalam proses enkripsi dan dekripsi.
- c. Sistem dapat membangkitkan kunci publik dan kunci privat dari skema Nagaty yang digunakan dalam proses *sign* dan *verify*.
- d. Sistem dapat melakukan proses *sign* dan *verify* terhadap plaintext menggunakan skema Nagaty.
- e. Sistem dapat melakukan proses enkripsi dan dekripsi terhadap plaintext menggunakan algoritma XRSA.
- f. Sistem dapat menyimpan ciphertext.

2. Kebutuhan Nonfungsional

Kebutuhan nonfungsional adalah persyaratan yang menentukan kualitas atau cara kerja suatu sistem, seperti kecepatan, keamanan, keandalan, dan

kemudahan digunakan. Berikut merupakan beberapa kebutuhan nonfungsional yang diperlukan dalam perancangan sistem ini:

- a. Sistem harus memiliki antarmuka yang sederhana, mudah dipahami, serta dapat menampilkan dan menyimpan hasil masukan pengguna.
- b. Sistem menampilkan notifikasi kesalahan ketika pengguna melakukan kekeliruan, seperti memberikan input yang tidak lengkap atau tidak sesuai dengan persyaratan yang ditentukan.
- c. Sistem dapat melakukan proses *sign-then-encrypt* dan *decrypt-then-verify* dengan *input* dan *output* yang tepat.

3.1.3. Analisis Proses

Sistem yang dibangun menggunakan algoritma XRSA dan skema Nagaty. Algoritma XRSA digunakan pada proses enkripsi dan dekripsi *file* teks. Sedangkan skema Nagaty digunakan pada proses *sign* dan *verify file* teks. Dalam penelitian ini *file* teks yang digunakan adalah *file* teks dalam format *.docx.

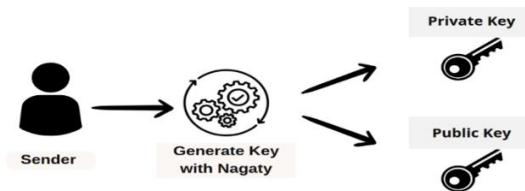
Penerima membangkitkan kunci publik dan privat menggunakan algoritma XRSA yang akan digunakan untuk proses enkripsi dan dekripsi *file* teks. Pengirim membangkitkan kunci publik dan privat menggunakan skema nagaty yang akan digunakan untuk proses *sign* dan *verify file* teks.

3.2. Pemodelan Sistem

Pemodelan sistem merupakan proses merepresentasikan suatu sistem yang dirancang dalam bentuk diagram. Pemodelan sistem pada penelitian ini dibuat dalam diagram umum, *use case diagram* dan *activity diagram*.

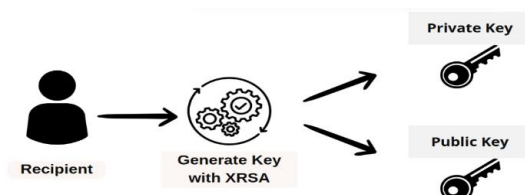
3.2.1. Diagram Umum

Diagram umum adalah representasi visual yang menggambarkan hubungan atau alur kerja dalam suatu sistem untuk mempermudah pemahaman. Pada penelitian ini terdapat 3 bagian diagram umum, yaitu diagram umum pembangkitan kunci oleh *sender*, pembangkitan kunci oleh *recipient*, dan diagram umum *signcrypton*.



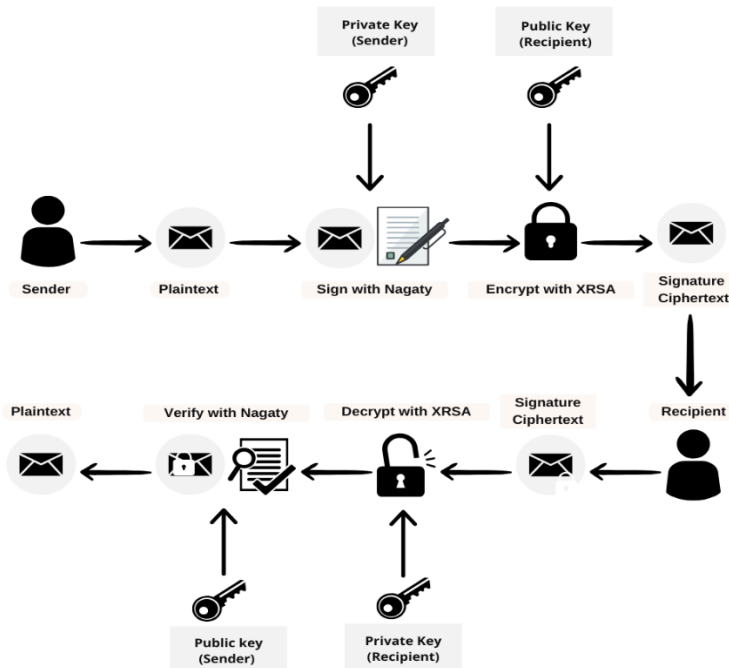
Gambar 3.1 Diagram Umum Pembangkitan Kunci *Sender*

Pada gambar 3.1 dapat dilihat proses pembangkitan kunci oleh *sender* dengan skema Nagaty. Langkah pertama dalam proses ini adalah membangkitkan satu bilangan prima yang digunakan untuk menghitung nilai y_r sebagai kunci privat dan y_k sebagai kunci publik. Kunci tersebut nantinya akan digunakan dalam proses penandatanganan (*sign*) dan verifikasi (*verify*).



Gambar 3.2 Diagram Umum Pembangkitan Kunci *Recipient*

Pada gambar 3.2 dapat dilihat proses pembangkitan kunci oleh *recipient* dengan algoritma XRSA. Langkah pertama dalam proses ini adalah membangkitkan empat bilangan prima yang digunakan dalam perhitungan nilai *private key* (D) dan *public key* (E, n). Dimana kunci ini akan digunakan dalam proses *encrypt* dan *decrypt*.

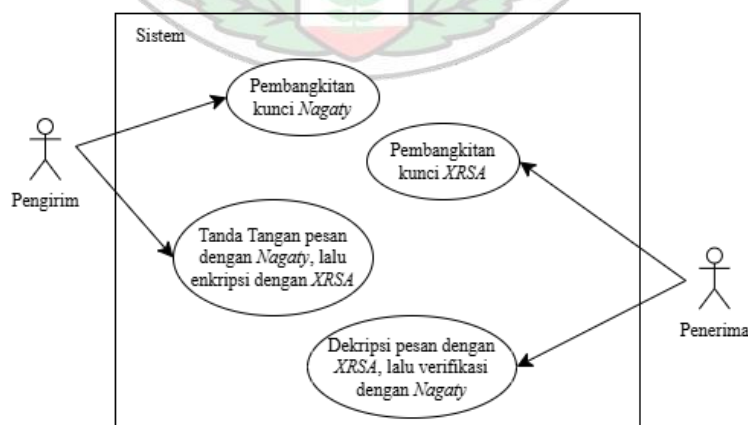


Gambar 3.3 Diagram Umum *Signcrypt*

Pada gambar 3.3 dapat dilihat dalam proses *signcryption*, pengirim (*sender*) terlebih dahulu memasukkan pesan (*plaintext*). Selanjutnya, pengirim melakukan proses tanda tangan pada *plaintext* menggunakan kunci pribadinya, sehingga menghasilkan *signature message*. *Signature message* tersebut kemudian dienkripsi menggunakan kunci publik penerima, menghasilkan *signature ciphertext*. *Signature ciphertext* ini dikirim melalui jalur yang tidak aman (*unsecure channel*). Setelah diterima, penerima (*recipient*) melakukan proses dekripsi menggunakan kunci pribadinya untuk memperoleh kembali *plaintext* dan *signature message*. Terakhir, penerima melakukan proses verifikasi pada *signature message* menggunakan kunci publik pengirim. Jika verifikasi berhasil, penerima dapat memastikan bahwa pesan tidak mengalami perubahan selama transmisi.

3.2.2. Use Case Diagram

Use Case Diagram adalah diagram dalam pemodelan perangkat lunak yang digunakan untuk menggambarkan interaksi antara sistem dan aktor eksternal yang berperan di dalamnya. Diagram ini menunjukkan serangkaian fungsi (*use case*) yang dapat dilakukan oleh sistem serta siapa yang dapat menggunakannya. Selain itu, *use case* diagram berperan dalam mengidentifikasi dan memahami bagaimana sistem beroperasi dalam situasi dunia nyata dengan memvisualisasikan hubungan antara pengguna atau entitas eksternal lainnya dengan sistem yang dikembangkan.



Gambar 3.4 *Use Case Diagram*

Pada Gambar 3.4, terdapat dua aktor yang berinteraksi dengan sistem, yaitu pengirim dan penerima. Pengirim membangkitkan kunci menggunakan algoritma Nagaty dengan, yang kemudian digunakan dalam proses penandatanganan dan

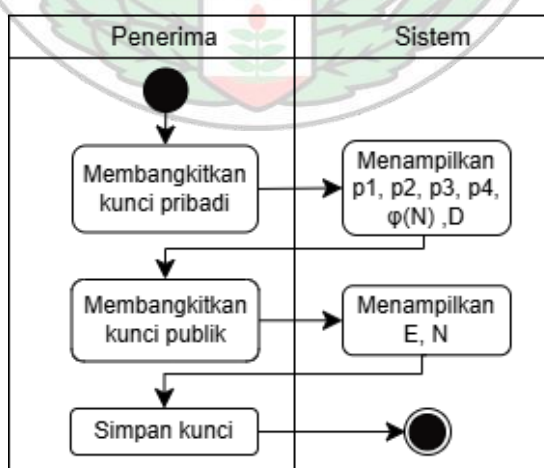
verifikasi. Sementara itu, penerima membangkitkan kunci menggunakan algoritma XRSA dengan bilangan prima yang dihasilkan secara acak, yang kemudian digunakan dalam proses enkripsi dan dekripsi.

Setelah kunci dibangkitkan, pengirim melakukan penandatanganan pesan menggunakan algoritma Nagaty dan kunci privatnya. Pesan yang telah ditandatangani kemudian dienkripsi menggunakan algoritma XRSA dengan kunci publik penerima sehingga menghasilkan ciphertext. Penerima kemudian mendekripsi ciphertext menggunakan algoritma XRSA dan kunci privatnya. Setelah proses dekripsi selesai, penerima melakukan verifikasi tanda tangan menggunakan algoritma Nagaty dengan kunci publik pengirim untuk memastikan bahwa pesan tidak mengalami perubahan selama proses pengiriman.

3.2.3. Activity Diagram

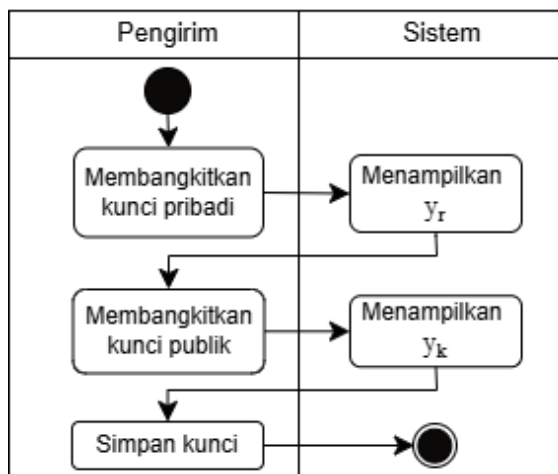
Activity diagram merupakan diagram yang digunakan untuk memodelkan proses atau alur kerja dalam sebuah sistem, menunjukkan urutan aktivitas dan perpindahan alur antar aktivitas. Diagram ini berguna untuk menggambarkan logika sistem atau proses sistem secara rinci.

3.2.3.1. Activity Diagram Pembangkit Kunci



Gambar 3.5 Activity Diagram Pembangkit Kunci XRSA Penerima

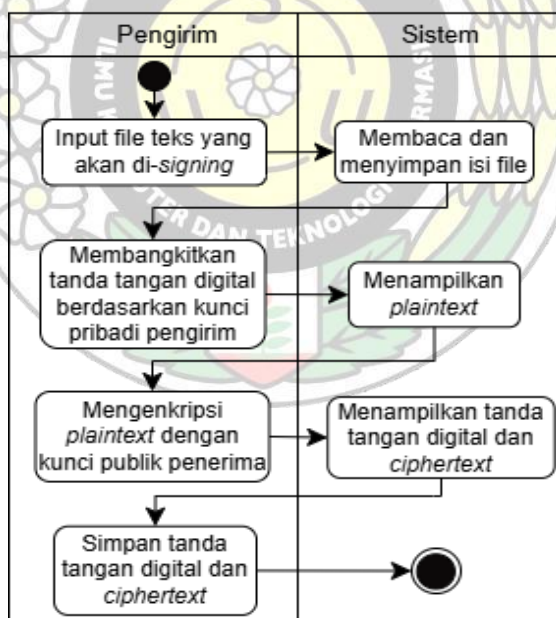
Pada gambar 3.5 dapat dilihat aktivitas pengguna pada sisi penerima terhadap sistem saat melakukan pembangkitan kunci publik dan kunci privat.



Gambar 3.6 Activity Diagram Pembangkit Kunci Nagaty Pengirim

Pada gambar 3.6 dapat dilihat aktivitas pengguna pada sisi pengirim terhadap sistem saat melakukan pembangkitan kunci privat dan kunci publik.

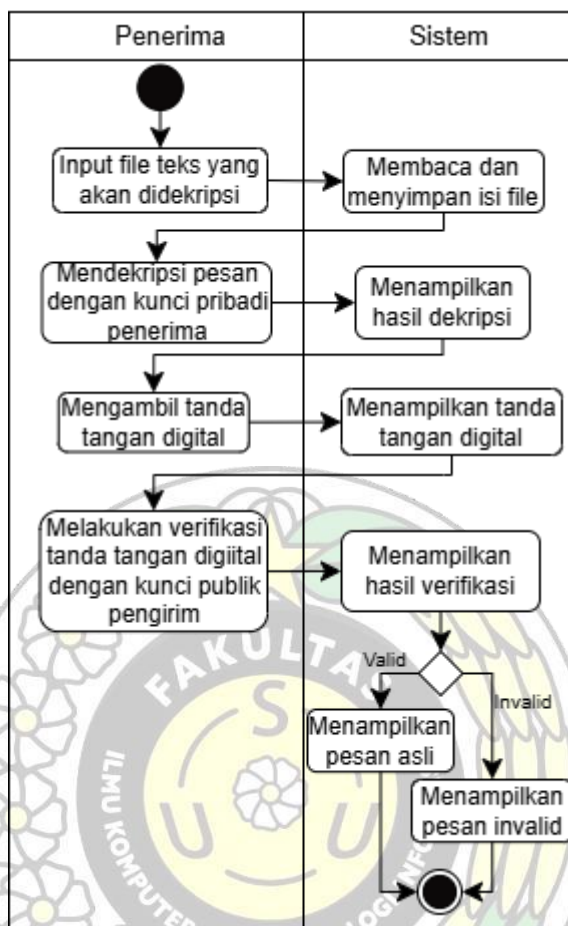
3.2.3.2. Activity Diagram Proses Tanda Tangan dan Enkripsi



Gambar 3.7 Activity Diagram Proses Tanda Tangan dan Enkripsi

Pada gambar 3.7 dapat dilihat aktivitas pengguna sebagai pengirim yang akan melakukan proses tanda tangan menggunakan kunci pribadi pengirim dan proses enkripsi menggunakan kunci publik penerima.

3.2.3.3. Activity Diagram Proses Dekripsi dan Verifikasi



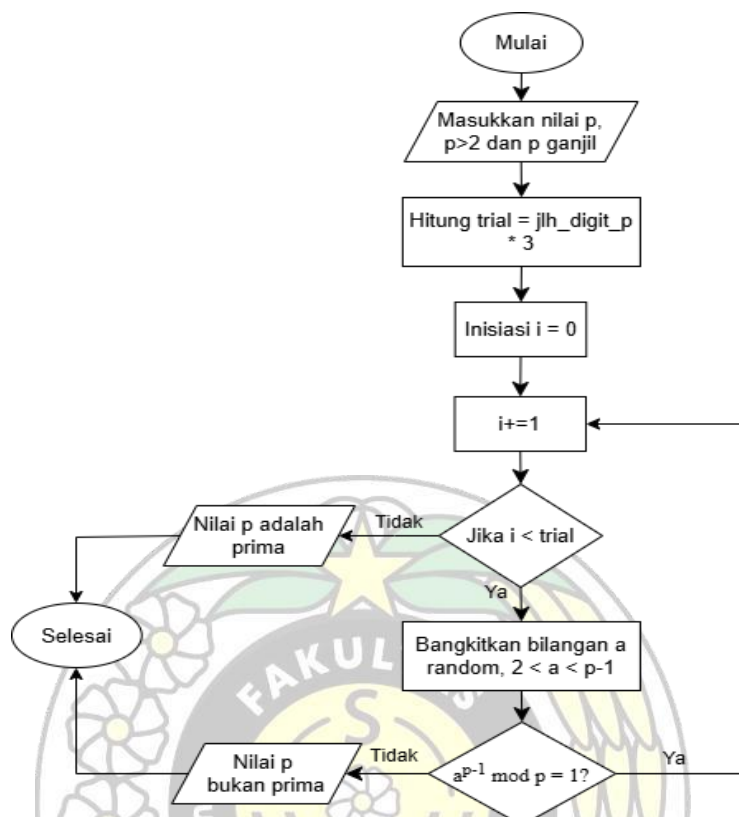
Gambar 3.8 Activity Diagram Proses Dekripsi dan Verifikasi

Pada gambar 3.8 dapat dilihat aktivitas pengguna sebagai penerima yang akan melakukan proses dekripsi menggunakan kunci pribadi penerima dan proses verifikasi menggunakan kunci publik pengirim.

3.3. Flowchart (Diagram Alir)

Flowchart merupakan diagram alir yang digunakan untuk menggambarkan alur atau urutan suatu proses, sistem, atau prosedur secara visual. Diagram ini memanfaatkan simbol-simbol grafis untuk menggambarkan setiap langkah dalam proses dan hubungan antara setiap tahapan dalam urutan yang logis.

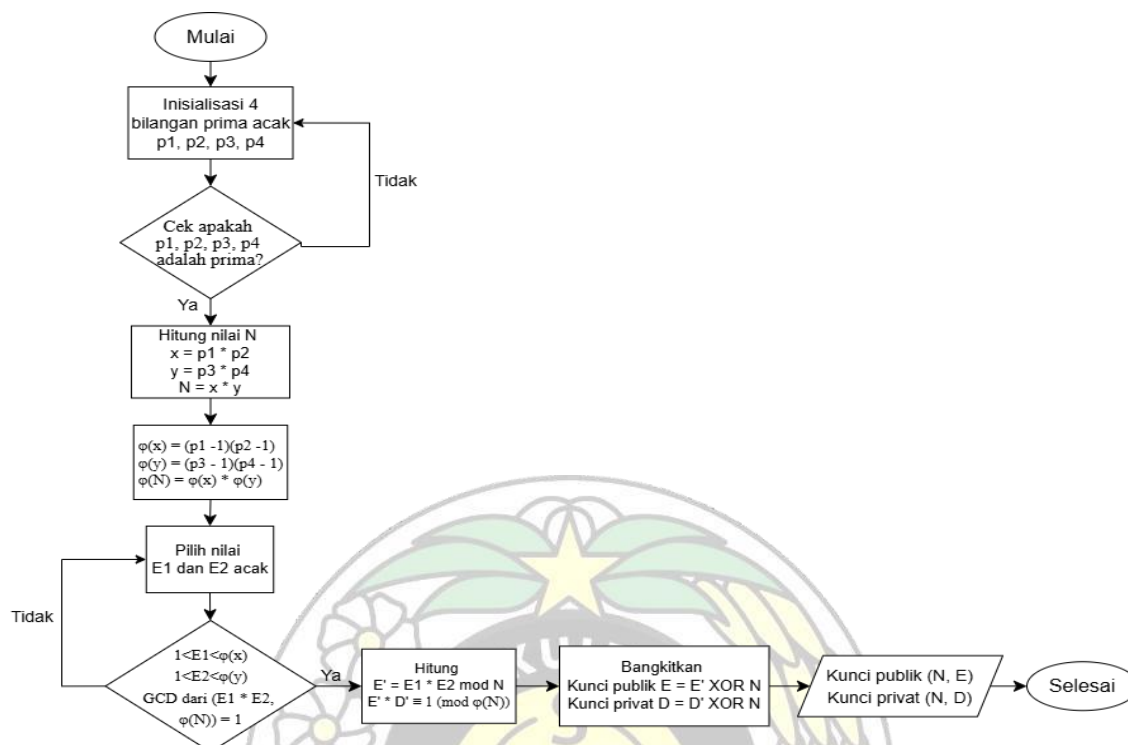
3.3.1. Flowchart Pembangkitan Bilangan Prima *Fermat's Little Theorem*



Gambar 3. 9 Flowchart Pembangkit Bilangan Prima

Pada Gambar 3.9 dapat dilihat proses pembangkitan bilangan prima secara acak dengan menggunakan *fermat's little theorem*. Pengujian dilakukan sebanyak tiga kali dari panjang bilangan p . Sistem akan memilih bilangan a secara acak dalam rentang 2 hingga $p - 1$. Jika hasil perhitungan $a^{p-1} \bmod p \neq 1$ terpenuhi, maka proses perulangan akan dilanjutkan. Namun, jika tidak, maka program akan berhenti.

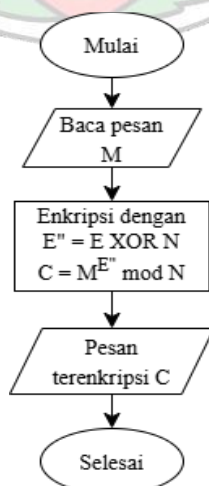
3.3.2. Flowchart Pembangkit Kunci Algoritma XRSA



Gambar 3.10 Flowchart Pembangkit Kunci pada Algoritma XRSA

Pada gambar 3.10 dapat dilihat proses pembangkit kunci publik dan kunci privat pada algoritma XRSA yang dilakukan oleh penerima. Kunci publik dan kunci privat tersebut digunakan dalam proses enkripsi dan dekripsi *file* teks.

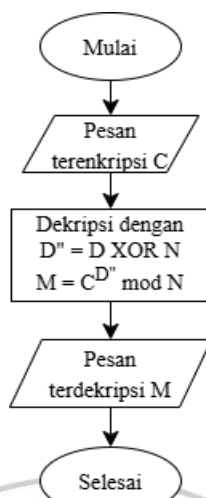
3.3.3. Flowchart Enkripsi XRSA



Gambar 3.11 Flowchart Enkripsi Algoritma XRSA

Pada Gambar 3.11 memperlihatkan proses enkripsi pada algoritma XRSA yang dilakukan oleh pengirim.

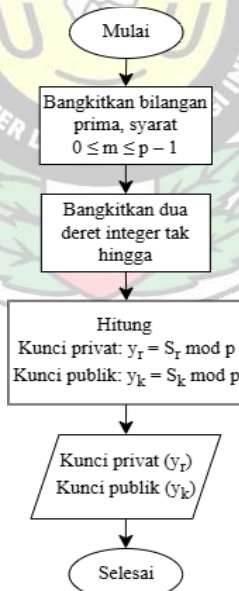
3.3.4. Flowchart Dekripsi XRSA



Gambar 3.12 Flowchart Dekripsi Algoritma XRSA

Pada Gambar 3.12 memperlihatkan proses dekripsi pada algoritma XRSA yang dilakukan oleh penerima.

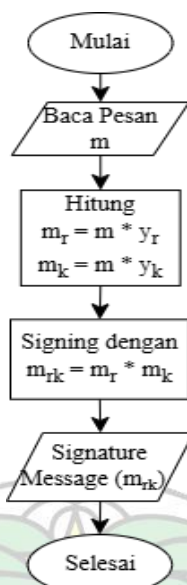
3.3.5. Flowchart Pembangkit Kunci Skema Nagaty



Gambar 3.13 Flowchart Pembangkit Kunci pada Skema Nagaty

Pada gambar 3.13 dapat dilihat proses pembangkit kunci publik dan kunci privat pada skema Nagaty yang dilakukan oleh pengirim. Kunci publik dan kunci privat tersebut digunakan dalam proses tanda tangan dan verifikasi.

3.3.6. Flowchart Tanda Tangan Nagaty



Gambar 3.14 Flowchart Tanda Tangan Skema Nagaty

Pada Gambar 3.14 memperlihatkan proses tanda tangan pada skema Nagaty yang dilakukan oleh pengirim.

3.3.7. Flowchart Verifikasi Nagaty



Gambar 3.15 Flowchart Verifikasi Skema Nagaty

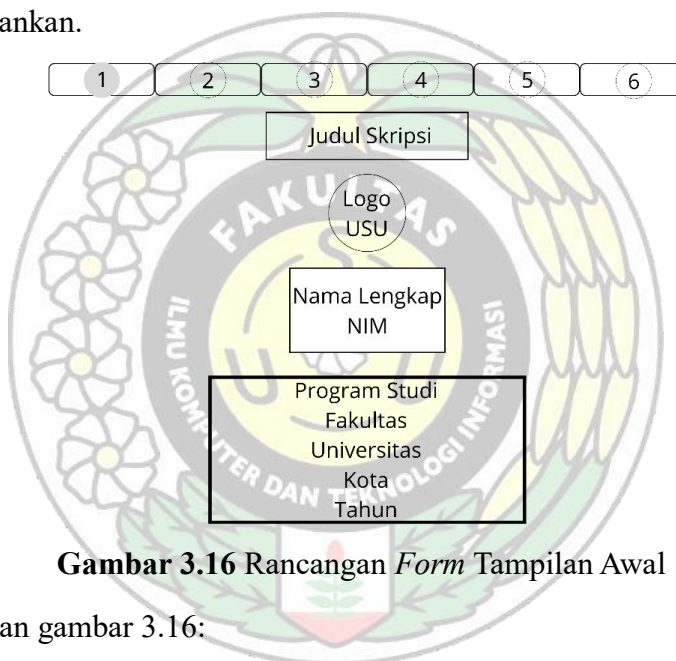
Pada Gambar 3.15 memperlihatkan proses verifikasi pada skema Nagaty yang dilakukan oleh penerima.

3.4. Perancangan Antarmuka (*Interface*)

Desain antarmuka (*interface*) adalah proses perancangan tampilan dan interaksi dalam suatu sistem atau aplikasi agar mudah digunakan oleh pengguna. Elemen dalam desain ini mencakup tata letak, ikon, warna, serta fitur interaktif seperti tombol dan navigasi yang bertujuan meningkatkan kenyamanan serta efisiensi saat berinteraksi dengan sistem.

3.4.1. *Form Tampilan Awal (Home)*

Form tampilan awal berisi informasi penulis yang akan ditampilkan pertama saat program dijalankan.



Gambar 3.16 Rancangan *Form Tampilan Awal*

Keterangan gambar 3.16:

1. *Button* untuk menampilkan *form* home.
2. *Button* untuk menampilkan *form* pembangkitan kunci Nagaty.
3. *Button* untuk menampilkan *form* pembangkitan kunci XRSA.
4. *Button* untuk menampilkan *form* tanda tangan dan enkripsi.
5. *Button* untuk menampilkan *form* dekripsi dan verifikasi.
6. *Button* untuk menampilkan *form* faktorisasi *brute-force*.

3.4.2. *Form Pembangkitan Kunci Nagaty*

Form pembangkitan kunci Nagaty dapat diakses oleh pengirim untuk membangkitkan kunci Nagaty berdasarkan panjang bit primanya.

Pembangkitan Kunci Nagaty

Masukkan Bit Length:

8

Gambar 3.17 Rancangan *Form* Pembangkitan Kunci Nagaty

Keterangan gambar 3.17:

1. *Button* untuk menampilkan *form* home.
2. *Button* untuk menampilkan *form* pembangkitan kunci Nagaty.
3. *Button* untuk menampilkan *form* pembangkitan kunci XRSA.
4. *Button* untuk menampilkan *form* tanda tangan dan enkripsi.
5. *Button* untuk menampilkan *form* dekripsi dan verifikasi.
6. *Button* untuk menampilkan *form* faktorisasi *brute-force*.
7. *Input* berisi panjang prima dalam bit.
8. *Button submit* untuk melakukan proses pembangkitan kunci Nagaty.

3.4.3. *Form* Pembangkitan Kunci XRSA

Form pembangkitan kunci XRSA dapat diakses oleh penerima untuk membangkitkan kunci XRSA berdasarkan panjang bit primanya.

Pembangkitan Kunci XRSA

Masukkan Bit Length:

8

Gambar 3.18 Rancangan *Form* Pembangkitan Kunci XRSA

Keterangan gambar 3.18:

1. *Button* untuk menampilkan *form* home.
2. *Button* untuk menampilkan *form* pembangkitan kunci Nagaty.
3. *Button* untuk menampilkan *form* pembangkitan kunci XRSA.
4. *Button* untuk menampilkan *form* tanda tangan dan enkripsi.
5. *Button* untuk menampilkan *form* dekripsi dan verifikasi.

6. *Button* untuk menampilkan *form* faktorisasi *brute-force*.
7. *Input* berisi panjang prima dalam bit.
8. *Button submit* untuk melakukan proses pembangkitan kunci XRSA.

3.4.4. *Form* Tanda Tangan dan Enkripsi

Form tanda tangan dan enkripsi dapat diakses oleh pengirim untuk melakukan proses tanda tangan pesan dan enkripsi, dengan memasukkan kunci publik dari penerima yaitu nilai n dan E .

Gambar 3.19 Rancangan *Form* Tanda Tangan dan Enkripsi

Keterangan gambar 3.19:

1. *Button* untuk menampilkan *form* home.
2. *Button* untuk menampilkan *form* pembangkitan kunci Nagaty.
3. *Button* untuk menampilkan *form* pembangkitan kunci XRSA.
4. *Button* untuk menampilkan *form* tanda tangan dan enkripsi.
5. *Button* untuk menampilkan *form* dekripsi dan verifikasi.
6. *Button* untuk menampilkan *form* faktorisasi *brute-force*.
7. *Input* berisi nilai E .
8. *Input* berisi nilai n .
9. *Input file* untuk memasukkan *file* yang akan ditanda tangani dan dienkripsi.
10. *Button submit* untuk melakukan proses tanda tangan dan enkripsi.
11. *Button download* untuk mengunduh *file* hasil tanda tangan dan enkripsi.

3.4.5. Form Dekripsi dan Verifikasi

Form dekripsi dan verifikasi dapat diakses oleh penerima untuk melakukan proses dekripsi dan verifikasi tanda tangan, dengan memasukkan kunci publik dari pengirim yaitu nilai *yk*.

Proses Dekripsi & Verifikasi

Masukkan Kunci Publik Pengirim

Masukkan *yk*:

Masukkan file hasil enkripsi

Download Hasil

Gambar 3.20 Rancangan *Form* Dekripsi dan Verifikasi

Keterangan gambar 3.20:

1. *Button* untuk menampilkan *form* home.
2. *Button* untuk menampilkan *form* pembangkitan kunci Nagaty.
3. *Button* untuk menampilkan *form* pembangkitan kunci XRSA.
4. *Button* untuk menampilkan *form* tanda tangan dan enkripsi.
5. *Button* untuk menampilkan *form* dekripsi dan verifikasi.
6. *Button* untuk menampilkan *form* faktorisasi *brute-force*.
7. *Input* berisi nilai *yk*.
8. *Input file* untuk memasukkan *file* yang akan didekripsi dan verifikasi.
9. *Button submit* untuk melakukan proses dekripsi dan verifikasi.
10. *Button download* untuk mengunduh *file* hasil dekripsi dan verifikasi.

3.4.6. Form Faktorisasi *Brute-Force*

Form faktorisasi *brute-force* dapat diakses oleh kriptanalis untuk melakukan proses faktorisasi *Brute-force* terhadap nilai *n* yang telah dibangkitkan dengan algoritma XRSA sehingga menghasilkan nilai *p1*, *p2*, *p3*, dan *p4*.



Faktorisasi Brute Force

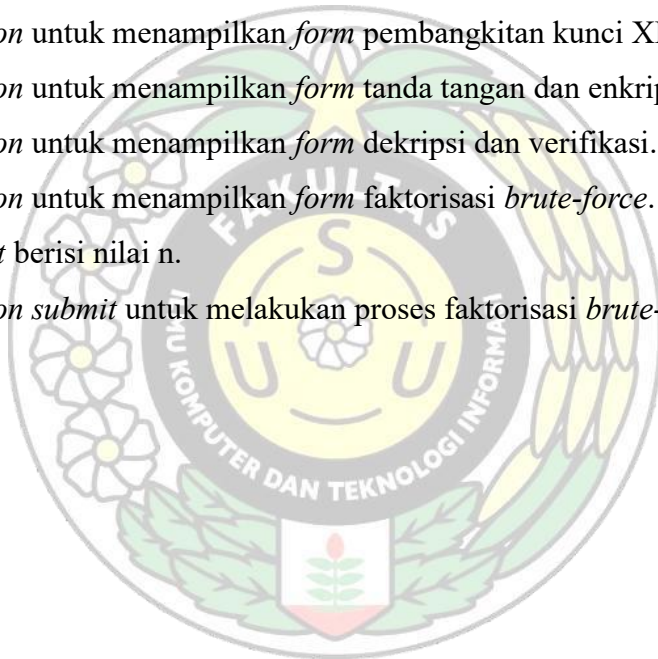
Masukkan nilai n:

Hasil Faktorisasi:

Gambar 3.21 Rancangan *Form Faktorisasi Brute-Force*

Keterangan gambar 3.21:

1. *Button* untuk menampilkan *form* home.
2. *Button* untuk menampilkan *form* pembangkitan kunci Nagaty.
3. *Button* untuk menampilkan *form* pembangkitan kunci XRSA.
4. *Button* untuk menampilkan *form* tanda tangan dan enkripsi.
5. *Button* untuk menampilkan *form* dekripsi dan verifikasi.
6. *Button* untuk menampilkan *form* faktorisasi *brute-force*.
7. *Input* berisi nilai n.
8. *Button submit* untuk melakukan proses faktorisasi *brute-force*.



BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

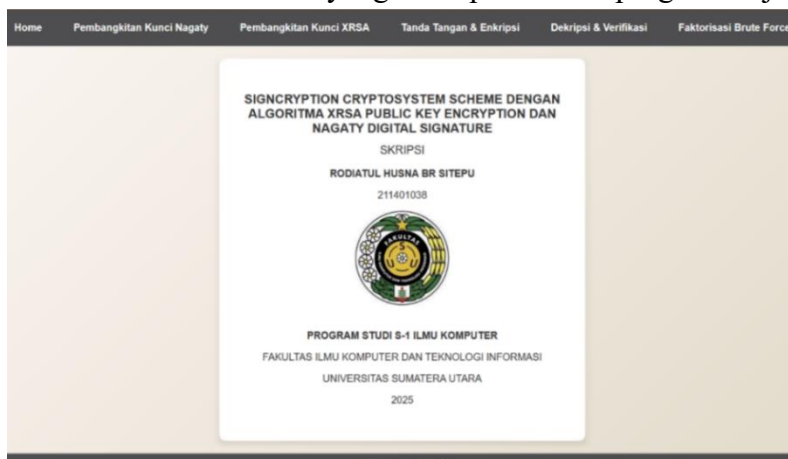
4.1. Implementasi Sistem

Dalam penelitian ini, sistem yang dikembangkan menerapkan teknik *signcryption*, di mana proses enkripsi dan dekripsi dilakukan menggunakan algoritma XRSA, sedangkan tanda tangan digital serta verifikasinya menggunakan skema Nagaty. Data yang digunakan dalam penelitian ini berbentuk teks dengan ekstensi *.docx. Sistem dibangun dengan bahasa pemrograman Python dengan *framework* Django menggunakan IDE *Visual Studio Code* versi 1.97.2. Spesifikasi *hardware* yang digunakan adalah *processor Intel® Core™ i7-10750H*, *RAM 16GB*, dan sistem operasi *Windows 11 Home Single Language*.

Sistem yang dirancang terdiri dari enam laman utama, yaitu laman *Home*, laman Pembangkitan Kunci Nagaty, laman Pembangkitan Kunci XRSA, laman untuk proses Tanda Tangan dan Enkripsi, laman Dekripsi dan Verifikasi, serta laman Faktorisasi *Brute-Force*.

4.1.1. Laman *Home*

Laman *home* adalah laman utama yang ditampilkan saat program di jalankan.



Gambar 4.1 Laman Home

Pada gambar 4.1 dapat dilihat laman utama yaitu laman *home* yang berisi judul skripsi untuk menjelaskan algoritma yang digunakan dalam sistem.

4.1.2. Laman Pembangkitan Kunci Nagaty

Pada laman ini pengguna dapat membangkitkan kunci Nagaty yaitu nilai y_r (kunci privat) dan y_k (kunci publik) yang akan digunakan dalam proses tanda tangan dan verifikasi.



Gambar 4.2 Laman Pembangkitan Kunci Nagaty

Pada gambar 4.2 memperlihatkan bahwa pengguna dapat membangkitkan kunci Nagaty dengan memasukkan panjang bit prima terlebih dahulu. Panjang bit prima minimal yang dimasukkan adalah 39-bit, karena terdapat syarat bahwa prima Nagaty harus lebih besar dari *plaintext*.

4.1.3. Laman Pembangkitan Kunci XRSA

Pada laman ini pengguna dapat membangkitkan kunci XRSA yaitu nilai p_1 , nilai p_2 , nilai p_3 , nilai p_4 , nilai n , nilai D , dan nilai E . Dimana nilai kunci XRSA tersebut akan digunakan dalam proses enkripsi dan dekripsi.



Gambar 4.3 Laman Pembangkitan Kunci XRSA

Pada gambar 4.3 memperlihatkan bahwa pengguna dapat membangkitkan kunci Nagaty dengan memasukkan panjang bit prima terlebih dahulu, lalu klik tombol bangkitkan kunci XRSA. Maka akan tampil nilai – nilai kunci publik dan kunci privat algoritma XRSA yang akan digunakan dalam proses enkripsi dan dekripsi.

4.1.4. Laman Tanda Tangan dan Enkripsi

Pada laman ini pengguna dapat melakukan proses tanda tangan dan enkripsi.



Gambar 4.4 Laman Tanda Tangan dan Enkripsi

Pada gambar 4.4 memperlihatkan laman tanda tangan dan enkripsi, dimana pengguna memasukkan *file* yang ingin ditanda tangan dan enkripsi. Setelah itu klik tombol tanda tangan dan enkripsi, maka hasilnya akan keluar dalam bentuk *file* docx yang dapat di unduh.

4.1.5. Laman Dekripsi dan Verifikasi

Pada laman ini pengguna dapat melakukan proses dekripsi dan verifikasi.



Gambar 4.5 Laman Dekripsi dan Verifikasi

Pada gambar 4.5 memperlihatkan laman dekripsi dan verifikasi, dimana pengguna diminta memasukkan nilai kunci privat XRSA yaitu D, kunci publik XRSA yaitu n, dan kunci publik Nagaty yaitu yk. Selanjutnya pengguna di minta memasukkan *file* hasil enkripsi atau *file* yang ingin di dekripsi, lalu klik tombol dekripsi dan verifikasi. Maka hasilnya akan keluar, jika verifikasi valid maka ditampilkan pesan aslinya. Namun jika hasilnya tidak valid maka pesan tidak di konversi ke karakter, dan ditampilkan pesan tidak terverifikasi.

4.1.6. Laman Faktorisasi *Brute-Force*

Pada laman ini, seorang kriptanalisis dapat melakukan proses kriptanalisis dengan faktorisasi *Brute-force* terhadap nilai n yang telah dibangkitkan dengan algoritma XRSA sehingga menghasilkan nilai p_1 , p_2 , p_3 , dan p_4 .



Gambar 4.6 Laman Faktorisasi *Brute-Force*

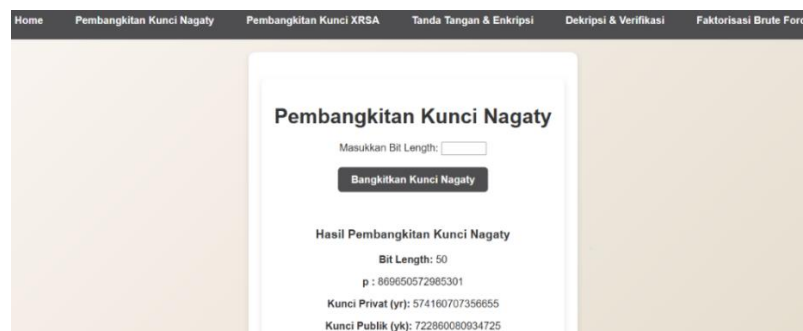
Pada gambar 4.6 memperlihatkan laman faktorisasi *Brute-force*, dimana pengguna diminta memasukkan nilai n yang ingin di faktorisasi. Lalu klik tombol faktorisasi, maka hasil faktorisasi akan keluar berupa nilai p_1 , p_2 , p_3 , dan p_4 .

4.2. Pengujian Sistem

Pada tahap ini, sistem akan diuji untuk memastikan bahwa fungsinya berjalan sesuai dengan yang telah dijelaskan. Pengujian dilakukan pada proses pembangkitan kunci, tanda tangan dan enkripsi, serta dekripsi dan verifikasi. Tujuan dari pengujian ini adalah untuk memastikan apakah pesan dapat diverifikasi dengan benar dan dikembalikan ke bentuk aslinya menggunakan kombinasi algoritma XRSA dan skema Nagaty.

4.2.1. Pengujian Pembangkitan Kunci

Melakukan pengujian terhadap laman pembangkitan kunci Nagaty yang dilakukan oleh pengirim dan laman pembangkitan kunci XRSA yang dilakukan oleh penerima.



Gambar 4.7 Pengujian Pembangkitan Kunci Nagaty

Pada gambar 4.7 dapat dilihat hasil dari pembangkitan kunci Nagaty yang dilakukan pengirim, yaitu nilai p , yr , dan yk .

Gambar 4.8 Pengujian Pembangkitan Kunci XRSA

Pada gambar 4.8 dapat dilihat hasil dari pembangkitan kunci XRSA yang dilakukan penerima, yaitu nilai p_1 , p_2 , p_3 , p_4 , E , n , dan nilai D .

4.2.2. Pengujian Tanda Tangan dan Enkripsi

Melakukan pengujian terhadap laman tanda tangan dan enkripsi yang dilakukan oleh pengirim. Hasil tanda tangan dan enkripsi dapat diunduh dalam bentuk *file* *.docx, dapat dilihat pada gambar 4.9.

Gambar 4.9 Pengujian Tanda Tangan dan Enkripsi

Adapun hasil tanda tangan dan enkripsi yang telah di unduh dalam bentuk *file* *.docx, dapat dilihat pada gambar 4.10 di bawah ini.

```

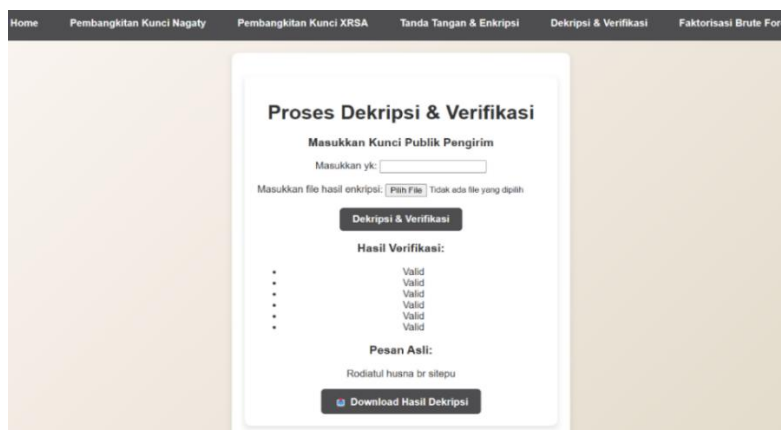
Hasil Sign & Enkripsi:
C[0] = 319661669410061080680758058086786718338930720673893457771890
C[1] = 365408847259447243434566010060647836394759235051005719350333
C[2] = 12583129872215963523774284656611340764600550518634705622052
C[3] = 63920020769905558854588351709400686989246857153822908209776
C[4] = 294359413358405788985697938614581236277914220710259192083367
C[5] = 147948548834601566680247655561039964441686029598205481443241

```

Gambar 4.10 Dokumen Hasil Tanda Tangan dan Enkripsi

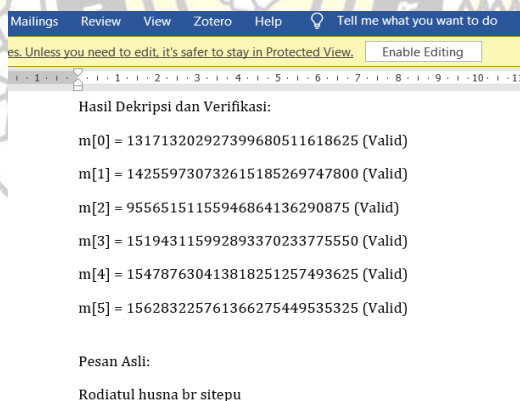
4.2.3. Pengujian Dekripsi dan Verifikasi

Melakukan pengujian terhadap laman dekripsi dan verifikasi yang dilakukan oleh penerima. Penerima memasukkan nilai y_k , D , n , serta *file* yang ingin didekripsi dan verifikasi. Hasil dekripsi dan verifikasi dapat dilihat langsung di web maupun diunduh dalam bentuk *file* *.docx, seperti pada gambar 4.11.



Gambar 4.11 Pengujian Dekripsi dan Verifikasi

Adapun hasil dekripsi dan verifikasi yang telah di unduh dalam bentuk *file* *.docx, dapat dilihat pada gambar 4.12 di bawah ini.



Gambar 4.12 Dokumen Hasil Dekripsi dan Verifikasi

4.2.4. Pengujian Faktorisasi *Brute-Force*

Melakukan pengujian pada laman faktorisasi *brute-force* dengan memasukkan nilai n , lalu mengklik tombol faktorisasi. Hasilnya akan menampilkan faktor-faktor dari n , yaitu p_1 , p_2 , p_3 , dan p_4 . Nilai n yang dimasukkan merupakan kunci publik XRSA yang digunakan dalam proses enkripsi dan dekripsi. Adapun tampilan *brute-force* setelah di uji, dapat dilihat pada gambar 4.13 di bawah ini.



Gambar 4.13 Pengujian Faktorisasi *Brute-Force*

4.2.5. Pengujian *Real Running Time* Sistem

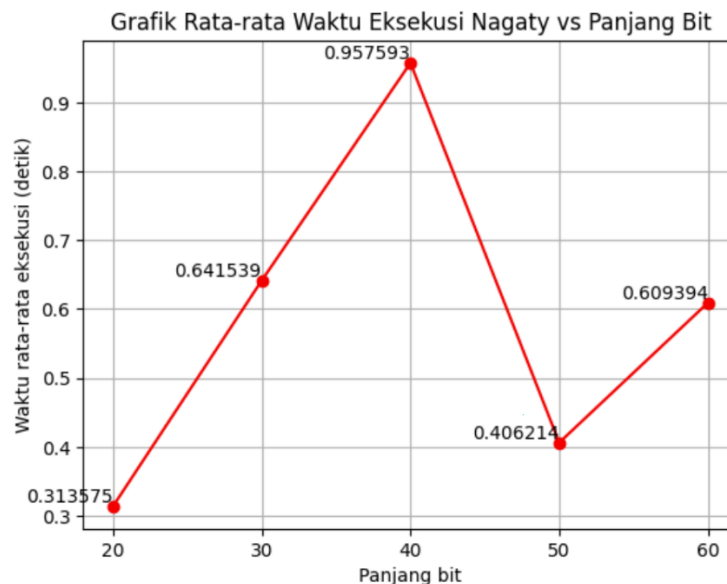
Pengujian *real running time* adalah metode untuk mengukur waktu eksekusi sebenarnya dari suatu program atau algoritma dalam kondisi tertentu. Tujuan pengujian ini adalah untuk menilai efisiensi dan kinerja sistem berdasarkan faktor-faktor seperti ukuran dokumen dan panjang bit bilangan prima. Dengan pengujian ini, waktu eksekusi dapat diukur secara akurat, sehingga dapat diketahui seberapa cepat sistem bekerja dalam kondisi nyata.

4.2.5.1. Pengujian *Real Running Time* pada Pembangkitan Kunci

Pengujian *real running time* pada pembangkitan kunci dilakukan pada pembangkitan kunci Nagaty dan pembangkitan kunci XRSA, dimana pengujian ini dilakukan untuk mengetahui waktu proses pembangkitan kunci terhadap panjang bit prima. Pengujian ini dilakukan pada panjang prima 20-bit, 30-bit, 40-bit, 50-bit, dan 60-bit.

Tabel 4.1 Pengujian Waktu Pembangkitan Kunci Nagaty

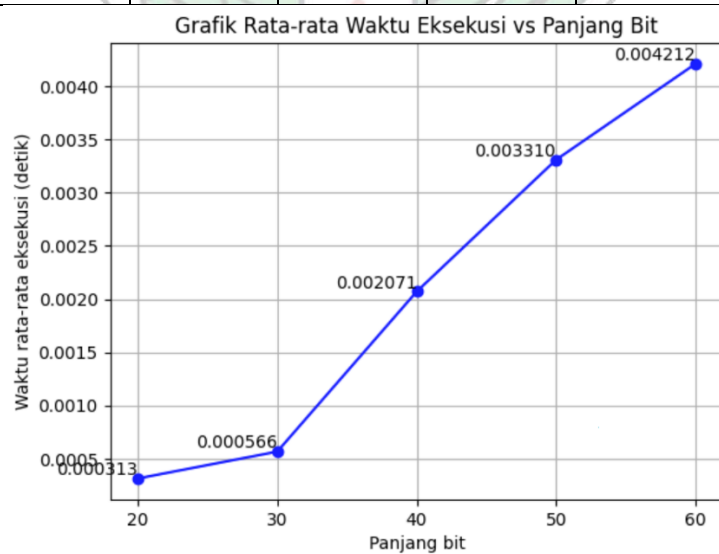
Panjang bit	Waktu Eksekusi (s)					
	Tes ke 1	Tes ke 2	Tes ke 3	Tes ke 4	Tes ke 5	Rata-rata
20	0.073180	0.664409	0.149047	0.335759	0.345478	0.313575
30	0.485957	0.755592	0.572543	1.088962	0.304639	0.641539
40	0.951634	0.838862	1.372772	1.090883	0.533814	0.957593
50	0.713242	0.383981	0.128788	0.415867	0.389194	0.406214
60	0.757708	0.665281	0.674330	0.663602	0.286050	0.609394



Gambar 4.14 Grafik Pengujian Pembangkitan Kunci Nagaty

Tabel 4.2 Pengujian Waktu Pembangkitan Kunci XRSA

Panjang bit	Waktu Eksekusi (s)					
	Tes ke 1	Tes ke 2	Tes ke 3	Tes ke 4	Tes ke 5	Rata - rata
20	0.000267	0.000326	0.000346	0.000325	0.000302	0.000313
30	0.000591	0.000410	0.000545	0.000592	0.000695	0.000566
40	0.002701	0.001672	0.001973	0.002070	0.001939	0.002071
50	0.003199	0.003510	0.003865	0.002664	0.003314	0.003310
60	0.003918	0.003570	0.003451	0.005008	0.005116	0.004212



Gambar 4.15 Grafik Pengujian Pembangkitan Kunci XRSA

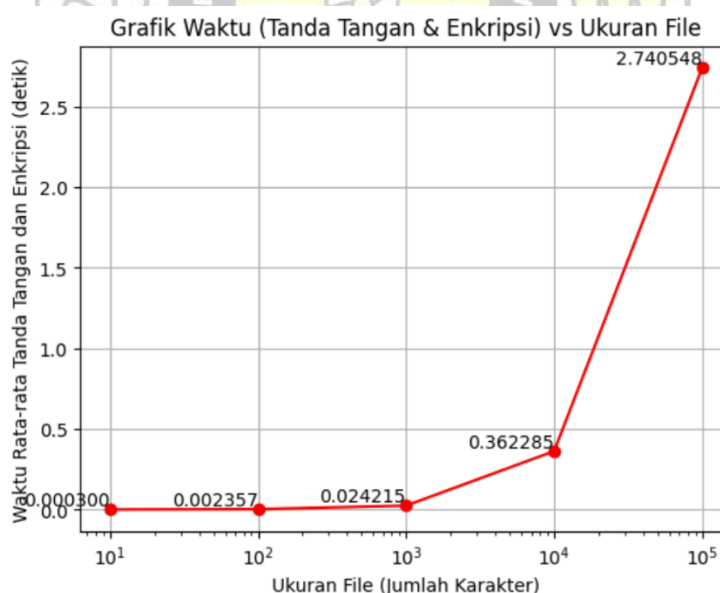
4.2.5.2. Pengujian *Real Running Time* pada Tanda Tangan dan Enkripsi

Pengujian real running time pada tanda tangan dan enkripsi dilakukan dengan kondisi panjang bilangan prima 50-bit, baik bilangan prima pada XRSA maupun Nagaty. Pengujian ini dilakukan pada *file* teks docx yang berisi 10 karakter, 100 karakter, 1000 karakter, 10000 karakter, dan 100000 karakter.

Tabel 4.3 Pengujian Waktu Tanda Tangan dan Enkripsi

Panjang Karakter	Waktu Eksekusi (s)					
	Tes ke 1	Tes ke 2	Tes ke 3	Tes ke 4	Tes ke 5	Rata - rata
10	0.000303	0.000292	0.000330	0.000293	0.000284	0.000300
100	0.002435	0.002392	0.002309	0.002314	0.002337	0.002357
1000	0.023928	0.024046	0.024583	0.024523	0.023996	0.024215
10000	0.267904	0.258601	0.424640	0.429224	0.431057	0.362285
100000	2.470089	2.820837	2.423477	3.538724	2.449613	2.740548

Pada tabel 4.3 dapat dilihat bahwa waktu eksekusi tanda tangan dan enkripsi meningkat seiring bertambahnya panjang karakter pesan yang diuji.



Gambar 4.16 Grafik Pengujian Waktu Tanda Tangan dan Enkripsi

4.2.5.3. Pengujian *Real Running Time* pada Dekripsi dan Verifikasi

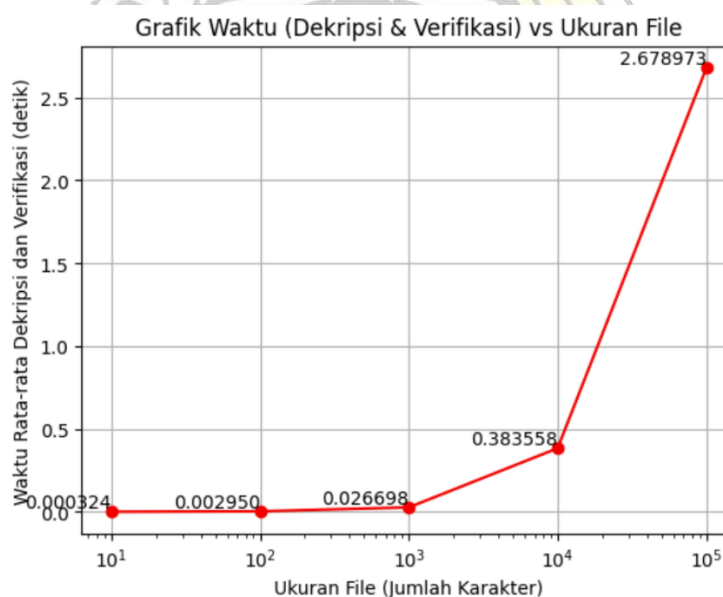
Pengujian real running time pada dekripsi dan verifikasi dilakukan dengan kondisi panjang bilangan prima 50-bit, baik bilangan prima pada XRSA maupun Nagaty. Pengujian ini dilakukan pada *file* teks docx yang berisi 10 karakter, 100 karakter,

1000 karakter, 10000 karakter, dan 100000 karakter yang telah di-tanda tangani dan dienkripsi.

Tabel 4.4 Pengujian Waktu Dekripsi dan Verifikasi

Panjang Karakter	Waktu Eksekusi (s)					
	Tes ke 1	Tes ke 2	Tes ke 3	Tes ke 4	Tes ke 5	Rata - rata
10	0.000368	0.000330	0.000307	0.000319	0.000295	0.000324
100	0.002720	0.002851	0.002450	0.002469	0.004260	0.002950
1000	0.025737	0.026001	0.025933	0.026517	0.029303	0.026698
10000	0.279857	0.446169	0.477333	0.446211	0.268221	0.383558
100000	3.272611	2.515453	2.545426	2.517317	2.544057	2.678973

Pada tabel 4.4 dapat dilihat bahwa waktu eksekusi dekripsi dan verifikasi meningkat seiring bertambahnya panjang karakter pesan yang diuji.



Gambar 4.17 Grafik Pengujian Waktu Dekripsi dan Verifikasi

4.2.6. Analisis Penyerangan Kunci dengan *Brute-Force*

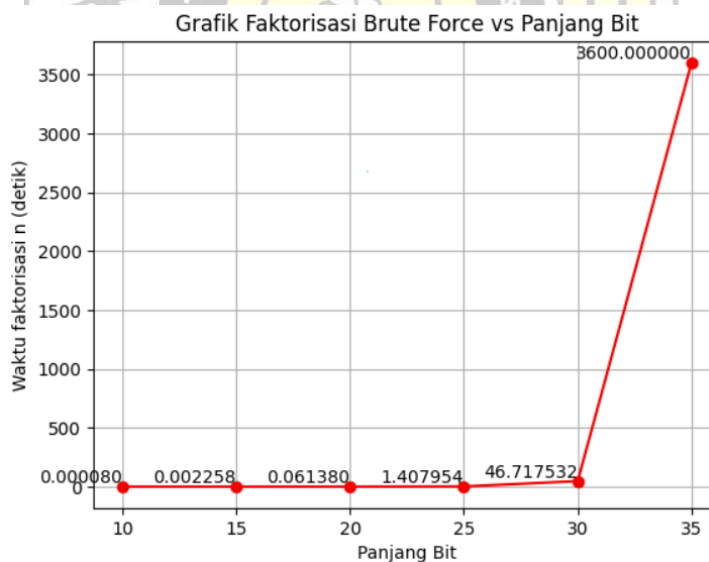
Analisis ini bertujuan untuk mengevaluasi ketahanan kunci terhadap serangan faktorisasi *brute-force*, di mana nilai n akan difaktorisasi untuk memperoleh nilai p_1 , p_2 , p_3 dan p_4 . Proses faktorisasi akan dilakukan pada berbagai panjang bit bilangan prima, yaitu 10-bit, 15-bit, 20-bit, 25-bit, 30-bit, dan 35-bit, guna mengukur tingkat kesulitan pempfaktoran seiring bertambahnya ukuran bilangan prima.

Selain itu, waktu analisis dibatasi hingga 1 jam (3600 detik) untuk setiap ukuran bit guna mengevaluasi efektivitas serangan dalam batas waktu yang wajar. Hasil analisis ini akan digunakan untuk memahami seberapa aman skema kunci terhadap serangan faktorisasi *brute-force* dalam berbagai ukuran bilangan prima.

Tabel 4.5 Analisis Faktorisasi *Brute-Force*

Panjang Bit	Waktu Eksekusi (s)					
	Tes ke 1	Tes ke 2	Tes ke 3	Tes ke 4	Tes ke 5	Rata - rata
10	0.000104	0.000104	0.000061	0.000068	0.000061	0.000068
15	0.001636	0.001908	0.001790	0.001907	0.004047	0.002257
20	0.065311	0.062737	0.061705	0.062709	0.054436	0.061379
25	1.709760	1.321342	1.493203	1.330512	1.184951	1.407954
30	50.23167	41.62799	51.77599	45.82857	44.12344	46.71753
35	3600	3600	3600	3600	3600	3600

Berdasarkan hasil percobaan pada tabel 4.5 dapat dilihat pada bit 35 waktu eksekusi melewati 1 jam dan menghasilkan grafik seperti di bawah ini.



Gambar 4.18 Grafik Faktorisasi *Brute-Force*

Pada gambar 4.18 dapat dilihat bahwa pada panjang 10-bit hingga 30-bit, waktu faktorisasi *brute-force* meningkat secara bertahap dengan faktor sekitar 23–33 kali lipat setiap kenaikan 5-bit. Hal ini menunjukkan bahwa meskipun jumlah kemungkinan faktor yang harus diuji bertambah, proses faktorisasi masih dapat diselesaikan dalam rentang waktu yang wajar. Namun, ketika panjang bit mencapai 35-bit, terjadi lonjakan drastis dalam waktu eksekusi hingga lebih dari 3600 detik,

menunjukkan bahwa sistem tidak mampu menyelesaikan faktorisasi dalam batas waktu yang diberikan.

Peningkatan eksponensial ini disebabkan oleh bertambahnya panjang bilangan n , yang merupakan hasil perkalian empat bilangan prima p_1 , p_2 , p_3 , dan p_4 . Setiap kenaikan 5-bit pada setiap bilangan prima menambah 20-bit pada n , sehingga jumlah kemungkinan faktor yang harus diuji bertambah sangat besar. Selain itu, semakin panjang bit suatu bilangan prima, semakin jarang kandidat yang tersedia untuk difaktorisasi, yang membuat pencarian faktor semakin lambat. Faktor inilah yang menyebabkan waktu faktorisasi untuk 35-bit meningkat drastis dibandingkan 30-bit, sehingga penggunaan kunci yang lebih panjang sangat direkomendasikan untuk meningkatkan keamanan terhadap serangan faktorisasi.



BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

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

1. Sistem berhasil menerapkan teknik *signcryption* dengan metode *sign-then-encrypt* pada *file* teks dengan format *.docx, memastikan keamanan dan keaslian pesan.
2. Implementasi teknik *signcryption* dilakukan dengan menandatangani plaintext dan verifikasi untuk mengecek keaslian pesan menggunakan skema Nagaty serta mengenkripsi pesan dan dekripsi pesan untuk mengembalikan pesan ke bentuk aslinya dengan algoritma XRSA. Pengujian dilakukan pada *file* teks *.docx dengan berbagai panjang karakter, yaitu 10, 100, 1.000, 10.000, dan 100.000 karakter.
3. Waktu pemrosesan tanda tangan dan enkripsi, serta dekripsi dan verifikasi, berbanding lurus dengan jumlah karakter dalam pesan. Semakin besar jumlah karakter yang diproses, semakin lama waktu eksekusi (*real running time*) yang dibutuhkan.
4. Sistem yang dirancang terbukti tahan terhadap serangan faktorisasi *brute-force*. Berdasarkan analisis keamanan, waktu eksekusi *brute-force* terhadap kunci 35-bit melebihi 1 jam, sedangkan pada kunci 30-bit, faktorisasi dapat dilakukan dalam rata-rata waktu 46 detik. Hal ini menunjukkan bahwa semakin panjang bit kunci, semakin tinggi tingkat keamanannya terhadap serangan *brute-force*.

5.2. Saran

Dari hasil pengujian dan analisis yang telah dilakukan, beberapa rekomendasi diberikan agar penelitian ini dapat lebih berkembang dan diterapkan secara lebih luas di masa mendatang.

1. Penelitian ini tidak menggunakan fungsi *hash* pada proses tanda tangan. Untuk meningkatkan keamanan tanda tangan digital, penulis berharap penelitian selanjutnya dapat menambahkan fungsi *hash* sehingga integritas tanda tangan lebih terjamin dan lebih sulit dimanipulasi.
2. Penelitian ini hanya menggunakan teks *file* dengan format *.docx untuk proses *sign-then-encrypt*. Penulis berharap penelitian selanjutnya dapat mengembangkan sistem agar mendukung format *file* lain, seperti PDF atau *file* dengan elemen gambar dan tabel, sehingga lebih fleksibel dalam penggunaannya.
3. Penelitian ini hanya menerapkan kriptografi kunci publik tanpa menggunakan *hybrid cryptosystem*. Untuk meningkatkan efisiensi dan keamanan sistem, penelitian selanjutnya dapat mengembangkan skema *hybrid cryptosystem* dengan mengombinasikan kriptografi kunci publik dan kriptografi kunci simetris. Pendekatan ini memungkinkan pemanfaatan keunggulan masing-masing metode, di mana kriptografi kunci publik digunakan untuk pertukaran kunci secara aman, sementara kriptografi kunci simetris digunakan untuk proses enkripsi dan dekripsi yang lebih cepat serta efisien.
4. Penelitian ini hanya dilakukan penyerangan faktorisasi *brute-force*, perlu dilakukan uji coba lebih lanjut dengan berbagai skenario serangan keamanan untuk memastikan ketahanan sistem terhadap ancaman seperti *man-in-the-middle attack* atau *chosen-ciphertext attack*.
5. Penelitian ini masih dilakukan pada komputer *stand-alone*. Oleh karena itu, diharapkan penelitian selanjutnya dapat diterapkan dalam kondisi nyata pada berbagai platform. Misalnya, pada *WhatsApp* untuk memastikan keamanan enkripsi pesan, pada perangkat pintar dengan keterbatasan daya komputasi agar tetap efisien, dalam sistem transaksi digital untuk mencegah akses tidak sah, serta dalam penyimpanan data di *cloud* agar lebih terlindungi dari serangan yang memanfaatkan kelemahan keamanan kunci.

DAFTAR PUSTAKA

- Arteta, A., Gomez, N., & Gonzalo, R. (2012). Solving Diophantine Equations with a Parallel Membrane Computing Model. *International Journal*.
- Budiman, M. A., & Rachmawati, D. (2018). Using random search and brute force algorithm in factoring the RSA modulus. *Data Science: Journal of Computing and Applied Informatics*, 2(1), Article 1. <https://doi.org/10.32734/jocai.v2.i1-91>
- Budiman, M. A., & Rachmawati, D. (2020). A tutorial on using Benaloh public key cryptosystem to encrypt text. *Journal of Physics: Conference Series*, 1542(1), 012039. <https://doi.org/10.1088/1742-6596/1542/1/012039>
- Budiman, M., Tarigan, J., & Winata, A. (2020). Arduino UNO and Android Based Digital Lock Using Combination of Vigenere Cipher and XOR Cipher. *Journal of Physics: Conference Series*, 1566, 012074. <https://doi.org/10.1088/1742-6596/1566/1/012074>
- Daepp, U., & Gorkin, P. (2011). Fermat's Little Theorem. In U. Daepp & P. Gorkin (Eds.), *Reading, Writing, and Proving: A Closer Look at Mathematics* (pp. 315–323). Springer. https://doi.org/10.1007/978-1-4419-9479-0_28
- 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), Article 1. <https://doi.org/10.32734/jocai.v8.i1-12226>

- Imam, R., Anwer, F., & Nadeem, M. (2022). An Effective and enhanced RSA based Public Key Encryption Scheme (XRSA). *International Journal of Information Technology*, 14. <https://doi.org/10.1007/s41870-022-00993-y>
- Nagaty, K. (2020). A public key cryptosystem and signature scheme based on numerical series. *SN Applied Sciences*, 2. <https://doi.org/10.1007/s42452-019-1928-8>
- Negre, C., & Plantard, T. (2017). Efficient regular modular exponentiation using multiplicative half-size splitting. *Journal of Cryptographic Engineering*, 7(3), 245–253. <https://doi.org/10.1007/s13389-016-0134-5>
- Pooja, & Yadav, M. (2018). Digital Signature. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, 3(6), 71–75. <https://doi.org/10.32628/CSEIT18364>
- Schneier, B. (2015). Protocol Building Blocks. In *Applied Cryptography, Second Edition* (pp. 21–46). John Wiley & Sons, Ltd. <https://doi.org/10.1002/9781119183471.ch2>
- Smart, N. P. (2016). *Cryptography Made Simple*. Springer International Publishing. <https://doi.org/10.1007/978-3-319-21936-3>
- Zhang, P., Li, Y., & Chi, H. (2022). An Elliptic Curve Signcryption Scheme and Its Application. *Wireless Communications and Mobile Computing*, 2022(1), 7499836. <https://doi.org/10.1155/2022/7499836>