Available online at https://ejournal.almaata.ac.id/index.php/IJUBI

Indonesian Journal of Business Intelligence

Volume 8 | Issue 1 | June (2025)

ISSN 2621-3915 (PRINT), ISSN 2621-3923 (ONLINE), Published by **Alma Ata University Press**

IJUBI

Indonesian Journal
---- of ----
Business Intelligence

IMPLEMENTASI ALGORITMA RIVEST-SHAMIR-ADLEMAN (RSA) MULTIPRIMA DAN CHINESE REMAINDER THEOREM (CRT) PADA PENGAMANAN PESAN TEKS

Adhi Yoga Pratama[1*], Hendro Wijayanto[2]

[1,2]Program Studi Informatika, Fakultas Teknik, Universitas Tiga Serangkai.

*adhiyogapratama34@gmail.com

Jl. KH Samanhudi No.84-86 Laweyan, Surakarta, Jawa Tengah, Indonesia

Article history: *Received: 19 April 2025; Revised: 20 June 2025; Accepted: 30 June 2025*

Abstract

RSA (Rivest-Shamir-Adleman) is a popular public-key cryptography algorithm in its implementation. The difficulty of factoring large numbers into prime factors is the basis for the security of the RSA algorithm. The initial stage of RSA is to choose random prime numbers *p* and *q*, and then calculate the modulus *n*. The value of the modulus *n* is proportional to the value of the prime. The value of modulus *n* is directly proportional to the security level and the high performance time of the algorithm. In the security aspect, to obtain a large modulus *n* value, RSA Multiprima is implemented with three random prime numbers (*p*, *q*, and *r*) in key generation. Handling the time efficiency of the decryption process, utilizing *Chinese Remainder Theorem* (CRT) to calculate the private key in several separate modules, so as to speed up the calculation in the decryption process. The results of this study show that the implementation of CRT in RSA Multiprima can speed up the decryption process, so it can be a resource efficiency solution in the implementation of the RSA Multiprima cryptographic algorithm. In this study, the implementation uses the *Python* programming language with version 3.12.2. Factorization test of the keys generated in this study using *Pollard's Rho* algorithm. This study is expected to provide benefits in the development of modern cryptography that is more efficient.

Keywords: *RSA, Chinese Remainder Theorem (CRT), Cryptography, Encryption, Decryption, Python.*

Abstrak

RSA (Rivest-Shamir-Adleman) adalah algoritma kriptografi kunci public yang popular dalam implementasinya. Sulitnya memfaktorkan bilangan besar menjadi faktor-faktor prima adalah dasar keamanan algoritma RSA. Tahap awal RSA adalah memilih bilangan prima acak *p* dan *q*, untuk selanjutnya dihitung modulus *n*. Besarnya nilai modulus *n* berbanding lurus dengan tingkat keamanan dan tingginya waktu performansi algoritma. Dalam aspek keamanan, untuk memperoleh nilai modulus *n* yang besar maka diimplementasikan RSA Multiprima dengan tiga bilangan prima acak (*p*, *q*, dan *r*) dalam pembangkitan kunci. Menangani efisiensi waktu proses dekripsi, memanfaatkan *Chinese Remainder Theorem* (CRT) untuk menghitung kunci privat dalam beberapa modul terpisah, sehingga dapat mempercepat kalkulasi dalam proses dekripsi. Hasil dari kajian ini menunjukan bahwa implementasi CRT pada RSA Multiprima dapat mempercepat proses dekripsi, sehingga dapat menjadi solusi efisiensi sumber daya dalam implementasi algoritma kriptografi RSA Multiprima. Dalam kajian ini, implementasi menggunakan bahasa pemrograman *Python* dengan versi 3.12.2. Uji faktorisasi terhadap kunci yang dihasilkan dalam kajian ini menggunakan algoritma *Pollard's Rho*. Kajian ini diharapkan dapat memberikan manfaat dalam pengembangan kriptografi modern yang lebih efisien.

Kata Kunci: *RSA, Chinese Remainder Theorem (CRT), Kriptografi, Enkripsi, Dekripsi, Python.*

Pendahuluan

Era digital ini, keamanan informasi menjadi isu krusial yang masih berkembang seiring dengan kemajuan teknologi informasi. Teknologi informasi adalah salah satu bentuk globalisasi yang masuk ke Indonesia, dan menyebar dengan sangat pesat. Bagi masyarakat, teknologi informasi membawa dampak positif, karena dapat menjadi sarana mempermudah kegiatan sehari-hari. Namun, akibat dari penyalahgunaan teknologi yang tidak bertanggungjawab, teknologi juga membawa pengaruh negatif yang dapat merugikan masyarakat itu sendiri [1]. Dengan perkembangan teknologi informasi, masyarakat mudah untuk mengakses apapun yang diinginkan, dengan secara sah atau bahkan ilegal, seperti serangan siber. Dengan semakin kompleks sifat serangan siber, kebutuhan keamanan yang efektif menjadi lebih penting dari sebelumnya. Salah satu cara terbaik dalam mengamankan data sensitif dari serangan siber dengan menggunakan pendekatan dan algoritma enkripsi [2]. Enkripsi merupakan bagian dari teknik kriptografi. Kriptografi dirasa mampu untuk menjaga keamanan informasi, meliputi kerahasiaan dan autentikasi [3].

Teknik kriptografi merupakan sebuah seni kuno menulis dalam kode rahasia, dimana penggunaan kriptografi tercatat pertama kali pada 1900 SM ketika juru tulis Mesir menggunakan hieroglif non-standar dalam penulisan sebuah prasasti [4]. Kriptografi tidak hanya ditujukan untuk mengamankan data dari pencurian atau modifikasi, tetapi juga dapat digunakan untuk mengautentikasi pengguna. Secara umum, terdapat tiga jenis skema kriptografi: kriptografi kunci rahasia (simetris), kriptografi kunci publik (asimetris), dan fungsi hash [4]. Dalam penelitian lain juga disebutkan, konsep matematika seperti algoritma, kunci enkripsi, dan fungsi hash digunakan dalam kriptografi untuk melindungi data dari ancaman serangan terhadap data maupun penyadapan [5].

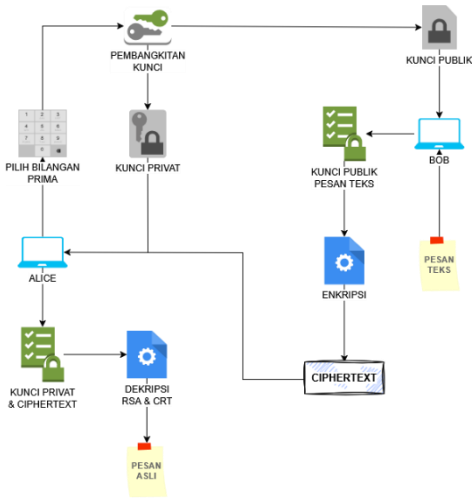
Menyediakan keamanan informasi yang meliputi enkripsi data, tanda tangan digital, dan dapat digunakan dalam pertukaran kunci, menjadikan RSA adalah salah satu algoritma kriptografi kunci publik yang populer digunakan [6]. Keamanan RSA bergantung pada sukarnya pemfaktoran bilangan komposit besar menjadi faktor prima, diasumsikan bahwa ketidakmungkinan komputasi dalam memfaktorkan bilangan komposit besar menjadi faktor-faktor prima [7]. RSA dalam konsep teori dasarnya menggunakan dua bilangan prima untuk menghasilkan kunci, namun untuk meningkatkan nilai keamanan pada RSA, digunakan tiga bilangan prima acak untuk menghasilkan kunci dengan nilai kemanan yang lebih tinggi.

Tingkat keamanan RSA berbanding lurus dengan besarnya bilangan yang digunakan, sehingga semakin kuat maka semakin lama prosesnya. *Chinese Remainder Theorem* (CRT) merupakah suatu teori matematis yang digunakan untuk mengubah eksponensiasi modular yang berukuran besar menjadi eksponensiasi modular yang relatif kecil [8]. Dengan menerapkan CRT pada RSA, proses dekripsi dipecah menjadi beberapa bagian modulo lebih kecil, sehingga dapat mempercepat proses dekripsi pada RSA.

Dalam kajian ini mengekplorasi penerapan algoritma RSA dengan tiga bilangan prima dipadukan dengan CRT untuk mempercepat proses dekripsi. Implementasi menggunakan *Python* 3.12.2 untuk memudahkan dalam memahami konsep system yang diterapkan. *Python* merupakan bahsa pemrogaman yang dikenal dengan kemudahan dan kesederhanaan dalam penggunaanya, dan juga menawarkan berbagai pustaka yang menunjang pengembang dalam menulis kode yang lebih efisien dan produktif [9]. *Python* melakukan eksekusi kode dengan baris demi baris tanpa dilakukan kompilasi terlebih dahulu, sehingga debugging menjadi lebih lebih mudah bagi pemula dibandingkan dengan bahasa pemrograman terkompilasi lainnya [10]. Kajian ini juga ditujukan untuk menguji efektivitas penerapan kombinasi RSA multi kunci dengan CRT dalam mengamankan pesan berupa teks.

Metode

Pendekatan penelitian kuantitatif dipilih sebagai metode penelitian. Penelitian kuantitatif diterapkan untuk mengkuantifikasi data dan menggeneralisasi apa yang ditemukan dari sampel yang diteliti dari berbagai perspektif. Oleh karena itu, pertanyaan seperti "berapa banyak", "berapa lama", dan "sejauh mana" sering terlibat dalam penelitian kuantitatif [11].
Dalam metode ini memuat rancangan system agar memudahkan dalam mengembangkan aplikasi kriptografi dengan mengimplementasikan algoritma RSA Multiprima dan CRT yang diterapkan dalam bahasa pemrograman *Python*. Flowchart menggambarkan proses pembangkitan kunci, proses enkripsi, dan proses dekripsi dari aplikasi. Desain flowchart menjadi pedoman dari setiap tahap dan fungsi dari aplikasi harus dapat berjalan sesuai dengan yang diharapkan sedemikian rupa.



Gambar 1. Proses RSA Multiprima – CRT

Gambaran umum dari ketiga langkah implementasi algoritma RSA Multiprima dan CRT adalah sebagai berikut:

A. Pembangkitan kunci
   Tahap pembangkitan kunci terdiri dari:
   1. Alice memilih tiga bilangan prima (p, q, dan r) secara acak
   2. Alice membangkitkan kunci dengan menghitung nilai
      i. $n = p.q.r$
      ii. $\varphi(n) = (p-1)(q-1)$
      iii. menentukan nilai e dan d
   3. Alice memperoleh pasangan kunci publik dan kunci privat

B. Enkripsi
   Pesan yang dikirimkan Bob akan dienkripsi dengan langkah sebagai berikut:
   1. Bob mengirimkan pesan teks kepada Alice, sehingga Bob mengenkripsi pesan dengan Kunci Publik Alice
   2. Bob mengirimkan *ciphertext* kepada Alice

C. Dekripsi
   Pesan yang diterima Alice dari Bob dienkripsi dengan Kunci Publik Alice, oleh karena itu Alice harus melakukan dekripsi terhadap *ciphertext* dari Bob dengan Kunci Private yang sesuai dengan Kunci Publik yang digunakan saat Bob melakukan enkripsi terhadap pesan asli. Langkah dekripsi yang dilakukan Alice sebagai berikut:
   1. Alice menerima *ciphertext* dari Bob
   2. Alice memilih Kunci Privat yang sesuai dengan Kunci Publik yang digunakan Bob
   3. Alice mengkombinasikan CRT untuk mempercepat proses dekripsi
   4. Jika, Kunci Privat yang dikombinasikan CRT sesuai dengan Kunci Publik yang digunakan Bob, maka Alice dapat mengetahui pesan asli yang dikirim oleh Bob.

Sistem RSA (Rivest-Shamir-Adleman) diterbitkan pada tahun 1977. Pengguna RSA, membuat dan mempublikasikan kunci publik yang dihasilkan dari dua bilangan prima besar yang harus dirahasiakan, siapapun dapat mengenkripsi pesan dengan kunci public tersebut, tapi pesan hanya dapat diterjemahkan oleh pengguna yang mengetahui bilangan prima tersebut [12]. Adapun tiga langkah dalam algoritma RSA, antara lain: pembangkitan kunci, enkripsi, dan dekripsi. Garis besar dari konsep system RSA adalah :

1. bilangan prima p dan q
2. hitung $n = pq$
3. hitung $\varphi(n) = (p-1)(q-1)$
4. pilih bilangan bulat e, dengan kondisi $1 < e < \varphi(n)$ dan $\gcd(e, \varphi(n)) = 1$
5. menentukan nilai d, dimana $e.d \equiv 1 \bmod \varphi(n)$
6. Kunci Publik KE = (n, e) boleh dipublikasikan, dan
   Kunci Privat KD = (n, d) yang harus dirahasiakan.

Fungsi enkripsi merupakan tahap mengubah plaintext menjadi *ciphertext* mengunakan pasangan kunci publik (n, e) yang sudah diketahui [13]. Fungsi enkripsi dihitung dengan persamaan:

$$C = M^e \ mod \ n \qquad (1)$$

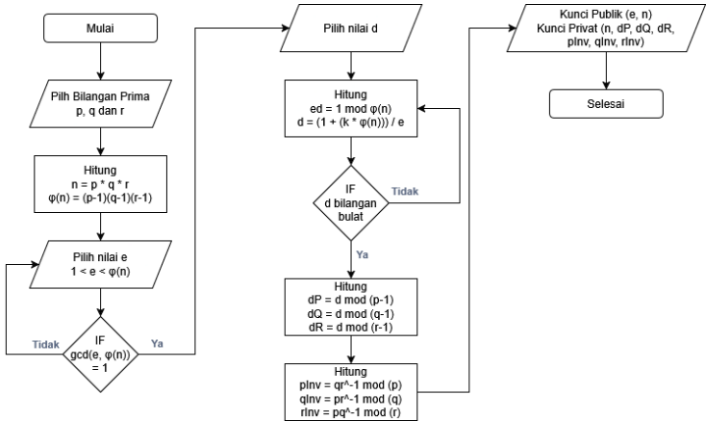
Pasangan kunci privat (n, d) digunakan dalam fungsi dekripsi, merupakan Langkah mengembalikan *ciphertext* menjadi pesan asli atau plaintext [13]. Fungsi dekripsi dihitung dengan persamaan:

$$M = C^d \ mod \ n \qquad (2)$$

Chinese Remainder Theorem (CRT) atau Teorema Sisa Cina adalah teorema yang memberikan solusi unik untuk kongruensi linier simultan dengan koprima modulus [14]. CRT muncul pertama kali di pada abad ke-3 masehi dalam sebuah karya matematika klasik dari Sun Zi, seorang ahli matematika di Tiongkok [15].

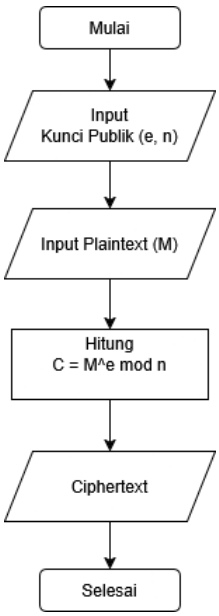
Jika $m_1, m_2, \ldots m_k$ adalah bilangan bulat positif relatif prima berpasangan, dan $a_1, a_2, \ldots a_k$ adalah bilangan bulat, maka kongruensi simultan $x \equiv a_1 \ (mod \ m_1)$, $x \equiv a_2 \ (mod \ m_2)$, … , $x \equiv a_k \ (mod \ m_k)$

RSA Mutiprima menggunakan tiga bilangan prima acak (p, q, dan r) dalam pembangkitan kunci. Nilai n dinyatakan dengan $n = pqr$. Nilai e dan d diperoleh dengan kalkulasi sama pada persamaan RSA tradisional.

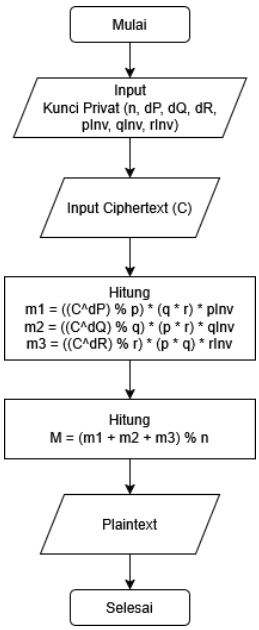


Gambar 2. Flowchart Pembangkitan Kunci

Fungsi enkripsi, pesan teks yang akan dienkripsi dikonversi menjadi bilangan desimal berdasarkan kode ASCII, untuk selanjutnya dibagi menjadi beberapa blok agar bisa dilakukan komputasi dari proses enkripsi pada RSA. Fungsi dekripsi menerapkan CRT untuk membagi proses dekripsi menjadi modulus-modulus yang lebih kecil agar dapat mempersingkat waktu pemrosesan.



Gambar 3. Flowchart Enkripsi



Gambar 4. Flowchart Dekripsi

Pembahasan

Pengembangan system dalam kajian ini menggunakan *Python* dengan versi 3.12.2. Python dipilih karena kemampuanya menangani operasi matematika atau visualisasi yang cukup komplek [16].

Data yang dienkripsi dalam kajian ini berupa teks dengan panjang 200, dan 500 karakter dengan menggunakan nilai modulus n = 3553. Tujuan dari kajian ini adalah menganalisis kinerja dari kombinasi algoritma RSA Multiprima – CRT.

Proses Pembangkitan Kunci RSA Multiprima - CRT

RSA tradisional menggunakan dua bilangan prima acak p dan q, namun dalam kajian ini menggunakan tiga bilangan prima acak p, q, dan r. Langkah-langkah dalam pembangkitan kunci:

1. Pilih bilangan prima acak p, q, dan r. Misal, p = 14897, q = 37337 dan r = 45131
2. Hitung $n = p.q.r$ = 14897 * 37337 * 45131 = **25102281421859**
3. Hitung $\varphi(n) = (p-1)(q-1)(r-1)$ = (14897-1)(37337-1)(45131-1) = **25099367937280**
4. Pilih nilai e, misalkan e = **3**, gcd(3, 25099367937280) = 1 (memenuhi syarat).
5. Nilai d diperoleh dari persamaan: $d = \frac{1+(k*\varphi(n))}{e}$ (3)

   dengan mencoba nilai-nilai k = 1,2,3…, dst sampai dengan nilai d adalah bilangan bulat [17], maka diperoleh d = **16732911958187**.
6. Menentukan nilai dP, dQ, dan dR dengan $dP = d \bmod (p-1)$, $dQ = d \bmod (q-1)$, dan $dR = d \bmod (r-1)$, sehingga diperoleh dP = **9931** , dQ = **24891**, dan dR = **30087**
7. Menentukan $pInv$, $qInv$, dan $rInv$ dengan persamaan:

   $$pInv = \frac{n}{p^{-1}} \bmod p,\ qInv = \frac{n}{q^{-1}} \bmod q, \text{ dan } rInv = \frac{n}{r^{-1}} \bmod r$$

   sehingga diperoleh $pInv$ = **8988**, $qInv$ = **10738**, dan $rInv$ = **4922**
8. Kunci Publik (e, n) = (3, 25102281421859) dan Kunci Privat (dP, dQ, dR, $pInv$, $qInv$, $rInv$, n) = (9931, 24891, 30087, 8988, 10738, 4922, 25102281421859)

Kunci dengan ukuran 2048 bit menawarkan tingkat keamanan yang baik untuk diimplementasikan, dan kunci dengan ukuran 3072-bit atau lebih, direkomendasikan untuk lingkungan keamanan yang tinggi [18].

Proses Enkripsi RSA Multiprima - CRT

Plaintext *M* diubah menjadi blok-blok $M_i, M_i, \ldots,$ pada setiap blok dienkripsi menjadi blok C_i [21]dengan mengacu pada (1). Plaintext yang akan dienkripsi "AKU ADALAH" dikonversi dengan sistem pengkodean ASCII, sehingga *M* = [65, 75, 85, 32, 65, 68, 65, 76, 65, 72]. Maka, $C_1 = 65^3 \ mod \ 25102281421859 = \mathbf{274625}$ begitu selanjutnya. Sehingga diperoleh *ciphertext* keseluruhan : *C* = [274625, 421875, 614125, 32768, 274625, 314432, 274625, 438976, 274625, 373248].

Nilai setiap blok data harus dalam [0, *n* - 1]. Contoh, diketahui nilai *M* = 07041111 dengan *n* = 3337, maka *M* tidak boleh memiliki lebih dari 4 karakter dalam setiap blok. Jika *n* - 1, yaitu 3337 – 1 = 3336 dan *M* dibagi ke dalam 5 karakter setiap blok, maka nilai m_1 = 07041, maka *M* > *n* - 1. Pembagian dalam blok ini harus diperhatikan, jika nilai mi melebihi dari nilai *n* - 1, ketika dienkripsi nanti korespondensinya tidak satu-ke-satu dengan *ciphertext* [19].

Proses Dekripsi RSA Multiprima - CRT

Dekripsi dilakukan dengan kunci private (*dP, dQ, dR, pInv, qInv, rInv, n*). Kunci private tersebut berbeda dengan kunci private pada RSA tradisional yang hanya terdiri dari gabungan (*d, n*). Jika pada proses sebelumnya sudah diketahui, bahwa,

$$M_1 = 274625^{167329119581 87} mod \ 25102281421859$$

yang merupakan representasi dari (2), memerlukan waktu yang cukup lama jika diproses dengan RSA tradisional. Oleh karena itu, implementasi CRT dimanfaatkan dalam proses ini untuk memecah nilai *d* menjadi modulus-modulus yang lebih kecil [20]. Proses dekripsi dengan CRT, sebagai berikut:

1. Diketahui *C* = 274625 dan kunci private sedemikian rupa (9931, 24891, 30087, 8988, 10738, 4922, 25102281421859)
2. Mencari nilai *mp*, *mq*, dan *mr*, dengan persamaan:

$$mp = (C^{dP} mod \ p) \ (qr)(pInv)$$
$$mq = (C^{dQ} mod \ q) \ (pr)(qInv)$$
$$mr = (C^{dR} mod \ r) \ (pq)(rInv)$$

   Maka, diperoleh *mp* = 984443502200340, *mq* = 469256752390790, dan *mr* = 177948037829770
3. Mencari nilai *M*, dengan persamaan:

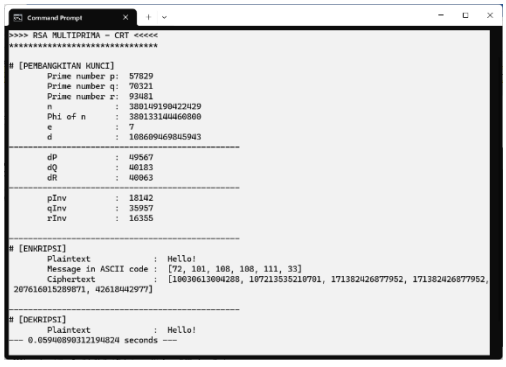
$$M = (mp + mq + mr) mod \ n$$

Sehingga diketahui M_1 = 65, begitu selanjutnya hingga *M* = [65, 75, 85, 32, 65, 68, 65, 76, 65, 72] dikonversi dengan sistem pengkodean ASCII sehingga menjadi karakter "AKU ADALAH", dan sesuai dengan pesan asli yang dikirim.

Hasil

Pengujian Progam

Program dijalankan melalui *command prompt* (CMD) yang ada di dalam sistem operasi Windows.



Gambar 5. Hasil Program

Pengujian Kecepatan

Pengujian ditujukan untuk mengetahui waktu rata-rata system menjalankan fungsi sesuai dengan yang diharapkan. Pengujian ini dilakukan dengan menjalankan program di dalam virtual environtment milik Python dan mengukur waktu setiap proses enkripsi dan dekripsi dengan Pustaka *time()* yang ada di dalam Python. Batasan dalam pengujian ini menggunakan teks dengan panjang 200 dan 500 karakter, merujuk pada penelitian [21] yang menguji data teks dengan rentang dari 100 sampai dengan 10.000 karakter. Modulus n = 3553 dihasilkan dari bilangan prima $p = 11$, $q = 17$, dan $r = 19$ dipilih dengan mempertimbangkan spesifikasi perangkat komputer yang digunakan untuk uji coba, yaitu:

- CPU Core i5
- RAM 8GB
- SSD 256GB

nilai modulus n berbanding lurus dengan lama waktu proses dan spesifikasi komputer yang dibutuhkan.

Tabel 1. Performansi Enkripsi RSA Multiprima

Algoritma	Modulus n	Karakter	Rata-Rata Waktu (*s*)
RSA Multiprima	3553	200	0,0012044
		500	0,0014552

Tabel 2. Performansi Dekripsi RSA Multiprima

Algoritma	Modulus n	Karakter	Rata-Rata Waktu (*s*)
RSA Multiprima	3553	200	0,0007228
		500	0,0023898

Tabel 3. Performansi Enkripsi RSA Multiprima – CRT

Algoritma	Modulus n	Karakter	Rata-Rata Waktu (*s*)
RSA Multiprima – CRT	3553	200	0,0012064
		500	0,0014608

Tabel 4. Performansi Dekripsi RSA Multiprima - CRT

Algoritma	Modulus n	Karakter	Rata-Rata Waktu (*s*)
RSA Multiprima – CRT	3553	200	0,0007186
		500	0,0008896

Pengujian Keamanan Kunci

Pengujian dilakukan dengan menggunakan algoritma *Pollard's Rho* untuk menguji faktorisasi kunci RSA yang dihasilkan. Algoritma *Pollard's Rho* adalah algoritma pemfaktoran dengan tujuan khusus untuk menemukan faktor kecil dari bilangan bulat komposit [22]. Misal, diketahui bahwa n merupakan bilangan yang akan difaktorkan, dimana $n = pq$ dengan p adalah faktor prima yang tidak lebih besar dari $\sqrt{n}$. Jika, sudah diketahui nilai p, maka faktor lainnya juga dapat diketahui, karena $n = pq$, dan nilai $\varphi(n)$ dapat dihitung. Karena kunci enkripsi e bukan rahasia, maka nilai dari kunci dekripsi d dapat dihitung dengan $e.d \equiv 1 \pmod{\varphi(n)}$ [12], [23].

Tabel 5. Waktu Pemfaktoran

Panjang n (digit)	Rata-Rata Waktu (*s*)
12	0,0002278
45	327,985758

Kesimpulan dan Saran

Berdasarkan pembahasan yang sudah disajikan, dapat ditarik beberapa simpulan bahwa algoritma RSA Multiprima – CRT dapat diimplementasikan dengan baik ke dalam system untuk mengamankan data. Penggunaan CRT pada RSA Multprima mampu mempersingkat waktu proses dekripsi dibandingkan dengan RSA tanpa CRT. Program dalam kajian ini dibangun dengan bahasa pemrograman Python, dan dapat berjalan dengan baik dalam proses pembangkitan kunci, enkripsi, dan dekripsi. Untuk menjaga keamanan algoritma RSA, sebaiknya menggunakan ukuran kunci 2048 bit atau lebih besar, karena semakin kecil nilai *n*, semakin mudah dilakukan serangan faktorisasi untuk menemukan faktor-faktor primanya.

Referensi

[1] Ni Komang Novia Widiasari and Emmy Febriani Thalib, "The Impact of Information Technology Development on Cybercrime Rate in Indonesia," *Journal of Digital Law and Policy*, vol. 1, no. 2, pp. 73–86, Jan. 2022, doi: 10.58982/jdlp.v1i2.165.

[2] Shuaib Ahmed Wadho, Areej Fatemah Meghji, Aun Yichiet, Roshan Kumar, and Farhan Bashir Shaikh, "Encryption Techniques and Algorithms to Combat Cybersecurity Attacks: A Review," *VAWKUM Transactions on Computer Sciences*, vol. 11, no. 1, pp. 295–305, Jun. 2023, doi: 10.21015/vtcs.v11i1.1521.

[3] Annisa Desianty and Imelda Imelda, "Systematic Literature Review: Cybersecurity by Utilizing Cryptography Using the Data Encryption Standard (DES) Algorithm," *Jurnal Teknik Informatika*, vol. 17, no. 1, pp. 30–39, Apr. 2024, doi: 10.15408/jti.v17i1.37256.

[4] Amosa Babalola, Ekuewa Bamidele, Olatunbosun Esther, and Oyetunji Olumayowa, "Cryptography: A Review," *International Journal of Advances in Engineering and Management (IJAEM)*, vol. 3, no. 10, pp. 1385–1391, Oct. 2021, doi: 10.35629/5252-031013851391.

[5] Agus Triono, Apri Setia Budi, Rafi Abdillah, and Wahyudi, "Implementasi Peretasan Sandi Vigenere Chipher Menggunakan Bahasa Pemrograman Python," *Jocitis-Journal Science Infomatica and Robotics*, vol. 1, no. 1, pp. 1–9, Sep. 2023.

[6] Muhammad Zaky Zachary, Sisilia Sylviani, and Edi Kurniadi, "Implementasi Algoritma RSA (Rivest-Shamir-Adleman) pada Kriptografi Klasik," *Mathematical Sciences and Applications Journal*, vol. 4, no. 2, pp. 54–59, Apr. 2024, doi: 10.22437/msa.v4i2.28863.

[7] Padmapriya V, Thanigaivelan A, Harish J, and Surya G, "Enhanced Integer Factor Optimization in RSA Cryptography using Shor Quantum," *International Journal of Research Publication and Reviews*, vol. 5, no. 5, pp. 2009–2016, May 2024, doi: 10.55248/gengpi.

[8] Aayush Shah, Prabhat Mahato, and Aadarsh Bhagat, "Enhancing Post-Quantum Cryptography: Exploring Mathematical Foundations and Comparative Analysis of Different Cryptographic Algorithm," *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, vol. 11, no. VIII, pp. 1626–1642, Aug. 2023, doi: 10.22214/ijraset.2023.55341.

[9] Tigus Juni Betri, Haya Nur Fadilah, and Galih Fajar Nugroho, "Data Science Analysis on UIN Raden Mas Said's Media Accounts," *International Journal of Science and Applied Science: Conference Series*, vol. 8, no. 1, pp. 76–85, 2024, doi: 10.20961/ijsascs.v7i2.96733.

[10] Vineesh Cutting and Nehemiah Stephen, "A Review on using Python as a Preferred Programming Language for Beginners," *International Research Journal of Engineering and Technology (IRJET)*, vol. 8, no. 8, pp. 4258–4263, Aug. 2021.

[11] Anahita Ghanad, "An Overview of Quantitative Research Methods,"*International Journal Of Multidisciplinary Research And Analysis*, vol. 6, no. 8, pp. 3794–3803, Aug. 2023, doi: 10.47191/ijmra/v6-i8-52.

[12] Arif Mandangan, Muhammad Asyraf Asbullah, Syed Farid Syed Adnan, and Mohammad Andri Budiman, "Comparative Of Rivest-Shamir-Adleman Cryptosystem And Its Four Variants Using Running Time And Memory Consumption Analysis," *Jurnal Teknologi (Sciences & Engineering)*, vol. 86, no. 4, pp. 181–190, 2024, doi: 10.11113/jurnalteknologi.v86.20723.

[13] Shovan Mondal and Aji Primajaya, "The Implementation Of Rsa Algorithm In Text Messages Encryption And Decryption," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 6, pp. 12620–12624, Dec. 2024, doi: 10.36040/jati.v8i6.12037.

[14] Muhammad Arief Budiman, Edi Kurniadi, Sukono, and Sisilia Sylviani, "Metode Pemecahan Sistem Kongruensi Linear," *Mathematical Sciences and Applications Journal*, vol. 5, no. 1, pp. 27–33, 2024, doi: 10.22437/msa.v5i1.38085.

[15] Elvis Adam Alhassan *et al.*, "On Some Algebraic Properties of the Chinese Remainder Theorem with Applications to Real Life," *Journal of Applied Mathematics and Computation*, vol. 5, no. 3, pp. 219–224, 2021, doi: 10.26855/jamc.2021.09.008.

[16] Panyawat Haarsa, "Solving Bernoulli's Equations Using Python: Enhancing Student Understanding Through Inquiry-Based Learning," *Unnes Journal of Mathematics Education*, vol. 13, no. 3, pp. 210–219, 2024, doi: 10.15294/ujme.v13i3.

[17] Triloka Mahesti, Arum Febriyanti Ciptaningtyas, and Agni Astungkara, "Perbandingan Penggunaan Algoritma Kriptografi DES, RSA, Modifikasi DES dan Modifikasi RSA untuk Penyandian Database," *Jurnal Ilmiah ILKOMINFO - Ilmu Komputer & Informatika*, vol. 8, no. 1, pp. 1–15, 2025, doi: 10.47324/ilkominfo.v8i1.294.

[18] Divya and Upasna Setia, "Evaluating RSA Key Length: Impact on Security Hardness and Computational Efficiency," *Journal Publication of International Research for Engineering and Management (JOIREM)*, vol. 10, no. 8, pp. 1–6, Aug. 2024.

[19] Asep Rizal Nurjaman, "Penanda Tanganan Dokumen Digital Pada Sistem Penyimpanan File Menggunakan Kombinasi Algoritma Sha3-512 Dan Rsa Untuk Mempertahankan Keaslian Data Dokumen," *JITTER (Jurnal Ilmiah Teknologi Informasi Terapan)*, vol. 10, no. 3, pp. 119–129, 2024, doi: 10.33197/jitter.vol10.iss3.2024.2200.

[20] Ishak Semuel Beno, "Analisis Teorema Sisa Cina dalan Dekripsi Data Text Terenkripsi RSA," *KOMBROF Jurnal Teknologi Informasi Papua (KJTIP)*, vol. 1, no. 1, pp. 40–44, 2023, doi: 10.31957/kjtip.v1i1.

[21] Deby Cynthia Rohara Manalu, Mutiara Enjelina, and Johannes Bastian Jasa Sipayung, "Perbandingan Performance Kriptografi RSA, RSA-CRT, Rabin dalam Proses Pengamanan Pesan Berbasis Teks," *Jurnal Quancom*, vol. 2, no. 2, pp. 23–30, Dec. 2024, doi: 10.62375/jqc.v2i2.417.

[22] Aminudin and Eko Budi Cahyono, "A practical analysis of the fermat factorization and pollard rho method for factoring integers," *Lontar Komputer: Jurnal Ilmiah Teknologi Informasi*, vol. 12, no. 1, pp. 33–40, Apr. 2021, doi: 10.24843/LKJITI.2021.v12.i01.p04.

[23] Gandhi Srushti and Ravi Gor, "Digital Image Encryption Using RSA And LFSR," *International Journal of Engineering Science Technologies*, vol. 6, no. 4, pp. 36–52, 2022, doi: 10.29121/ijoest.v6.i4.2022.351.