Progresif: Jurnal Ilmiah Komputer
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0877
p-ISSN: 0216-3284

Algoritme Kritpografi RSA Menggunakan Algoritme *Schönhage-Strassen*

Wahid Miftahul Ashari[1*], Jeki Kuswanto[2], Firman Asharudin[3]
[1,2]Teknik Komputer, Universitas Amikom Yogyakarta, Yogyakarta, Indonesia
[3]Teknik Informatika, Universitas Amikom Yogyakarta, Yogyakarta, Indonesia
**Email Corresponding Author*: wahidashari@amikom.ac.id

Abstract

The RSA algorithm (Rivest-Shamir-Adleman) is a public key cryptographic algorithm used in the process of encrypting symmetric cryptographic keys or private keys. The RSA algorithm relies on the difficulty of finding a large number to factor into prime factors. The RSA algorithm has a slower speed than the symmetric cryptographic algorithm, this means that the multiplication process of the numbers used is very large, more than 1024-bits. The number multiplication process requires special algorithms such as Montgomerry, Karatsuba, and Chinese Remainders. Based on this problem, this research attempts to develop an RSA algorithm library that uses the Schönhage-Strassen algorithm. The library will be developed using the SDLC (Software Development Life Cycle) methodology and the RAD (Ravid Application Development) method. After the library can be developed, the library will be tested with other multiplication algorithms, namely, Montgomerry, Karatsuba, and Chinese Remainder. The test results from this study have different values, where the Schönhage-Strassen algorithm has good performance, but not on all key variants.

Keywords: *Rivest-Shamir-Adleman; multiplication; Schönhage-Strassen; Software Development Life Cycle; Ravid Application Development.*

Abstrak

Algoritme RSA (*Rivest-Shamir-Adleman*) merupakan algoritme kriptografi kunci publik yang digunakan pada proses enkripsi kunci kriptografi simetris atau kunci privat. Algoritme RSA mengandalkan sulitnya mencari factor bilangan yang besar menjadi factor-faktor prima. Algoritme RSA memiliki kecepatan yang lebih lambat daripada algoritme kriptografi simetris, hal tersebut proses multiplication bilangan-bilangan yang digunakan sangat besar lebih dari 1024-bit. Proses multiplication bilangan membutuhkan algoritme-algortime khusus seperti *Montgomerry, Karatsuba*, dan *Chinese Remainders*. Berdasarkan pada masalah itu penelitian ini mencoba mngembangkan library algoritme RSA yang menggunakan algoritme *Schönhage-Strassen*. Library akan dikembangkan menggunakan metodologi SDLC (*Software Development Life Cycle*) dan metode RAD (*Ravid Application Development*). Setelah library dapat dikembangkan, library akan dilakukan pengujian tingkat eksekusi waktu dengan algoritme *multiplication* lainya yaitu*, Montgomerry, Karatsuba*, dan *Chinese Remainder*. Hasil pengujian dari penelitian ini memiliki nilai yang berbeda-beda, dimana algoritme *Schönhage-Strassen* memiliki kinerja yang baik, namun tidak pada semua varian kunci.

Kata Kunci: *Rivest-Shamir-Adleman**;** Multiplication; Schönhage-Strassen; Software Development Life Cycle*; *Ravid Application Development*.

1. Pendahuluan

Algoritme RSA merupakan algoritme yang ditemukan pada tahun 1977 oleh Rivest Shamir Adi yang merupakan ilmuwan matematika dari MIT [1]. RSA mengandalkan kekuatan pada sulitnya mencari faktor bilangan prima yang merupakan hasil kali pesan dan kunci privat maupun kunci public [2]. Algoritme Kriptografi RSA sudah memiliki riwayat terpecahkan hingga panjang kunci 829 bit pada Februari 2020 [3]. Penemu Algoritme Kriptografi RSA merekomendasikan bahwa panjang kunci yang ideal untuk Algoritme Kriptografi RSA adalah lebih dari 1024 bit. Semua algoritme kriptografi memang sedang dalam masalah yang cukup

besar terlebih dengan kemunculan Komputer yang berbasis *quantum*. Komputer *quantum* diramalkan dapat memprediksi 45 jam ke depan lalu-lintas yang ada. Komputer *quantum* juga diramalkan dapat memecahkan AES-256 bit hanya dalam waktu 10 menit [4].

Berdasarkan masalah tersebut maka kerahasiaan proses komunikasi dalam dunia internet mulai terancam. Ancaman paling besar adalah pada algoritme kriptografi enkripsi simetris yang sampai saat ini masih digunakan untuk enkripsi pesan pada beberapa layanan komunikasi internet [5]. Salah satu cara untuk mengantisipasi hal tersebut adalah dengan mempersiapkan algoritme kriptografi yang sangat kuat seperti *quantum crypto* atau algoritme kriptografi kunci publik sebagai pengganti algoritme kriptografi kunci simetris.

Quantum Kriptografi memiliki masalah tentang efisiensi dan keterbatasan sumber daya apabila diimplementasikan pada enkripsi pesan dengan mempertimbangkan pengguna enkripsi pesan terbesar adalah smartphone. Pada algoritme kriptografi kunci publik masalah terbesar adalah proses enkripsi dan dekripsi jauh lebih lambat daripada algoritme kriptografi pesan simetris pada hal ini adalah AES-256. Algoritme kriptografi kunci publik memiliki masalah pada perhitungan matematika yang melibatkan bilangan-bilangan yang sangat besar. Seperti halnya Algoritme Kriptografi RSA-2048 bit memiliki kunci sebuah bilangan prima dengan panjang lebih dari 100 digit, bilangan tersebut juga akan dipangkatkan dengan bilangan yang berukuran sama panjang. Sehingga dibutuhkan algoritme khusus untuk menangani proses perhitungan tersebut.

Beberapa algoritme untuk menangani proses perhitungan tersebut diantaranya adalah: *Montgomerry, Karastuba, Chinese-Remainder, Barett*, dan *Schönhage–Strassen*. Algoritme *montgomerry, Karastuba*, dan *Chinese Remainder* sudah pernah diimplementasikan pada Algoritme Kriptografi RSA [6][7][8]. Berdasarkan penleitian yang sudah ada algoritme *Montgomerry* memiliki kinerja terbaik untuk menyelesaikan proses perhitungan pada Algoritme Kriptografi RSA. Beberapa penelitian menyimpulkan bahwa Algoritme *Schönhage–Strassen* memiliki kinerja yang lebih baik dari algoritme perkalian lainya [9][10] namun belum ada penelitian yang pernah menulis bahwa algoritme ini dapat diterapkan pada algoritme RSA.

Penelitian kami bertujuan untuk menguji dan mengimplementasikan Algoritme *Schönhage–Strassen* untuk penyelesaian proses perhitungan pada Algoritme RSA. Penelitian ini nantinya akan mengembangkan sebuah library algoritme RSA yang menggunakan algoritme *Schönhage–Strassen*. Setelah library dapat dikembangkan akan dilakukan pengujian pada library tersebut dan dibandingkan dengan algortime-algoritme perhitungan atau perkalian lainya yang sudah disebutkan pada alinea sebelumnya. Setelah hasil pengujian dianalisa diharapkan pada penelitian ini akan menghasilkan libarary yang memiliki kinerja lebih baik dari library algoritme RSA sebelumnya.

2. Tinjauan Pustaka

Penelitian dikembaangkan berdasarkan penetlitian-penelitian terdahulu yang relevan, seperti disajikan berikut:

1) Analytical Comparison of RSA and RSA with Chinese Remainder Theorem [8]

Penelitian ini membahas tentang implementasi Algoritme *Chinese Remainder* pada Algoritme Kriptografi RSA. Penelitian ini menyimpulkan bahwa Algoritme *Chinese Remainder* tidak efektif jika diterapkan pada Algoritme Kriptografi RSA. Algoritme Kriptografi RSA dengan *montgomerry* memiliki kinerja yang lebih baik jika dibandingkan dengan Algoritme Kriptografi RSA dengan menggunakan Algoritme *Chinese Remainder*.

2) *Implementation of RSA Algorithm Based on RNS Montgomery Multiplication [6]*

Penelitian ini membahas tentang impelementasi Algoritme Kriptografi RSA dengan menggunakan Algoritme RNS *Montgomerry*. Hasil dari penelitian ini adalah Algoritme Kriptografi RSA dengan menggunakan Algoritme RNS *Montgomerry* memiliki kinerja yang lebih stabil.

3) *Review Of Fast Multiplication Algorithms for Embedded Systems Design* [10]

Penelitian ini berfokus pada perbandingan algoritme yang menyelesaikan perkalian pada bilangan besar. Penelitian ini tidak berfokus pada Algoritme Kriptografi RSA, namun hanya berfokus pada komputasi matematika yang memiliki kesamaan dengan komputasi matematika pada Algoritme Kriptografi RSA.

4) *Fast Integer Multiplication Using Generalized Fermat Primes [11]*

Penelitian yang berjudul “Fast Integer Multiplication Using Generalized Fermat Primes” ini meneliti tentang perkalian bilangan prima dengan ukuran yang sangat besar. Penelitian ini menyimpulkan bahwa algoritme *Schönhage-Strassen* efektif digunakan untuk perkalian ketika bilangan kurang dari 2^{40}.

5) *Schönhage-Strassen Algorithm with MapReduce for Multiplying Terabit Integers [9]*

Penelitian “*Schönhage-Strassen Algorithm with MapReduce for Multiplying Terabit Integers*” membahas tentang algoritme *Schönhage- Strassen* yang digunakan untuk memproses bilangan terabit *interger*. Penelitian ini menyimpulkan bahwa algoritme *Schönhage-Strassen* mampu menghitung bilangan perkalian bilangan bulat dengan ukuran 8 Tera Bit.

6)*High speed implementation of RSA algorithm with modified keys exchange [12]*

Penelitian ini membahas tentang peningkatan kecepatan algoritme RSA dengan memodifikasi pada tingkat pertukaran kunci. Pertukaran kunci dibuat lebih sederhana jika dibandingkan dengan algoritme RSA pada versi aslinya. Kecepatan Algoritme RSA pada penelitian meningkat namun belum diteliti tingkat keamanan pada algoritme RSA dengan varian ini.

7) *Implementation of efficient method of RSA key-pair generation algorithm [13]*

Penelitian ini berfokus untuk menghitung kunci yang dibangkitkan atau di generate oleh algoritme RSA pada tahap kedua, algoritme RSA memiliki masalah untuk menemukan kunci yang cocok Ketika terjadi pertukaran kunci, pada penelitian ini sudah dapat dikembangkan sebuah proses pembangkitan kunci namun tidak dibandingkan dengan penelitian sebelumnya.

8) Implementation of RSA Signatures on GPU and CPU Architectures [14]

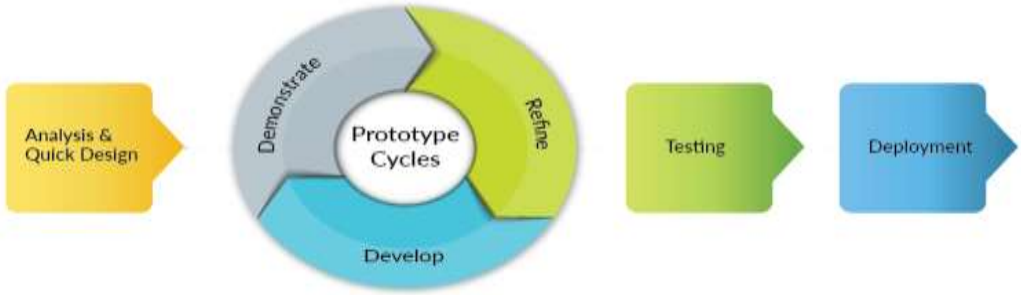
Penetitian ini mencoba mencari algoritme multiplication terbaik untuk diterapkan pada proses RSA agar dapat diimplementasikan pada CPU dan GPU. Hasil dari algoritme ini menyimpulkan bahwa CPU dan memproses algoritme RSA jika dibandingkan dengan GPU. Pengujian CPU dan GPU juga menggunakan beberapa varian algoritme RSA.

State of the art penelitian kami adalah menguji pengimplementasian Algoritme *Schönhage–Strassen* untuk menyelesaikan proses perhitungan pada Algoritme RSA

3. Metodologi

Penelitian ini mengembangkan sebuah library memodifikasi algoritme *Schönhage-Strassen* untuk menyelesaikan permasalahan pada Algoritme Kriptografi RSA. Metodologi pengembangan yang dipakai pada penelitian ini adalah metodologi SDLC (*Software Development Life Cycle*) [15]. Pemilihan menggunakan metodologi SDLC pada penelitian ini karena penelitian ini nantinya akan menghasilkan sebuah *library* algoritme Kriptografi RSA menggunakan algoritme algoritme *Schönhage-Strassen* yang merupakan sebuah library yang berbentuk perangkat lunak.

Pada metodologi SDLC terdapat banyak sekali metode-metode yang digunakan untuk mengembangkan sebuah sistem atau perangkat lunak. Sedangkan metode pada penelitian ini adalah menggunakan metode RAD *(Rapid Application Model)* [16]. Penelitian ini menggunakan metode tersebut karena penelitian ini merupakan sebuah penelitian yang mengembangkan sebuah perangkat lunak untuk keperluan penelitian dan bukan untuk proses bisnis dari pelanggan [17].



Gambar 1 Metode RAD

Setelah tahap develop selesai maka tahap selanjutnya adalah pengujian. Pengujian pada penelitian ini dibagi menjadi dua bagian utama yaitu pengujian fungsional dan pengujian perbandingan. Pengujian menggunakan pada satu buah laptop yang sama dengan spesifikasi prosesor core i5 dan RAM 8GB. Berikut penjelasan singkat tentang pengujian-pengujian yang akan dilakukan].

3.1 Pengujian fungsional

Pengujian yang pertama merupakan pengujian tentang kinerja library apakah sudah menghasilkan output yang dihasilkan atau hasil yang benar. Pengujian pertama dilakukan dengan menguji library dengan proses pertukaran kunci dan percobaan proses enkripsi dan dekripsi pada tiga varian kunci algoritme yaitu 1024-bit, 2048-bit, dan 4096-bit.

3.2 Pengujian Perbandingan

Pada pengujian tahap kedua ini algoritme RSA dengan algoritme *Schönhage-Strassen* akan dibandingkan dengan algoritme RSA yang menggunakan algoritme-algoritme lainya yang ada pada tabel dibawah ini.

Tabel 1 Algoritme yang diujikan

No	Algoritme
1	*Schönhage-Strassen*
2	*Montgomerry*
3	*Karatsuba*
4	*Chinese Remainders*

4. Hasil dan Pembahasan

Pada pengujian pertama library akan diuji untuk proses enkripsi dan dekripsi. Pengujian akan dilakukan dengan 3 varian kunci yang akan disajikan dalam bentuk tabel. Berikut adalah tabel hasil pengujian yang telah dilakukan.

Tabel 2 Hasil pengujian fungsi library

No	Proses Pengujian	Pesan	Enkripsi	Dekripsi
1	Enkripsi 1024-bit	Negara Republik Indonesia	MIICXAIBAAKBgQC9QCWAdSElMHiznjf5x/kiQTYXAt50sMqY9Z61gs46gMlIham6I4J3wae3zVdSLajCBsilPKfbBgtKXupkOkxLCOQ/MUDq8LxX2Aol2uvwxRNmv8DMrcst+Fw4QDDwZO/3iI5n/cnXm41CbfR7mvx63AUVT80lCYVrnaYj/9QphQIDAQABAoGAb1Z5uaUvJs3pdOK0U8MOEtUCmW8aqNQJnQuDn42g/lIeGQhp3hReAU5w+XVY1gP+sIeFK5EPv49CvTIx+11A3DCOl4zQqkjhrAKB+UJPfjoBsqmcWlHq3NKZt21g5lO4jFxbiIOKkp7z6/D7iG8eW/VaVYtBuTmp63gHlP4QlNECQQDOfVsHGB17LiNh	Negara Republik Indonesia
2	Enkripsi 2048-bit	Negara Republik Indonesia	MIICXAIBAAKBgQCVoeKMLTghhLRSXbJ25KM9LHfQ17vruKEr2ysyinl3klypuLTsmZAxX727kdkJ9KhJKkOnIIN78fDhKndS00ouXGJSt9zR4Tkmf10lntRxXsLSGIJTGMKh6z9S9LAm+sHciunU/SX+OvqYvUvVbGy5DzJgZ4lUtxH8okYyZ7mHiQIDAQABAoGAJsJYPJc6AvStAOjkYYe4ARCTuLUpKFYa6ZFMDztvW1m+RvhhZSgiAagbLYsJ/VoyfUFFZPU3GdQB4VFutzWj8Rj3zGkL33wu6Ev70KOE5HNe7qmQKC3cxc0dTEJgHGu9K4LXB/KnzigzK08GPfVvoa9NxaA+XM6ajT48Q7hBRkECQQDDKdxXovtPWvMn1eqFd+ORilL5vjyhiQ3sqhzHocAEg6wT7dti73BLst2+49gKB8ffKQ16KFn/SQtQHYA+KfmHAkEAxEad7vCZ4vaAoQ6kERMVhMj+Y4yRERaJeGIo6RH	Negara Republik Indonesia

No	Proses Pengujian	Pesan	Enkripsi	Dekripsi
			MXOJklN1pK3ft9vsw8o9dPQguuwXfJFS37biZHt0lL4R6bwJBALGJiRpn9znPQNeCL9/OmkQxkx4BwuIn7XTXo2qvaCAzL/zJKgQsUPlr83DNkCA/x8DodYwk3QnwhswQqjLiuE0CQE2SDyb4xgjSHUIJ1EO/RBjnispzKn4TZx9BxCNg2mC3RMMkwkTgsBaVbeHfUR0m17kod+kW5l44P6PYCLM/61cCQCr/k+kx5BmuEWyU23XkvJCtlTSNNloCCrKha80SOZDnxN26mrvS5XXfQA2h3yZwuiHxxnD+VwOXbC4HTZMqcSo=	
3	Enkripsi 4096-bit	Negara Republik Indonesia	MIIEpAIBAAKCAQEAq1a/bGop33yC+msyeVTYh1xJ4fQBjl6Q0/OvGkmEswofNka6OxEiGNOKuho5nXrH3itCT6qUcNSsD/3Y8hFXQW8Rrg8d3fLKc91XLUN7cgJ0pq0UrW+rU94UUlKyZ+mxrVLfokLH7yIhPgrxMXfRs5SRXc9UMISGCdN4TDKdSVQUL8UeomBHIweHoE7W6kh8wefAsFeqH4GnusXkQTfgdzwbOe11/0DqYn7vUgaRZWPED7svPta0bWUG9YUez/F/4GKh5gP+5eNsO0M217WhCaq4hMHQ0g/OX44zEzo/JLdfOCEDQHgXsOUFYQKugeRRxJ2xTTq3nzSKcoG9YhbUQIDAQABAoIBAHIjhb2pM1/HyFEd3C/rj5sMpcM6E1dOM9ieOwWd+CJ1GbnEgAcdMMOtGHtFQSBhTIf59WCw19YaNs0oY79OQfw7C2ISe+kA9lEdIBftHm0iqabnIiEvhMiH5q24nVSP32cBEC09o4bwC4kTLP2xPicNT5FYrlLr1YbyMu8/wHu5yelmiTWYHjPTfmrbBnhKo69tLXZ9mahIHyNt tGH3HkOf+Vzkj47ngGs2b/CzBnzitP0TPLoUuHlbnKssHoA8aLGl+JhVL5qhTrYZKa5Do+Tm4FlCyuLwaYgqwOHPKjlpIvXmFk873zlPgwfdmr/6v0l73SQ+K62T0+px 9D5OKAkCgYEAzVZvx5G8nUXfvVzIbWorP09BaqkltXk2UXoWFhVhbx5CmKtvMqU0h8USD6z5zsEDmsJcKeGxmeIVA6UmRIwHmJ5bGaqI0BwUFL6lHDzJGRQfA38c0hi2jb4Vd36zl/w2RY0BnWdebYYSP5OZjsLMCThrcR4Oe3nc0kKvqwekWQMCgYEA1ZzgGHWrdL5+NI28sLLrPptHSLkBU2wKq1Sg48pzRdNXX3Sk7AWmQVsxXNQvVs/dkveU50BG6Y9l/qmaYFJSyBOsnS4WFkbT4VMmq5U699NkhEkJiKTzK27h7exLnxRxZVyjJeXDc0kAuTEfERtnNYSB2pm/5uLf5+UZK6tDqBsCgYEAoUN3OlccuK+9zixmRHgTSv9lsuzw7pspq8ekPOfBw9ESIHDTPgU9QjvluWXyMwynVg268DGOl/pU4q37Ze5LsPL9PSUPmdHTe49WgNzEWuvQh6B29BHwnVVh6p0Z	Negara Republik Indonesia

Berdasarkan tabel 2 diatas bahwa library yang telah dikembangkan sudah berhasil untuk digunakan dalam proses enkripsi dan dekripsi. Pengujian dengan 3 kunci yang berbeda sudah berjalan dengan benar, pesan yang telah di dekripsi sudah sama dengan pesan yang belum di enkrpisi. Pada algoritme RSA panjang pesan dapat menjadi berlipat kali ukuranya menyesuaikan dengan panjang kunci yang digunakan. Pada panjang kunci 1024-bit pesan rahaisa dapat menjadi berukuran 1024-bit juga.

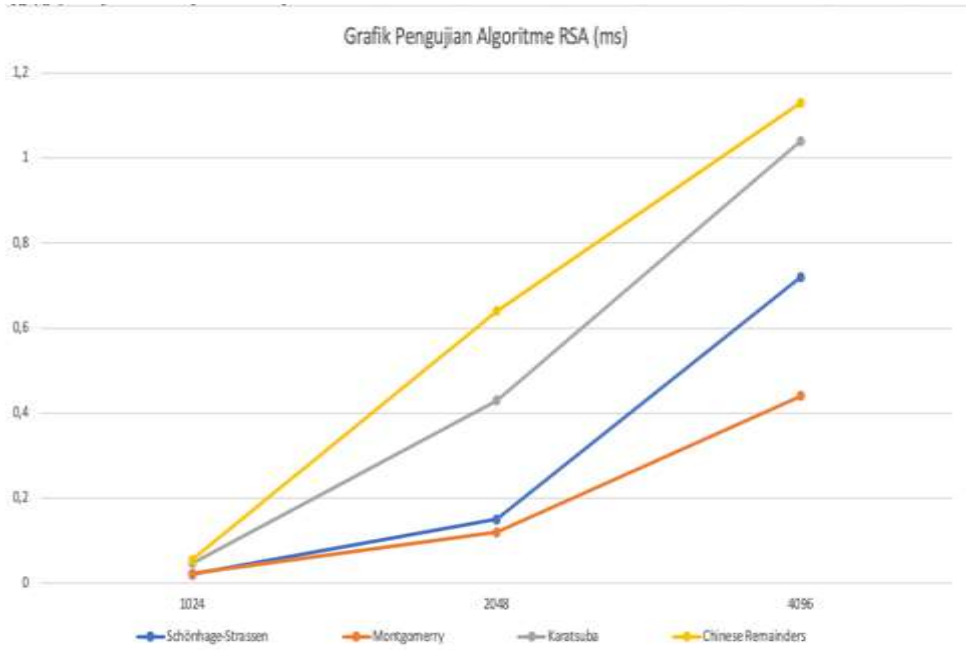
3.1 Pengujian Perbandingan Algoritme Perkalian

Pada bagian ini akan dijelaskan dan ditunjukan hasil dari pengujian yang kedua yaitu pengujian kebutuhan waktu eksekusi dari semua algortime yang diujikan pada tabel 3. Hasil pengujian akan dikumpulkan dan dirata-rata hasilnya. Berikut merupakan hasil dari pengujian perbandingan algoritme yang diujikan.

Tabel 3 Hasil pengujian algoritme RSA dengan beberapa model algoritme perkalian

No	Algoritme	Key 1024-Bit	Key 2048-Bit	Key 4096-Bit
1	***Schönhage-Strassen***	**0.21 s**	**0.15 s**	**0.72 s**
2	*Montgomerry*	0.23 s	0.12 s	0.44 s
3	*Karatsuba*	0.47 s	0.43 s	1.04 s
4	*Chinese Remainders*	0.55 s	0.64 s	1.13 s

Pada Tabel 3 pengujian perbandingan ke 4 algoritme RSA telah dilakukan, adapaun nilai pada masing-masing algoritme RSA dapat dilihat pada tabel 3 pada varian 1024 algoritme RSA dengan algoritme *Schönhage-Strassen* menjadi varian yang paling cepat. Sedangkan pada Panjang kunci 2048bit dan 4096 bit, algoritme RSA dengan varian algoritme *Montgomerry* menjadi varian yang paling cepat.



Gambar 2. Grafik Hasil Perbandingan Algoritme RSA Dengan Beebrapa Algoritme Perkalian

Pada penelitian ini telah dikembangkan dan diuji sebuah algoritme RSA dengan menggunakan algoritme *Schönhage-Strassen* sebagai algoritme inti pada proses perkalian interger yang ada pada algoritme RSA. Berdasarkan tabel 4 dan gambar 7 dapat dilihat bahwa algoritme *Schönhage-Strassen* memiliki kebutuhan waktu antara 0.02 s sampai 0.72 s untuk melakukan enkripsi mulai dari panjang unci 1024-bit sampai dengan 4096-bit. Algoritme RSA dengan menggunakan algoritme *Schönhage-Strassen* dapat berjalan lebih cepat jika dibandingkan dengan Algoritme RSA yang menggunakan algoritme *Chinese remainder* pada penelitian [8]. Namun hasil algoritme RSA dengan menggunakan algoritme *Schönhage-Strassen* tidak lebih baik dari hasil algortime RSA yang menggunakan algoritme montgomerry [6]. Algoritme RSA dengan *Schönhage-Strassen* memiliki hasil yang lebih cepat dari algoritme RSA *Montgomerry* pada Panjang kunci 1024-bit, namun perbedaan kecepatan tersebut tidak signifikan yaitu hanya sebesar 0.02 s. Algoritme *Karatsuba* yang digunakan pada algoritme

RSA juga belum memiliki hasil yang lebih baik dari RSA dengan *Montgomerry.* Hasil ini sejalan dengan penelitian [14], dimana dari semua pengujian hasil terbaik masih didapatkan oleh algoritme RSA dengan proses *Multiplication interger* menggunakan algoritme *Montgomerry*.

5. Simpulan

Algoritme *Schönhage-Strassen* dapat diimplementasikan pada algoritme RSA. Algoritme *Schönhage-Strassen* bukan merupakan algoritme paling cepat dan efektif pada kasus algoritme RSA. Algoritme paling cepat dan efektif adalah algoritme *Montgomerry*, sehingga penelitian ini masih merekomendasikan algoritme RSA dengan algoritme *Multiplication montgomerry* yang digunakan untuk proses enkripsi dekripsi pada proses pengamanan informasi dan data. Untuk Pengembangan masa mendatang, kami menginginkan bahwa pada penelitian lanjutan menghasilkan sebuah library algoritme RSA yang memiliki kinerja yang sangat cepat dengan mengganti atau meningkatkan algoritme *Montgomerry* yang pada algoritme RSA digunakan sebagai algoritme perkalian interger. Hal ini berkaitan dengan belum digunakannya algoritme RSA dan algortime kunci publik lainya dalam proses percakapan pesan singkat karena komputasi yang berat dan waktu eksekusi yang cukup lama.

Daftar Referensi

[1] L. M. Batten, “The RSA Scheme,” in *Public Key Cryptography: Applications and Attacks*, IEEE, 2013. doi: 10.1002/9781118482261.ch4.

[2] N. Thiranant, Y.S. Lee, & H. Lee, "Performance comparison between RSA and elliptic curve cryptography-based QR code authentication". In *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops, IEEE,* pp. 278-282, 2015

[3] J. Kilgallin, & R. Vasko, "Factoring RSA keys in the IoT era. In *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA),* IEEE, pp. 184-189, 2019

[4] L.C.C. Garcıa, "Can Schönhage multiplication speed up the RSA decryption or encryption? *Technische Universität Darmstadt*, pp. 1-6, 2017

[5] D. Mahto and D. K. Yadav, “RSA and ECC: A Comparative Analysis,” vol. 12, no. 19, p. 9, 2017.

[6] H. Nozaki, M. Motoyama, A. Shimbo, & S. Kawamura, "Implementation of RSA algorithm based on RNS Montgomery multiplication. In *Cryptographic Hardware and Embedded Systems—CHES 2001: Third International Workshop Paris, France, May 14–16, Proceedings,* Springer Berlin Heidelberg, *3,* pp. 364-376, 2020.

[7] Z. Gu and S. Li, "Optimized Interpolation of Four-Term Karatsuba Multiplication and a Method of Avoiding Negative Multiplicands," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 3, pp. 1199-1209, March 2022, doi: 10.1109/TCSI.2021.3126797.

[8] A. Mantri, A. Razaque, H. Makwana, P. Parekh, and T. Soomro, “Analytical Comparison of RSA and RSA with Chinese Remainder Theorem,” *JISR Comput.*, vol. 14, pp. 16–21, Jan. 2016, doi: 10.31645/jisrc/(2016).14.1.0003.

[9] Sze, T. W. “Schönhage-Strassen algorithm with MapReduce for multiplying terabit integers.” *Symbolic-Numeric Computation. pp. 54-62,* 2012.

[10] M.M. Asad, I. Marouf, Q.A. Al-Haija, "Review of fast multiplication algorithms for embedded systems design". *International Journal of Scientific & Technology Research*, vol. 6, no. 8, pp. 238-242, 2017.

[11] S. Covanov, & E. Thomé, "Fast integer multiplication using generalized Fermat primes". *Mathematics of Computation*, vol. 88, no. 317, pp. 449-1477, 2019.

[12] S. A. Nagar and S. Alshamma, “High speed implementation of RSA algorithm with modified keys exchange,” in *2012 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, pp. 639–642, 2012. doi: 10.1109/SETIT.2012.6481987.

[13] Y. Wu and X. Wu, “Implementation of efficient method of RSA key-pair generation algorithm,” in *2017 IEEE International Symposium on Consumer Electronics (ISCE)*, pp. 72–73, 2017. doi: 10.1109/ISCE.2017.8355552.

[14] E. Ochoa-Jiménez, L. Rivera-Zamarripa, N. Cruz-Cortés, and F. Rodríguez-Henríquez, “Implementation of RSA Signatures on GPU and CPU Architectures,” *IEEE Access*, vol. 8, pp. 9928–9941, 2020, doi: 10.1109/ACCESS.2019.2963826.
[15] A.M. Fernandes, A. Pai, & L.M.M. Colaco, "Secure SDLC for IoT based health monitor. In *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), IEEE,* pp. 1236-1241, 2018.
[16] P. Beynon-Davies, C. Carne, H. Mackay, & D. Tudhope, "Rapid application development (RAD): an empirical review". *European Journal of Information Systems*, vol. 8, no. 3, pp. 211-223, 2019.
[17] S. D. Teasley, L. A. Covi, M. S. Krishnan, and J. S. Olson, “Rapid software development through team collocation,” *IEEE Trans. Softw. Eng.*, vol. 28, no. 7, pp. 671–683, Jul. 2012, doi: 10.1109/TSE.2002.1019481.