JITET (Jurnal Informatika dan Teknik Elektro Terapan)
Vol. 12 No. 3S1, pISSN: 2303-0577 eISSN: 2830-7062 http://dx.doi.org/10.23960/jitet.v12i3S1.5218

PENERAPAN HYBRID CRYPTOGRAPHY MENGGUNAKAN CAMELLIA DAN DUAL MODULUS RSA PADA PERTUKARAN FILE

Mitzaqon Gholizhan Ar Romandhon[1*], Achmad Junaidi[2], Andreas Nugroho Sihananto[3]

[1,2,3]Universitas Pembangunan Nasional "Veteran" Jawa Timur; Jl. Rungkut Madya, Gn. Anyar, Kec. Gn. Anyar, Surabaya; (031) 8706369

Received: 20 Agustus 2024
Accepted: 5 Oktober 2024
Published: 12 Oktober 2024

Keywords:
Kriptografi;
Hybrid Cryptography;
Pertukaran *File*;
Camellia;
Dual Modulus RSA;

Corespondent Email:
achmadjunaidi.if@upnjatim.ac.id

Abstrak. Kebutuhan akan keamanan atas data merupakan hal yang sangat penting dalam era digital saat ini, terutama pada proses pertukaran data yang bersifat sensitif terhadap serangan siber. Selain keamanan data, ukuran data yang semakin besar juga menjadi permasalahan dalam proses pertukaran file karena waktu yang dibutuhkan dalam pemrosesan juga semakin lama. Penelitian ini melakukan implementasi skema *hybrid cryptography* menggunakan algoritma *Camellia* dan *Dual Modulus RSA* yang bertujuan untuk mengatasi permasalahan-permasalah tersebut. Pemilihan skema *hybrid cryptography* adalah untuk mendapatkan kekuatan dari masing-masing algoritma, sehingga keamanan dan kecepatan dari tiap algoritma dapat didapatkan. Tahapan yang dilakukan dalam penelitian ini adalah studi literatur, perancangan, implementasi dan pengujian. Hasil dari pengujian yang telah dilakukan, diperoleh bahwa untuk proses pembangkitan kunci skema hybrid memiliki waktu tempuh yang mirip dengan algoritma DM-RSA dengan perbedaan 9.3% lebih cepat dan pada proses enkripsi dan dekripsi memiliki waktu tempuh yang mirip dengan algoritma *Camellia* dengan perbedaan 2.3% lebih cepat. Untuk keseluruhan proses algoritma *hybrid* memiliki waktu tempuh yang mirip dengan algoritma *Camellia* untuk skenario ukuran data 600MB dan 1200MB dengan perbedaan 25.2% lebih lambat.

Abstract. *The need for data security is very important in today's digital era, especially in the process of exchanging sensitive data. In addition to data security, the increasing size of data also becomes a problem in the process of file exchange, because the cryptography process also require longer time. This research implements a hybrid cryptography using Camellia and Dual Modulus RSA algorithms aimed to accomplish those issues. The selection of the hybrid cryptography is to gain the strengths of each algorithm so that the security and speed of each algorithm can be achieved. The stages carried out in this research include literature study, design, implementation, and testing. The results of the tests showed that for the key generation process, the hybrid scheme has a similar time to the DM-RSA algorithm, with a difference of 9.3% faster, and for the encryption and decryption processes, it has a similar time to the Camellia algorithm, with a difference of 2.3% faster. For the overall process, the hybrid algorithm has a similar time to the Camellia algorithm for data sizes of 600MB and 1200MB, with a difference of 25.2% slower.*

1. PENDAHULUAN

Di era teknologi saat ini, pertukaran data menjadi hal yang lumrah, baik dalam

komunikasi pribadi, bisnis, maupun instansi pemerintah. Setiap perangkat yang terhubung ke internet terus-menerus melakukan pertukaran data, baik disadari maupun tidak. Proses ini tidak selalu memerlukan interaksi manusia, karena beberapa aplikasi yang berjalan di latar belakang juga dapat melakukan pertukaran data. Salah satu bentuk umum dari pertukaran data adalah pertukaran *file*, yang mencakup berbagai jenis *file* seperti dokumen, gambar, *video*, *audio*, hingga *file* arsip. [1]. Dalam lingkungan akademik, *file* yang dikirim dapat berupa materi kuliah, hasil ujian, penelitian, dan presentasi. Umunya, pertukaran *file* dapat dilakukan tanpa menggunakan koneksi internet, seperti melalui *Bluetooth*, namun teknologi ini memerlukan waktu yang lebih lama dan memiliki keterbatasan jarak penggunaan [2]. Pertukaran *file* menggunakan internet menjadi solusi atas permasalahan tersebut, karena kecepatan pengiriman dan penerimaan *file* bergantung pada kecepatan internet pengirim dan penerima yang dipengaruhi oleh jarak, perangkat keras, topologi jaringan, serta ukuran paket data yang dikirimkan [3].

Masalah yang kerap terjadi dalam pertukaran informasi atau data adalah kurangnya perhatian pada keamanan atas data atau informasi yang tengah dikirimkan [4]. Kriptografi sering digunakan sebagai solusi untuk masalah ini, dengan memanfaatkan algoritma dan matematika untuk menjaga kerahasiaan data [5]. Data diubah menjadi ciphertext sehingga jika data dicuri, pencuri tidak dapat memahami isi aslinya [6]. Meskipun kriptografi dapat mengatasi masalah keamanan data, ukuran file yang semakin besar menjadi tantangan tersendiri [7]. Hal ini dikarenakan, algoritma kriptografi yang umum untuk digunakan dalam pertukaran *file* adalah algoritma kriptografi asimetris, karena kriptografi simetris hanya menggunakan satu kunci, yang kurang aman untuk transmisi kunci [8]. Namun, algoritma kriptografi asimetris juga memiliki kelemahan, yaitu terletak pada proses enkripsi dan dekripsi yang lambat, terutama untuk *file* berukuran besar. Oleh karena itu, penelitian ini mengusulkan penggunaan *hybrid cryptography*, yang menggabungkan algoritma kriptografi simetris (kunci rahasia) dan asimetris (kunci publik) untuk meningkatkan keamanan dan performa [9]. *Hybrid cryptography* mendapatkan keamanan dalam proses *key-exchange* dari kriptografi asimetris dan kecepatan enkripsi serta dekripsi dari kriptografi simetris [10].

Berdasarkan uraian di atas, dapat disimpulkan bahwa pertukaran *file* adalah aktivitas yang sering dilakukan. Dalam proses pengiriman *file*, *file* yang dikirim sering kali bersifat rahasia dan memiliki variasi dalam ukuran. Keamanan tidak hanya diterapkan pada *file* yang bersifat rahasia, melainkan pada seluruh *file*. Oleh karena itu, penelitian ini mengambil judul "Penerapan Hybrid Cryptography Menggunakan Camellia dan Dual Modulus RSA dalam Pertukaran File." Hasil dari penelitian ini diharapkan dapat mempermudah pertukaran *file* secara aman dan cepat. Selain itu, diharapkan penelitian ini dapat memberikan kontribusi dalam bidang akademik dengan menyediakan hasil uji coba dari algoritma *hybrid cryptography* menggunakan algoritma *Camellia* dan *Dual Modulus* RSA (DM-RSA).

2. TINJAUAN PUSTAKA

Penelitian ini mengacu pada penelitian-penelitian terdahulu. Beberapa topik penelitian terdahulu yang digunakan untuk menjadi acuan dalam penulisan penelitian ini antara lain *hybrid cryptography*, serta algoritma *Camellia* dan DM-RSA. Penelitian terdahulu menjadi acuan dan bahan pendukung atau referensi untuk penelitian ini.

Penelitian [11] menjelaskan bahwa kekuatan algoritma Rivest Shamir Adleman (RSA) terletak pada nilai p dan q yang merupakan bilangan prima. Semakin besar bilangan prima tersebut, semakin sulit keamanan untuk ditembus; sebaliknya, semakin kecil bilangan primanya, semakin rentan keamanannya. Algoritma DM-RSA menggunakan 4 bilangan prima untuk menghasilkan 4 kunci, yaitu 2 kunci publik dan 2 kunci pribadi. Implementasi penelitian ini dilakukan pada aplikasi chat dengan arsitektur *client-server*, dan uji keamanan dilakukan menggunakan metode faktorisasi *Kraitchik*. Hasil pengujian menunjukkan bahwa algoritma DM-RSA memiliki ketahanan hingga dua kali lebih baik dibandingkan dengan algoritma RSA standar.

Pada penelitian [12] menjelaskan bahwa algoritma enkripsi simetris *Camellia* cocok digunakan dalam *embedded cryptography*

system. Penelitian ini melakukan implementasi algoritma *Camellia* ke dalam *hardware* guna memberikan perlindungan untuk data sensitif atau rahasia pada komunikasi menggunakan IOT. Perbedaan implementasi berada pada fase *key schedule* dimana komputasi untuk variabel KA dan KB dilakukan pada fase enkripsi guna mengurangi konsumsi daya pada perangkat. Hasil dari implementasi algoritma *Camellia* memberikan peningkatan dalam area dan *throughput*.

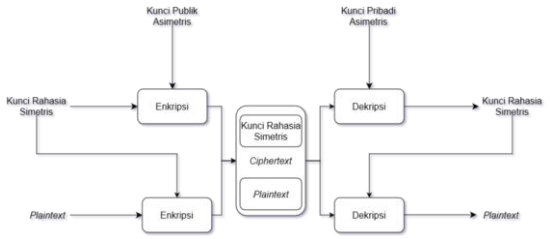
2.1. Kriptografi

Kriptografi merupakan kata yang berasal dari kata Yunani "*kryptos"* yang berarti "*hidden*" atau rahasia dan "*graphein"* yang memiliki arti "*writing*" atau tulisan. Kata kriptografi juga digunakan bersamaan dengan enkripsi dan merujuk pada satu hal yang sama (sinonim) [13]. Ada beberapa pustaka yang mengartikan kriptografi dalam bentuk lain, seperti penelitian yang mengartikan kriptografi sebagai ilmu pengetahuan dan seni dalam mengamankan pesan [14] serta pelatihan dan penyelidikan dalam menyembunyikan data serta sebuah ilmu yang digunakan untuk melindungi data [15].

Saat ini kriptografi telah diadopsi oleh banyak aplikasi, contohnya algoritma RSA yang telah digunakan dalam sistem pembayaran kredit *online* [13] , *password vault* [16], *iris recognition*, RFID, *smart grid*, dan *E-health* [17].

2.2. Hybrid Cryptography

Hybrid cryptosystem atau *hybrid cryptography* merupakan metode kriptografi yang menggabungkan 2 atau lebih metode kriptografi untuk mendapatkan kelebihan dari masing-masing metode kriptografi yang digunakan. Namun, pada umumnya penggunaan *hybrid cryptography* setidaknya menggunakan algoritma kriptografi simetris dan asimetris. Algoritma kriptografi simetris digunakan untuk enkripsi dan dekripsi pada plaintext, sementara algoritma kriptografi asimetris digunakan untuk enkripsi dan dekripsi pada kunci kriptografi simetris. Pendekatan ini dapat meningkatkan performa dari algoritma serta keamanan data [1].



Gambar 1. Skema *hybrid cryptography*

2.3. Camellia

Camellia merupakan salah satu algoritma kriptografi simetris menggunakan metode *block cipher* yang dikembangkan oleh *Nippon Telegraph, Telephone Corporation* (NTT), dan *Mitsubishi Electric Corporation* (Mitsubishi) guna memberikan fleksibilitas dan keamanan pada data. Algoritma *Camellia* telah terstandarisasi di *International Organization for Standardisation* (ISO) dan *the International Electrotechnical Commission* (IEC), serta telah diakui oleh *Cryptography Research and Evaluation Committee* (CRYPTREC) dan *New European Schemes for Signatures, Integrity, and Encryption* (NESSIE) [12].

Camellia memiliki ukuran 128 bit untuk blok serta 128, 192, dan 256 bit untuk pilihan ukuran kunci yang dapat digunakan. Algoritma *camellia* memiliki 2 komponen penting yang merupakan proses yang digunakan dalam algoritma itu sendiri, yaitu *key scheduling* dan *data randomizing* atau *data processing* [12].

2.4. Dual Modulus RSA

Dual Modulus RSA (DM-RSA) adalah pengembangan dari algoritma *Rivest Shamir Adleman* (RSA) untuk meningkatkan keamanan dengan menggunakan dua modulus dalam proses pembangkitan kunci, enkripsi, dan dekripsi, sehingga menghasilkan dua kunci pribadi dan dua kunci publik [11]. Algoritma RSA dikenal memiliki tingkat keamanan yang sulit ditembus jika bilangan primanya sangat besar, karena kekuatan utama algoritma ini terletak pada pemilihan bilangan prima tersebut [18]. Namun, ini juga menjadi kelemahan, di mana jika bilangan primanya kecil, algoritma menjadi lebih rentan. Penelitian [11] menyatakan bahwa kelemahan ini tidak ada pada algoritma DM-RSA, karena penggunaan dua modulus yang meningkatkan keamanan dari algoritma itu sendiri.

Berikut merupakan simbol-simbol yang digunakan pada algoritma ini:

1. p dan q, bilangan prima pembentuk nilai n
2. n=p.q, n merupakan modulus dari 2 bilangan prima
3. (n) merupakan maksimal bilangan untuk nilai e
4. e merupakan bilangan bulat untuk kunci publik, kunci publik akan berbentuk {e, n}
5. d merupakan bilangan acak untuk kunci pribadi, kunci pribadi akan berbentuk {d, n}
6. m merupakan pesan berbentuk *plaintext*
7. c merupakan pesan berbentuk *ciphertext*
8. i mengacu tiap *index* untuk variabel

DM-RSA yang digunakan pada penelitian [11] sendiri berdasarkan pada pengembangan algoritma RSA yang dilakukan oleh Swami Balram. Berikut merupakan proses pembangkitan kunci, proses enkripsi dan juga proses dekripsi:

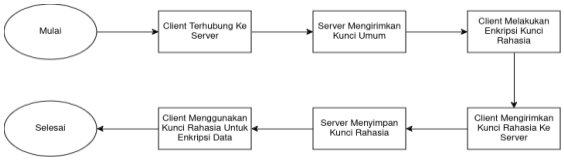
1. Pembangkitan bilangan prima p_1, p_2, q_1, dan q_2, dimana ukuran tiap pasangan p_i dan q_i haruslah sama
2. Hitung masing-masing nilai modulus menggunakan $n_i = p_i \,.\, q_i$
3. Hitung hasil bagi $\varphi(n_i) = (p_i - 1) \,.\, (q_i - 1)$
4. Pilih dua bilangan bulat e_1 dan e_2 untuk pembuatan 2 kunci publik, dimana $gcd(\varphi(n_1), e_1) = 1$ dan $gcd(\varphi(n_2), e_2) = 1$
5. Hitung bilangan acak d_1 dan d_2 untuk pembuatan 2 kunci pribadi dengan syarat $d_1 \,.\, e_1 \bmod \varphi(n_1) = 1$ dan $d_2 \,.\, e_2 \bmod \varphi(n_2) = 1$
6. Proses enkripsi pesan m menggunakan $c = (m^{e_1} \bmod n_1)^{e_2} \bmod n_2$
7. Proses dekripsi pesan c menggunakan $m = (c^{d_2} \bmod n_2)^{d_1} \bmod n_1$

3. METODE PENELITIAN

Metode yang digunakan dalam penelitian ini meliputi, perancangan algoritma *hybrid*, perancangan aplikasi, implementasi dan pengujian.

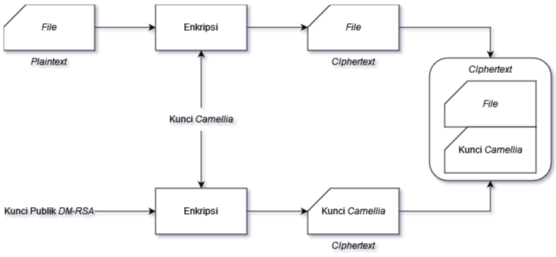
3.1. Perancangan Algoritma

Penelitian ini menggunakan dua jenis komunikasi yang masing-masing menggunakan skema yang berbeda. Hal ini bertujuan untuk menghilangkan redundansi pada data yang tidak dibutuhkan pada saat *client* melakukan komunikasi secara *general* dengan *server*. Pembagian atas skema yang digunakan yaitu, untuk pertukaran *file* dan komunikasi antara *client* dengan *server*.

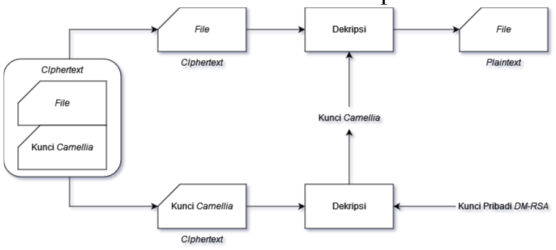


Gambar 2. Skema *Hybrid Cryptosystem Camellia-DM RSA*

Proses enkripsi dan dekripsi pada algoritma *hybrid* memiliki alur yang sama, yaitu dengan menggunakan kunci umum untuk melakukan enkripsi pada *file* beserta *metadata*-nya dan menggunakan kunci publik untuk melakukan enkripsi pada kunci umum tersebut. Setelah kunci umum dan *file* beserta *metadata* telah terenkripsi, maka kedua data tersebut akan disatukan dan ditransmisikan melalui jaringan. Untuk diagram mengenai alur proses enkripsi dan dekripsi pada *packet* atau data yang dikirimkan dapat dilihat pada Gambar 3 dan 4 di bawah ini.



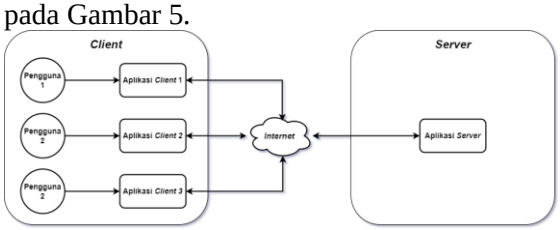
Gambar 3. Alur Enkripsi



Gambar 4. Alur Dekripsi

3.2. Perancangan Aplikasi

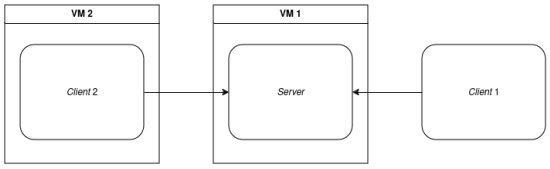
Aplikasi pertukaran *file* diimplementasikan dengan menggunakan arsitektur *server-client*, yaitu dimana tiap *client* tidak melakukan komunikasi secara langsung dengan *client* lain, melainkan *client* melakukan komunikasi atau mengirimkan data ke *server* dan kemudian *server* akan meneruskan data tersebut kepada *client* penerima. *Server* disini berperan sebagai penyalur dari komunikasi antar *client*. Arsitektur aplikasi dapat dilihat pada Gambar 5.



Gambar 5. Arsitektur Aplikasi

3.3. Perancangan Pengujian

Pada tahap ini dilakukan perancangan mengenai pengujian yang akan dilakukan pada aplikasi yang telah dibuat untuk mendapatkan informasi mengenai kinerja aplikasi, memastikan aplikasi telah berjalan dengan baik dan sesuai yang diharapkan, serta mendapatkan informasi mengenai performa dari algoritma yang telah dibuat. Pengujian terbagi atas dua bagian, yaitu pengujian pada aplikasi dengan melakukan inspeksi pada *packet* yang ditransmisikan dan pengujian *benchmark* untuk mendapatkan kecepatan atau waktu tempuh pada proses pembangkitan kunci, enkripsi, dekripsi, dan keseluruhan proses dengan skenario ukuran data 20*MB*, 600*MB*, dan 1200*MB*. Untuk skema pengujian yang akan dilakukan pada aplikasi dapat dilihat pada Gambar 6.



Gambar 6. Skema Pengujian Aplikasi

4. HASIL DAN PEMBAHASAN

Pada tahap ini ditampilkan hasil dari pengujian yang dilakukan pada aplikasi dan berfokus pada alur proses serta bentuk data atau paket dalam pengiriman. Selain itu, terdapat pengujian kecepatan untuk algoritma yang telah diimplementasikan dengan algoritma lain sebagai pembanding. Beberapa algoritma yang digunakan untuk perbandingan adalah algoritma *Dual Modulus RSA*, *RSA* (*Rivest Shamir Adleman*), *Camellia*, *AES* (*Advanced Encryption Standard*), dan algoritma *hybrid* yang diterapkan pada aplikasi pertukaran *file* pada penelitian ini.

4.1. Bentuk Paket

Bentuk paket untuk proses pengiriman *file* dapat dilihat pada Gambar 7. Dimana terdapat sebuah paket yang terdiri atas *header* dan *body*. Bagian *header* tidak akan dienkripsi karena memiliki nilai-nilai yang dibutuhkan dalam proses penerimaan paket, seperti panjang bagian *body*, jenis paket, tipe enkripsi, dan tujuan paket tersebut. Sedangkan *server* sendiri tidak dapat melihat isi paket yang tidak ditujukan untuk *server*.

Header	*Body*
0xe7 0x52 0x34 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x07 0x01 0x00 0x00 0x00	0x00 0x05 0x20 0x4f 0x63 0xac 0x27 0x02 0x00 0x00 0x00 0x63 0x9b 0xea 0x9c 0x00 …

Gambar 7. Bentuk Paket Pengiriman *File*

Setelah *server* menerima keseluruhan data yang merupakan kunci dan *file* beserta *metadata*-nya, *server* akan mengirimkan respon paket kepada *client* pengirim untuk menandakan bahwa paket telah sampai dengan aman dan dilanjutkan dengan meneruskan paket tersebut kepada *client* penerima. Sebelum meneruskan paket, *server* akan mengubah nilai penerima pada *header* yang semula merupakan id penerima menjadi id pengirim, sehingga *client* penerima dapat mengetahui pengirim *file* tersebut. Bentuk paket untuk respon yang dikirimkan oleh *server* ke *client* pengirim baik dalam keadaan terenkripsi dan terdekripsi dapat dilihat pada Gambar 8 dan Gambar 9.

Header	*Body*
0x10 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x0d 0x00 0x02 0x0a 0x00 0x00 0x00 0x00	0xa6 0x57 0x08 0xbf 0xd4 0x54 0xc0 0x3e 0xe8 0x39 0xa7 0x19 0x03 0x0d 0xfb 0xc1

Gambar 8. Bentuk *Response* Paket Terenkripsi

Header	*Body*
0x10 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x0d 0x00 0x02 0x0a 0x00 0x00 0x00 0x00	0x07 0x01 0x00

Gambar 9. Bentuk *Response* Paket Terdekripsi

4.2. Kecepatan

Pengujian kecepatan dilakukan pada proses pembangkitan kunci, enkripsi, dekripsi, dan proses secara keseluruhan yang dilakukan pada 5 algoritma kriptografi dan dengan ukuran kunci yang sama, yaitu 128 *bit*.

Tabel 1. Waktu tempuh pembangkitan kunci

Algoritma	Waktu (ms)
DM-RSA	4166.7
RSA	1892.4
Camellia	0.003288
AES	0.003312
Hybrid	3775.3

Dapat dilihat pada Tabel 1 bahwa algoritma *Camellia* dan *AES* memiliki waktu tempuh yang mirip, sedangkan untuk algoritma *DM-RSA* sendiri memiliki waktu tempuh 220% yang lebih tinggi dibandingkan pendahulunya, algoritma *RSA*. Begitupula untuk algoritma *hybrid* yang memiliki waktu yang sama dengan algoritma *DM-RSA* dan lebih cepat sekitar 9.3%.

Tabel 2. Waktu tempuh proses enkripsi

Algoritma	Waktu (ms)		
	20 MB	*600MB*	*1200MB*
DM-RSA	6391	190,893	391,56
RSA	819	24,988	45,665
Camellia	201	5965	11,970
AES	551	16,624	33,373
Hybrid	197	5392	11,888

Pada Tabel 2 algoritma *DM-RSA* memiliki waktu tempuh tertinggi dibandingkan dengan algoritma lainnya, yaitu 782.9% lebih tinggi dibandingkan dengan algoritma *RSA* untuk rata-rata waktu dari seluruh skenario ukuran data. Sedangkan algoritma *Camellia* memiliki waktu tempuh terpendek dari seluruh algoritma yang dibandingkan. Alasan dari tingginya waktu enkripsi dari algoritma *DM-RSA* adalah diperlukannya pemrosesan 2 kali lebih banyak dari algoritma *RSA*. Algoritma *hybrid* sendiri memiliki waktu tempuh yang serupa dengan algoritma *Camellia* untuk seluruh skenario ukuran data dengan selisih waktu tempuh sebanyak 219.6 *ms* untuk rata-rata seluruh ukuran skenario data.

Tabel 3. Waktu tempuh proses dekripsi

Algoritma	Waktu (ms)		
	20 MB	*600MB*	*1200MB*
DM-RSA	3268	97,255	199,326
RSA	956	29,048	58,002
Camellia	197	6199	12,604
AES	439	13,221	26,492
Hybrid	188	6186	12,283

Pada Tabel 3 bahwa algoritma *DM-RSA* memiliki waktu tempuh tertinggi dibandingkan dengan algoritma lainnya, yaitu 340.1% lebih tinggi dibandingkan dengan algoritma *RSA* untuk rata-rata waktu dari seluruh skenario ukuran data. Sedangkan algoritma *Camellia* memiliki waktu tempuh terpendek dari seluruh algoritma yang dibandingkan. Alasan dari tingginya waktu enkripsi dari algoritma *DM-RSA* adalah sama dengan proses enkripsi, yaitu diperlukannya pemrosesan 2 kali lebih banyak dari algoritma *RSA*. Hal ini dikarenakan, proses dekripsi untuk algoritma *RSA* dan *DM-RSA* adalah proses yang sama dengan proses enkripsi. Perbedaannya terletak pada penggunaan variabel pada proses tersebut, seperti proses enkripsi yang menggunakan variabel *e*, sedangkan pada proses dekripsi menggunakan variabel *d*. Algoritma *hybrid* sendiri memiliki waktu tempuh yang serupa dengan algoritma *Camellia* untuk seluruh skenario ukuran data dengan selisih waktu tempuh sebanyak 144.3 *ms* untuk rata-rata seluruh ukuran skenario data.

Tabel 4. Waktu tempuh keseluruhan proses

Algoritma	Waktu (ms)		
	20 MB	*600MB*	*1200MB*
DM-RSA	16,092	344,530	727,870
RSA	3440	50,839	111,331
Camellia	385	12,008	23,944
AES	975	19,472	58,633
Hybrid	4274	15,953	28,140

Sama seperti pada pengujian sebelumnya, algoritma *DM-RSA* memiliki waktu tempuh

tertinggi dibandingkan dengan algoritma lainnya, yaitu 599.7% lebih tinggi dibandingkan dengan algoritma *RSA* untuk rata-rata waktu dari seluruh skenario ukuran data. Sedangkan algoritma *Camellia* tetap memiliki waktu tempuh terpendek dari seluruh algoritma yang dibandingkan.

Jika diperhatikan dan dianalisis lebih lanjut, algoritma *hybrid* pada Tabel 2 memiliki waktu tempuh tertinggi kedua setelah algoritma *DM-RSA* pada skenario ukuran data 20MB. Sedangkan untuk skenario ukuran data 600MB dan 1200MB algoritma *hybrid* memiliki waktu tempuh terpendek setelah algoritma *Camellia*. Waktu tempuh untuk algoritma *hybrid* memiliki nilai yang tinggi untuk skenario ukuran data 20MB dikarenakan proses pembangkitan kunci algoritma *DM-RSA* yang digunakan pada algoritma *hybrid*. Hal ini dapat dilihat pada Tabel 1, dimana pembangkitan kunci pada algoritma *DM-RSA* membutuhkan waktu tempuh tertinggi. Sedangkan untuk skenario ukuran data 600MB dan 1200MB, waktu tempuh untuk pembangkitan kunci menjadi cukup bias, dikarenakan proses enkripsi dan dekripsi menjadi lebih dominan dengan digunakannya ukuran data yang berukuran besar.

5. KESIMPULAN

a. Implementasi dari algoritma *hybrid cryptography Camellia* dan *Dual Modulus RSA* dilakukan dengan menggunakan dua skema yang berbeda, yaitu dalam komunikasi antara *client* dengan *server* dan pertukaran *file*. Hal ini bertujuan untuk menghilangkan redundansi data yang tidak diperlukan dalam tiap proses.
b. Berdasarkan hasil pengujian bentuk paket yang ditransmisikan. Dapat dilihat bahwa paket dalam keadaan terenkripsi yang tidak memungkinkan selain pemilik kunci pribadi untuk mengetahui isi paket. Hal ini mengindikasikan bahwa implementasi algoritma *hybrid cryptography* menggunakan algoritma *Camellia* dan *Dual Modulus RSA* telah berhasil dilakukan.
c. Pada pengujian kecepatan (*benchmark*) dapat dilihat bahwa algoritma *hybrid* selalu unggul untuk tiap proses dengan waktu tempuh yang mirip dengan algoritma *Camellia* selain proses pembangkitan kunci. Hal ini dikarenakan proses pembangkitan kunci pada algoritma *hybrid* memerlukan algoritma *Dual Modulus RSA* yang merupakan algoritma dengan waktu tempuh tertinggi.
d. Algoritma *hybrid cryptography* yang telah diimplementasikan memberikan solusi yang seimbang antara kecepatan dan juga keamanan. Hal ini dibuktikan dengan waktu tempuh algoritma *hybrid* yang menyamai algoritma *Camellia* untuk tiap proses. Pada keseluruhan proses algoritma *hybrid* memiliki waktu tempuh yang cukup tinggi untuk skenario ukuran data 20 MB yang dikarenakan proses pembangkitan kunci yang dilakukan setiap perulangan. Hal ini dapat diatasi dengan melakukan pembangkitan kunci sekali yang dipastikan dapat mempercepat algoritma *hybrid* pada keseluruhan proses.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Allah *Subhanallahu wata'ala* yang telah melimpahkan rahmat, hidayah, serta karunia-Nya sehingga penelitian ini dapat diselesaikan dengan baik. Selain itu penulis juga mengucapkan terimakasih kepada orang tua dan teman-teman penulis yang selalu memberikan dukungan dalam pengerjaan penelitian ini.

DAFTAR PUSTAKA

[1] A. Guru and A. Ambhaikar, "Study of Data Security Using Hybrid Cryptographic Techniques," Solid State Technology, vol. 63, no. 6, 2020. https://www.researchgate.net/publication/356267558

[2] A. A. Mahapadi and S. Das, "F-Share for Hand Held Devices Using Wi-Fi Bluetooth Network," *International Research Journal of Engineering*

and Technology, vol. 2, no. 3, pp. 1312-1315, 2015. www.irjet.net

[3] F. Muttaqin, A. Junaidi, and B. S. Pw, "MODEL DAN SIMULASI JARINGAN MULTI PROTOKOL LABEL SWICTHING (MPLS) PADA VOIP DI UPN 'VETERAN' JAWA TIMUR," vol. 11, no. 3, pp. 75–84, Jan. 2016

[4] S. N. Nugraha, "PENERAPAN ALGORITMA KRIPTOGRAFI ELGAMAL PADA APLIKASI PENGAMANAN PESAN BERBASIS WEBSITE," Jurnal Informatika dan Teknik Elektro Terapan, vol. 12, no. 3, Aug. 2024, doi: https://doi.org/10.23960/jitet.v12i3.4794.

[5] A. M. Qadir and N. Varol, "A Review Paper on Cryptography," *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*, Jun. 2019, doi: https://doi.org/10.1109/isdfs.2019.8757514.

[6] J. Jamaludin, "Rancang Bangun Kombinasi Hill Cipher dan RSA Menggunakan Metode Hybrid Cryptosystem", *SinkrOn*, vol. 2, no. 2, pp. 86-93, Mar. 2018.

[7] D. Anggraini, "Penerapan Algoritma Fibonacci Code Pada File Transfer Berbasis Android," *Journal of Informatics Management and Information Technology*, vol. 1, no. 3, pp. 91-94, 2021. doi: 10.47065/jimat.v1i3.109.

[8] H. Najm, H. K. Hoomod, and R. Hassan, "A proposed hybrid cryptography algorithm based on GOST and salsa (20)," *Periodicals of Engineering and Natural Sciences*, vol. 8, no. 3, pp. 1829–1835, Sep. 2020, doi: https://doi.org/10.21533/pen.v8i3.1619.

[9] D. Rachmawati, M. S. Lydia, and W. A. Siregar, "Hybrid Cryptosystem Implementation Using IDEA and Knapsack Algorithm for Message Security," *Journal of Physics: Conference Series*, vol. 1090, p. 012030, Sep. 2018, doi: https://doi.org/10.1088/1742-6596/1090/1/012030.

[10] N. Francis and T. Monoth, "An Analysis of Hybrid Cryptographic Approaches for Information Security," International Journal of Applied Engineering Research, vol. 13, no. 3, pp. 124-127, 2018. http://www.ripublication.com

[11] Aminudin and I. Nuryasin, "Analisis dan Implementasi Algoritma Asimetris Dual Modulus RSA (DM-RSA) pada Aplikasi Chat," Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), vol. 5, no. 4, pp. 768–773, Aug. 2021, doi: https://doi.org/10.29207/resti.v5i4.3297.

[12] B. Rashidi, "Flexible and high-throughput structures of Camellia block cipher for security of the Internet of Things," IET Computers & Digital Techniques, vol. 15, no. 3, pp. 171–184, Mar. 2021, doi: https://doi.org/10.1049/cdt2.12025.

[13] N. J. Bouman, "Cryptography from quantum uncertainty: In the presence of quantum side information," Dissertation, Universiteit Leiden, 2012. http://hdl.handle.net/1887/20302

[14] Y. S. Santoso, "Message Security Using a Combination of Hill Cipher and RSA Algorithms," Jurnal Matematika Dan Ilmu Pengetahuan Alam LLDikti Wilayah 1 (JUMPA), vol. 1, no. 1, pp. 20–28, Mar. 2021, doi: https://doi.org/10.54076/jumpa.v1i1.38.

[15] H. Damghani, H. Hosseinian, and L. Damghani, "Cryptography review in IoT," in 2019 4th Conference on Technology in Electrical and Computer Engineering (ETECH 2019), Oct. 2019.

[16] A. Chatzikonstantinou, C. Ntantogian, G. Karopoulos, and C. Xenakis, "Evaluation of Cryptography Usage in Android Applications," Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), 2016, doi: https://doi.org/10.4108/eai.3-12-2015.2262471.

[17] R. Harkanson and Y. Kim, "Applications of elliptic curve cryptography," Proceedings of the 12th Annual Conference on Cyber and Information Security Research, Apr. 2017, doi: https://doi.org/10.1145/3064814.3064818.

[18] A. Y. Suseno, N. Sulistiyowati, and Purwantoro, "Analisis Peningkatan hybrid Cryptosystem Untuk Enkripsi dan Dekripsi Menggunakan Vigenere Cipher dan RSA Pada Text," STRING (Satuan Tulisan Riset dan Inovasi Teknologi), vol. 6, no. 2, p. 142, Dec. 2021, doi: https://doi.org/10.30998/string.v6i2.10309.