 ISSN:2302-8149
e-ISSN:2540-9719

Pemanfaatan AES dengan *Key* Dinamis sebagai Metode Pengamanan Data pada *Smart Card*

Noprianto*, Vivi Nur Wijayaningrum, Rudy Ariyanto
Jurusan Teknologi Informasi, Politeknik Negeri Malang
Jalan Soekarno Hatta 9, Malang
*e-mail: noprianto@polinema.ac.id

(***received*:** 18 Mei, ***revised*:** 24 Juli 2021, ***accepted*:** 3 Agustus 2021)

Abstrak

Terjadinya pandemi Covid-19 di hampir seluruh belahan dunia, termasuk Indonesia, menjadikan masyarakat mempunyai gaya hidup baru dalam bertransaksi untuk mencegah penularan virus sesuai anjuran Pemerintah. Semua transaksi cenderung dilakukan secara non tunai dengan memanfaatkan teknologi *smart card* untuk menghindari adanya kontak fisik dengan orang lain. Tingginya penggunaan *smart card* pada berbagai bidang untuk mendukung aktivitas sehari-hari menyebabkan rawannya terjadi pencurian data di dalam *smart card* oleh pihak yang tidak bertanggung jawab. AES sebagai algoritma kriptografi dapat digunakan untuk mengamankan data di dalam *smart card* dengan melakukan enkripsi data sebelum data yang bersifat rahasia tersebut disimpan ke dalam *smart card*. Untuk meningkatkan keamanan data, diusulkan sebuah mekanisme penggunaan *key* yang bersifat dinamis dengan memanfaatkan *Unique Identifier* (UID) setiap *smart card*. Dengan demikian, *key* yang digunakan untuk melakukan enkripsi dan dekripsi data dibentuk berdasarkan UID dan berbeda-beda untuk setiap *smart card*. Hasil pengujian menunjukkan bahwa penggunaan AES dengan *key* yang bersifat dinamis ini mampu mengamankan data 40 byte plainteks menjadi 48 byte cipherteks, dengan rata-rata waktu komputasi sebesar 71.2 ms untuk penulisan data dan 89.4 ms untuk pembacaan data menggunakan *key* 128 bit, 70.8 ms untuk penulisan data dan 88 ms untuk pembacaan data menggunakan *key* 192 bit, dan 72 ms untuk penulisan data dan 88.4 ms untuk pembacaan data menggunakan *key* 256 bit. Waktu komputasi ini hanya mempunyai selisih sekitar 2 ms dibandingkan dengan penulisan dan pembacaan data tanpa mekanisme enkripsi dan dekripsi.

Kata kunci: covid-19, dekripsi, enkripsi, keamanan, kriptografi

Abstract

The occurrence of the Covid-19 pandemic in almost all parts of the world, including Indonesia, has made people have a new lifestyle in transactions to prevent transmission of the virus as recommended by the Government. All transactions tend to be carried out in non-cash using smart card technology to avoid physical contact with other people. The high use of smart cards in various fields to support daily activities makes it prone to data theft on the smart card by irresponsible parties. AES as a cryptographic algorithm can be used to secure data on the smart card by encrypting the data before the confidential data is stored on the smart card. To improve data security, a dynamic key usage mechanism is proposed by utilizing the Unique Identifier (UID) of each smart card. Thus, the key used to encrypt and decrypt data is formed based on the UID and is different for each smart card. The test results show that the use of AES with a dynamic key is able to secure 40 bytes of plaintext to 48 bytes of ciphertext, with an average computation time of 71.2 ms for writing data and 89.4 ms for reading data using 128-bit keys, 70.8 ms for writing data and 88 ms for reading data using a 192-bit key, and 72 ms for writing data and 88.4 ms for reading data using a 256-bit key. This computation time only has a difference of about 2 ms compared to writing and reading data without encryption and decryption mechanisms.

Keywords: *covid-19, decryption, encryption, security, cryptography*

1 Pendahuluan

Teknologi *smart card* mengalami perkembangan yang sangat pesat. Hal ini dikarenakan *smart card* menawarkan fitur keamanan seperti kontrol akses, serta kemampuan untuk melakukan berbagai

fungsi lainnya sehingga menjadikan *smart card* dapat diterapkan untuk berbagai aplikasi di kehidupan sehari-hari [1]. Salah satu contoh penggunaan *smart card* adalah KTP elektronik (e-KTP) yang digunakan untuk menyimpan data pribadi setiap penduduk Indonesia [2]. Di dalam e-KTP terdapat chip berbasis mikroprosesor, sehingga dengan adanya teknologi *smart card* ini, e-KTP tidak hanya dapat dimanfaatkan sebagai autentikasi identitas saja, tetapi juga dapat mengakomodasi fungsi-fungsi lainnya [3].

Penggunaan *smart card* yang juga sering ditemui di Indonesia adalah *electronic money* (*e-money*) sebagai alat pembayaran pengganti uang tunai [4][5]. Keuntungan penggunaan *e-money* adalah mengurangi risiko kehilangan uang dan memberikan kemudahan transaksi bagi pengguna karena nominal yang dibayarkan menggunakan *e-money* pasti sesuai dengan tagihan pembayaran, tanpa perlu ada uang kembalian. Selain itu, adanya pandemi Covid-19 saat ini menyebabkan perilaku masyarakat Indonesia juga berubah ketika melakukan transaksi jual beli. Masyarakat cenderung memilih menggunakan *e-money* dan *e-wallet* untuk melakukan transaksi pembayaran guna menghindari adanya kontak fisik dengan orang lain saat melakukan transaksi pembayaran [6].

Saat ini, penggunaan *smart card* juga banyak dimanfaatkan untuk sistem parkir, baik untuk memudahkan pencatatan kendaraan yang keluar atau masuk lahan parkir [7], maupun untuk proses pembayaran parkir oleh pelanggan [8]. Penggunaan *smart card* untuk sistem parkir ini dapat mengatasi berbagai permasalahan yang terjadi pada sistem parkir konvensional, antara lain lamanya durasi pelayanan pelanggan, tingginya penggunaan karcis parkir, dan tidak adanya fitur untuk memulihkan data transaksi [9]. Selain itu, tindakan pencurian kendaraan bermotor yang terparkir juga dapat diminimalkan dengan melakukan monitoring kendaraan yang keluar dan masuk gerbang parkir.

Dengan adanya data pribadi yang bersifat rahasia tersimpan di dalam e-KTP maupun *e-money*, maka perlu adanya mekanisme pengamanan data yang dilakukan pada *smart card* untuk menjaga kerahasiaan data yang tersimpan di dalamnya sehingga data tersebut tidak dapat dibaca oleh pihak yang tidak bertanggung jawab [10]. Untuk mengamankan data yang tersimpan pada *smart card*, berbagai teknik dapat dilakukan, salah satunya adalah dengan menggunakan kriptografi. Terdapat beberapa algoritma kriptografi yang sering digunakan, antara lain Data Encryption Standard (DES), Triple Data Encryption Standard (3DES), dan Advanced Encryption Standard (AES). Perbedaan utama antara ketiganya adalah panjang *key* yang digunakan untuk proses enkripsi dan dekripsi. AES menggunakan tiga jenis panjang *key* yaitu 128 bit, 192 bit, dan 256 bit, sementara DES menggunakan 56 bit, dan 3DES menggunakan 168 bit. Secara teori, semakin panjang *key* yang digunakan, maka semakin tinggi tingkat keamanan yang diberikan untuk proses enkripsi data [11]. Algoritma kriptografi lainnya yang juga sering digunakan adalah Rivest Shamir Aldeman (RSA). Pada umumnya, *key* yang digunakan RSA mempunyai panjang 1024 bit dan 2048 bit. Dengan semakin panjangnya *key* yang digunakan pada RSA, tentu semakin tinggi tingkat keamanan data, namun waktu yang diperlukan untuk melakukan proses enkripsi dan dekripsi juga menjadi semakin tinggi. RSA terbukti membutuhkan waktu komputasi lebih lebih tinggi dibandingkan AES dan DES [12][13].

Pada penelitian sebelumnya, pemanfaatan AES dengan mode Cipher Block Chaining (CBC) terbukti dapat digunakan untuk mengamankan data saat proses pengiriman dari perangkat Internet of Things (IoT) ke aplikasi [14]. Sementara itu, pada penelitian ini, AES digunakan untuk mengamankan data pada *smart card*. Namun, karena AES merupakan algoritma yang menggunakan *symmetric key*, maka untuk meningkatkan proses pengamanan data, diusulkan penggunaan *key* bersifat dinamis berdasarkan UID setiap kartu. Dengan demikian, meskipun *key* yang digunakan pada AES bersifat *symmetric*, *key* yang digunakan pada setiap kartu akan berbeda-beda menyesuaikan dengan UID-nya. Penggunaan teknik kriptografi untuk proses enkripsi data pada *smart card* diharapkan dapat meningkatkan keamanan data yang tersimpan di dalam *smart card*. Selain itu, pemilihan algoritma AES untuk proses enkripsi data juga diharapkan tidak memberikan pengaruh yang signifikan terhadap waktu komputasi untuk proses pembacaan dan penulisan data pada *smart card*.

2 Tinjauan Literatur

Penggunaan algoritma DES yang telah dimodifikasi untuk mengamankan data pada *smart card* telah dilakukan oleh Sison, dkk [15]. Proses enkripsi diawali dengan mengubah *key* hexadesimal sepanjang 64 bit menjadi bilangan biner, kemudian hasilnya dikurangi sehingga menjadi 56 bit dan dikirimkan ke substitusi Ganjil-Genap. Pada proses subtitusi Ganjil-Genap dilakukan penggantian

nilai 1 untuk setiap posisi genap dan 0 untuk setiap posisi ganjil di blok 56 bit. Selanjutnya, blok 56 bit tersebut dibagi menjadi dua bagian dengan masing-masing berisi 28 bit bilangan. Pada setiap bagian dilakukan *left shift* dengan menggeser bilangan ke kiri sebanyak satu kali. Kedua bagian tersebut digabungkan kembali dan direduksi menjadi 48 bit, lalu diubah kembali menjadi hexadesimal. Bilangan 48 bit inilah yang digunakan ini menjadi *key* pertama. Selanjutnya, pembuatan *key* dilanjutkan hingga 16 putaran. Adanya substitusi Ganjil-Genap pada penelitian ini memastikan bahwa meskipun data ditahan oleh jaringan lain maupun dialihkan ke tujuan lain, integritas dan kerahasiaan data tetap dapat terjamin [15]. Namun, DES terbukti sangat tidak aman dan tidak dapat diandalkan, sehingga kemudian muncul algoritma lainnya yaitu 3DES atau Triple DES untuk mengatasi permasalahan pada DES [16].

Pada penelitian yang dilakukan oleh Darwito, Yuliana, dan Soelistijorini, 3DES digunakan untuk mengamankan data riwayat kesehatan yang tersimpan di cloud. Hal ini bertujuan untuk memungkinkan penyimpanan dan pertukaran data riwayat pasien sehingga dapat diakses oleh berbagai penyedia layanan kesehatan seperti puskemas, rumah sakit, maupun perusahaan asuransi. Selain itu, *smart card* juga digunakan untuk mengontrol akses informasi pasien. Sebelum menerapkan 3DES untuk enkripsi data, tiga buah *key* (K1, K2, dan K3) perlu dibangkitkan terlebih dahulu. Selanjutnya, enkripsi plainteks dapat dilakukan dengan menggunakan DES tunggal menggunakan K1. Setelah itu, dekripsi dilakukan menggunakan K2 terhadap hasil enkripsi pada langkah sebelumnya. Dengan menggunakan K3, dilakukan enkripsi kembali terhadap hasil dekripsi dari langkah sebelumnya, sehingga diperoleh cipherteks. Hasil pengujian *avalanche effect* yang dilakukan menunjukkan bahwa sistem tersebut memenuhi syarat keamanan dengan nilai 51.4% [17]. 3DES terbukti mampu mengungguli DES, namun 3DES ini juga mempunyai keterbatasan pada ukuran *key* yang digunakan, yaitu 56 bit sesuai dengan standar DES. Selain itu, *key* enkripsi harus diganti setiap 32GB transfer data untuk menghindari terjadinya kebocoran data [16]. Beberapa penelitian menyebutkan bahwa AES bekerja lebih baik dibandingkan DES dan 3DES [16][18]. AES memberikan tingkat keamanan yang tinggi karena menggunakan *key* dengan ukuran lebih panjang dibandingkan dua algoritma lainnya [19].

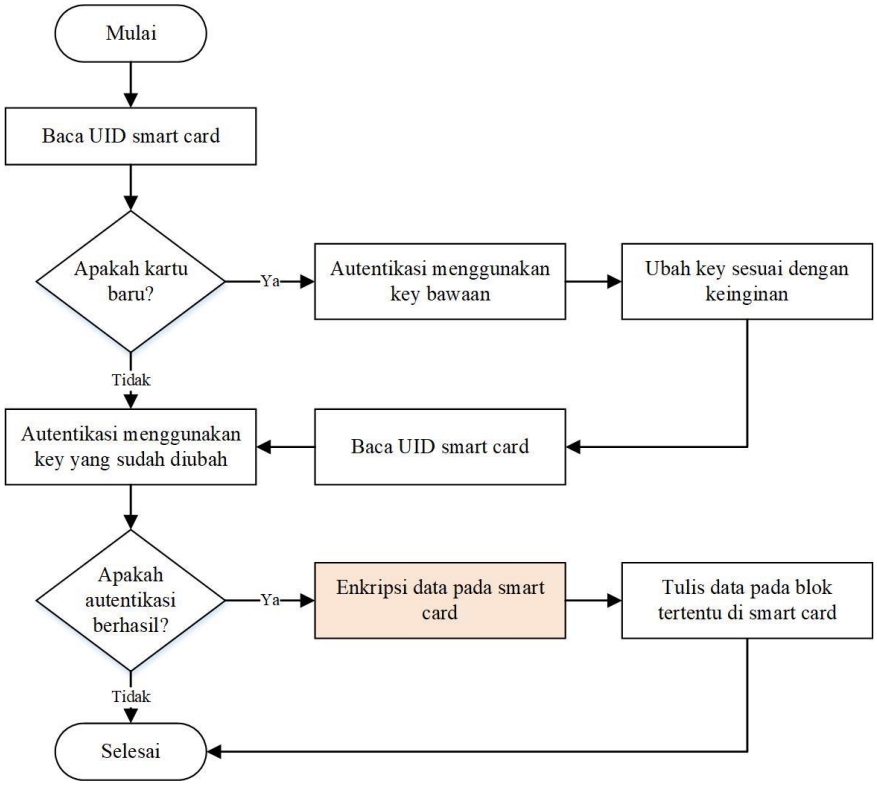
Penelitian ini menggunakan algoritma AES dengan *key* dinamis untuk mengamankan data pada *smart card*. Kebaruan yang ditawarkan pada penelitian ini adalah penerapan *key* yang bersifat dinamis saat melakukan proses enkripsi dan dekripsi data pada *smart card*. Pada umumnya, penelitian-penelitian sebelumnya menggunakan sebuah *key* yang bersifat statis, artinya *key* yang digunakan untuk proses enkripsi dan dekripsi bersifat tetap sesuai dengan nilai yang telah ditentukan. Pada penelitian ini, *key* dinamis yang digunakan untuk melakukan proses enkripsi dan dekripsi data dibentuk berdasarkan kode alfanumerik (UID) yang dimiliki oleh masing-masing *smart card*. Setiap kartu mempunyai nilai unik berukuran empat atau tujuh byte yang bersifat sebagai identitas dari setiap kartu. Dengan demikian, *key* yang digunakan oleh setiap kartu untuk proses enkripsi dan dekripsi nantinya akan menjadi berbeda-beda sesuai dengan UID yang dimiliki oleh setiap kartu tersebut.

3 Metode Penelitian

Penerapan teknologi *smart card* pada penelitian ini membutuhkan beberapa perangkat untuk proses pengembangan dan analisis hasil. Perangkat keras yang dibutuhkan antara lain CPU 2.5 GHz Intel Core i5, memori DDR3 16 GB, Intel HD Graphics 4000 1536 MB, sistem operasi MacOS Mojave, media penyimpanan SSD 512 GB, *smart card* Mifare 1K, dan reader NFC ACR122U SAM. Perangkat lunak yang digunakan adalah Java 8, Intellij IDEA Ultimate 2021.1, dan Gitlab.

Pada penelitian ini, tahapan pembacaan maupun penulisan data pada *smart card* hampir sama seperti cara mengakses *smart card* pada umumnya. Namun, terdapat penambahan proses enkripsi dan dekripsi data dengan menggunakan *key* dinamis untuk meningkatkan level keamanan data. Gambar 1 menunjukkan tahapan yang digunakan untuk melakukan penulisan data ke dalam *smart card*.

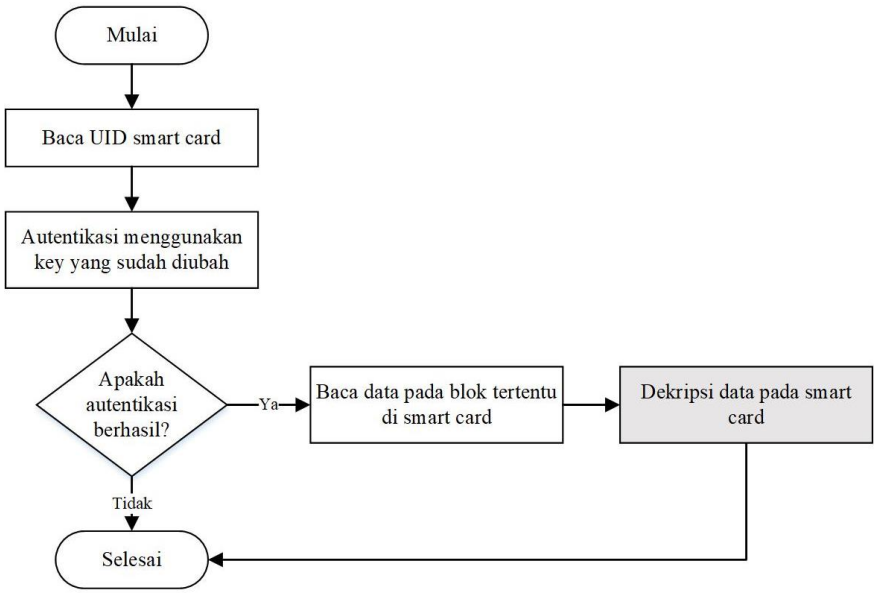
 ISSN:2302-8149
e-ISSN:2540-9719



Gambar 1. Mekanisme Penulisan Data Pada *Smart Card*

Pada Gambar 1, ditunjukkan bahwa penulisan data pada *smart card* dilakukan dengan mendeteksi UID tujuan sehingga dapat diketahui jenis tag yang digunakan. Apabila tag kartu sudah diketahui, selanjutnya perlu dibedakan apakah kartu tersebut merupakan kartu baru atau tidak. Jika kartu yang dideteksi tersebut adalah kartu baru, artinya belum pernah dilakukan pengubahan *key* pada kartu tersebut, maka pengguna harus mengubah *key* terlebih dahulu sebelum menggunakan kartu tersebut. Kemudian, autentikasi dan enkripsi data dapat dilakukan untuk melakukan pengamanan data sebelum data tersebut dituliskan ke dalam *smart card*.

Ketika data yang tersimpan tersebut diperlukan untuk didistribusikan ke aplikasi-aplikasi lain yang membutuhkan, maka proses pembacaan data perlu dilakukan dengan menggunakan reader. Gambar 2 menunjukkan tahapan yang digunakan untuk melakukan pembacaan data pada *smart card*.



Gambar 2. Mekanisme Pembacaan Data Pada *Smart Card*

 ISSN:2302-8149
e-ISSN:2540-9719

Pada Gambar 2, ditunjukkan bahwa pembacaan data pada *smart card* diawali dengan mendeteksi tag kartu dengan cara mendapatkan UID kartu terlebih dahulu. Setelah itu, autentikasi dapat dilakukan menggunakan *key* yang digunakan untuk proses pembacaan data. Apabila proses autentikasi berhasil, maka pembacaan data dapat dilakukan pada sector dan blok tertentu di dalam *smart card*. Karena data yang terbaca tersebut masih berbentuk cipherteks, selanjutnya perlu diubah ke dalam bentuk plainteks terlebih dahulu dengan proses dekripsi data.

4 Hasil dan Pembahasan

Data yang digunakan pada penelitian ini terdiri dari beberapa field untuk kebutuhan sistem parkir, yaitu Tanda Nomor Kendaraan Bermotor (TNKB), tanggal transaksi, status masuk, kode gate, Nomor Induk Pegawai (NIP), kedaluwarsa kartu, dan status kartu [20]. Contoh data yang diisikan pada setiap field ditunjukkan pada Tabel 1.

Tabel 1. Data pada *Smart Card*

Field	Data
TNKB	AB2039YQ
Tanggal transaksi	2021-05-06 16:12:00
Status masuk (keluar/masuk)	1
Kode gate	255
NIP	198911082019031020
Kedaluwarsa kartu	2021-05-06 16:12:00
Status kartu (aktif/masuk)	1

Untuk dapat disimpan ke dalam *smart card*, data tersebut harus diubah ke dalam format hexadesimal terlebih dahulu agar lebih mudah dalam merepresentasikan, meskipun nantinya format yang digunakan ketika data disimpan ke dalam kartu berbentuk byte. Secara keseluruhan, panjang data yang diperlukan untuk menyimpan data pada Tabel 1 adalah 40 byte. Hasil transformasi data tersebut ke dalam format hexadesimal ditunjukkan pada Tabel 2.

Tabel 2. Hasil Transformasi Data ke dalam Format Hexadesimal

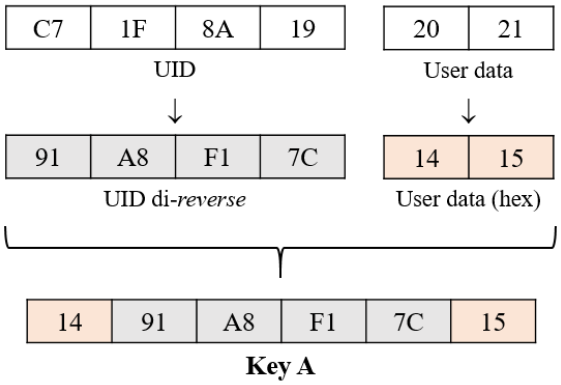
Field	Data (Hexadesimal)
TNKB	30 30 41 42 32 30 33 39 59 51
Tanggal transaksi	60 93 B2 E0
Status masuk (keluar/masuk)	01
Kode gate	00 FF
NIP	31 39 38 39 31 31 30 38 32 30 31 39 30 33 31 30 32 30
Kedaluwarsa kartu	60 93 B2 E0
Status kartu (aktif/masuk)	01

Selanjutnya, data yang telah diubah ke dalam bentuk hexadesimal tersebut dienkripsi menggunakan AES. Meskipun AES merupakan algoritma enkripsi yang menggunakan *symmetric key*, artinya *key* yang digunakan untuk melakukan enkripsi dan dekripsi adalah *key* yang sama, namun *key* yang digunakan pada penelitian ini merupakan *key* dinamis yang dibentuk berdasarkan UID *smart card*. Dengan demikian, *key* yang digunakan oleh setiap kartu nantinya akan menjadi berbeda-beda sesuai dengan UID setiap kartu tersebut.

Untuk membentuk *key* A dengan panjang 6 byte yang bersifat dinamis, digunakan empat byte UID *smart card* dan dua byte data yang bisa didefinisikan sesuai dengan kebutuhan (*user* data). Mekanisme pembentukan *key* A adalah melakukan *reverse* UID *smart card* dengan cara membaca secara terbalik semua angka (hexadesimal) pada UID, kemudian dikombinasikan dengan user data. Misalnya UID sebuah *smart card* bernilai A9 C3 1C E5, setelah dilakukan *reverse* menjadi 5E C1 3C 9A. Empat byte UID yang telah di-*reverse* tersebut digunakan sebagai byte kedua sampai kelima dari *key* A, sedangkan byte pertama dan terakhir dari *key* A diperoleh dari *user* data. Misalnya *user* data

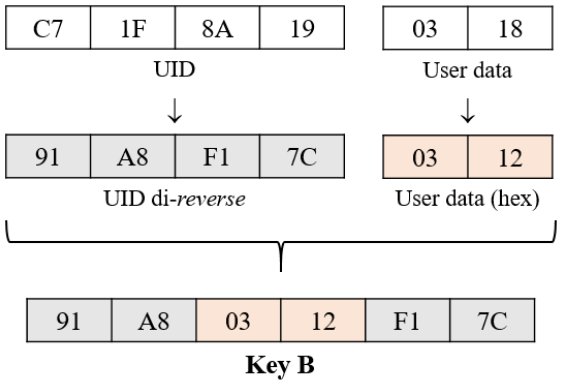
 ISSN:2302-8149
e-ISSN:2540-9719

yang digunakan adalah 20 dan 21 (desimal), yang merepresentasikan tahun saat ini, kemudian diubah ke dalam format hexadesimal menjadi 14 dan 15. Dengan demikian, *key* A untuk *smart card* tersebut adalah 14 5E C1 3C 9A 15. Tentunya *key* A ini akan berbeda untuk kartu yang lain karena UID setiap kartu berbeda-beda dan pembentukan *key* A bergantung pada UID. Mekanisme pembentukan *key* A ditunjukkan pada Gambar 3.



Gambar 3. Mekanisme Pembentukan *Key* A

Selanjutnya, pembentukan *key* B juga dilakukan dengan cara yang hampir sama seperti pembentukan *key* A. Hal yang membedakan adalah pola yang digunakan saat mengombinasikan empat byte UID yang telah di-*reverse* dengan dua byte *user* data. Untuk membentuk *key* B, dua byte pertama dari UID yang telah di-*reverse* digunakan sebagai byte pertama dan kedua dari *key* B, kemudian dua byte yang tersisa digunakan sebagai byte kelima dan keenam dari *key* B, sedangkan byte ketiga dan keempat dari *key* B diperoleh dari user data. Dalam kasus ini, UID *smart card* yang digunakan bernilai sama seperti pada penjelasan pembentukan *key* A karena kartu yang digunakan tetap, sehingga diperoleh UID yang telah di-*reverse* bernilai 5E C1 3C 9A. Selanjutnya, *user* data yang digunakan adalah bulan dan tanggal yaitu 03 dan 18 (desimal), kemudian diubah ke dalam format hexadesimal menjadi 03 dan 12. Dengan demikian, *key* B untuk *smart card* tersebut adalah 5E C1 03 12 3C 9A. Mekanisme pembentukan *key* B ditunjukkan Gambar 4.



Gambar 4. Mekanisme Pembentukan *Key* B

Beberapa *key* yang digunakan dalam penelitian ini berukuran 128 bit, 192 bit, dan 256 bit. *Key* dibentuk dengan mengombinasikan *key* A, *key* B, dan UID. Dalam hal ini, *key* A dan *key* B merupakan *key* yang digunakan untuk proses autentikasi dan UID merupakan nomor unik yang tersedia pada setiap kartu. Terdapat beberapa kombinasi *key* yang digunakan pada penelitian ini, antara lain *key* 16 byte atau 128 bit, *key* 24 byte atau 192 bit, dan *key* 32 byte atau 256 bit.

4.1 *Key* 16 byte atau 128 bit

Pembentukan *key* 128 bit dilakukan dengan cara mengombinasikan *key* A, UID, dan *key* B. Diketahui UID sebuah kartu adalah A9 C3 1C E5, maka *key* A dan *key* B dapat dibentuk dengan

 ISSN:2302-8149
e-ISSN:2540-9719

menggunakan cara yang sama seperti yang telah diilustrasikan pada Gambar 3 dan Gambar 4, sehingga 14 5E C1 3C 9A 15 sebagai *key* A dan 5E C1 03 12 3C 9A sebagai *key* B. Ilustrasi penyusunan *key* 128 bit menggunakan *key* A, UID, dan *key* B ditunjukkan pada Gambar 5.

14 5E C1 3C 9A 15	A9 C3 1C E5	5E C1 03 12 3C 9A
Key A	UID	Key B

14 5E C1 3C 9A 15 A9 C3 1C E5 5E C1 03 12 3C 9A
Key 128 bit

Gambar 5. Mekanisme Penyusunan *Key* 128 Bit

Selanjutnya, *key* 128 bit yang telah terbentuk tersebut digunakan untuk melakukan enkripsi data pada Tabel 2 yang telah tersimpan di dalam *smart card*. Hasil enkripsi menggunakan *key* 128 bit adalah DB 3E EF 51 38 2D 29 10 5A 94 83 0E 33 2B 06 0E 9F 70 95 63 91 D4 13 F2 9B D2 A8 6B E8 FE 1E 02 FC C6 5E DB 7F DF 98 8C C4 89 55 88 AC C4 5D 47. Terlihat bahwa setelah proses enkripsi, data yang awalnya mempunyai panjang 40 byte tersebut berubah menjadi 48 byte.

4.2 *Key* 24 byte atau 192 bit

Pembentukan *key* 192 bit dilakukan dengan cara mengombinasikan *key* A dan *key* B saja, tanpa UID. Hal ini dikarenakan pembentukan *key* 24 byte dapat dilakukan dengan menggunakan *key* A dan *key* B masing-masing sebanyak dua kali. Dalam hal ini, penyusunan *key* 192 bit dilakukan dengan mengombinasikan *key* A, *key* B, *key* B, dan *key* A seperti yang diilustrasikan pada Gambar 6.

14 5E C1 3C 9A 15	5E C1 03 12 3C 9A	5E C1 03 12 3C 9A	14 5E C1 3C 9A 15
Key A	Key B	Key B	Key A

14 5E C1 3C 9A 15 5E C1 03 12 3C 9A 5E C1 03 12 3C 9A 14 5E C1 3C 9A 15
Key 192 bit

Gambar 6. Mekanisme Penyusunan *Key* 192 Bit

Selanjutnya, *key* 192 bit yang telah terbentuk tersebut juga digunakan untuk melakukan enkripsi data pada Tabel 2, sehingga cipherteks yang dihasilkan adalah 6C 48 F7 8F 28 92 98 27 96 37 3D 0B EA 73 9F 36 63 6D 2E 82 41 3A 57 80 E0 3E 38 EA EE FE 13 96 57 01 D7 1F 76 D0 D9 DE 23 37 C9 30 5B D0 03 D4. Pada kasus ini, panjang data setelah enkripsi sama seperti panjang data hasil enkripsi menggunakan *key* 128 bit, yaitu 48 byte, yang menunjukkan bahwa cipherteks lebih panjang dari plainteks.

4.3 *Key* 32 byte atau 256 bit

Pembentukan *key* 256 bit dilakukan dengan cara mengombinasikan *key* A, *key* B, dan UID masing-masing sebanyak dua kali sehingga terbentuk *key* sepanjang 32 byte. Ilustrasi penyusunan *key* 256 bit menggunakan *key* A, *key* B, dan UID ditunjukkan pada Gambar 7.

14 5E C1 3C 9A 15	5E C1 03 12 3C 9A	A9 C3 1C E5	A9 C3 1C E5	5E C1 03 12 3C 9A	14 5E C1 3C 9A 15
Key A	Key B	UID	UID	Key B	Key A

14 5E C1 3C 9A 15 5E C1 03 12 3C 9A 14 5E C1 3C 5E C1 03 12 5E C1 03 12 3C 9A 14 5E C1 3C 9A 15
Key 256 bit

Gambar 7. Mekanisme Penyusunan *Key* 256 Bit

Selanjutnya, *key* 256 bit yang telah terbentuk tersebut digunakan untuk melakukan enkripsi data pada Tabel 2, sehingga cipherteks yang dihasilkan adalah F3 24 0D 88 DB 83 26 7F 83 33 C7 5C F9

68 0C CC BB CF B0 63 C9 86 38 C6 AC 22 16 1F 94 F6 8A F8 60 A1 A8 17 55 ED 0D 8F 77 26 20 ED F3 2F CD 59. Cipherteks yang dihasilkan dari proses enkripsi menggunakan *key* 256 bit juga mempunyai panjang data sebesar 48 byte, sama seperti *key* dengan ukuran lainnya. Hal ini menunjukkan bahwa panjang *key* tidak mempengaruhi panjang cipherteks. Panjang data hanya mengalami perubahan dari plainteks menjadi cipherteks, yaitu dari 40 byte menjadi 48 byte.

Untuk mengetahui kinerja algoritma yang diusulkan, skenario pengujian dilakukan pada data sepanjang 40 byte dengan menggunakan semua *key* yang telah didefinisikan, yaitu *key* berukuran 128 bit, 192 bit, dan 256 bit. Pengujian dilakukan untuk memperoleh informasi mengenai durasi penulisan data ke dalam kartu, durasi pembacaan data dari kartu, dan proses *generate key* dari UID setiap kartu. Hasil pengujian algoritma menggunakan *key* 128 bit, 192 bit, dan 256 bit masing-masing ditunjukkan pada Tabel 3 sampai dengan Tabel 5. Durasi penulisan data dihitung mulai dari proses enkripsi 40 byte data (plainteks) sampai dengan penyimpanan data ke dalam *smart card*, sedangkan durasi pembacaan data dihitung mulai dari proses pembacaan data pada *smart card* sampai dengan proses dekripsi data.

Tabel 3. Hasil Pengujian Menggunakan *Key* 128 bit

No	UID	*Key*	Durasi Penulisan Data (ms)	Durasi Pembacaan Data (ms)
1	A9C31CE5	145EC13C9A15A9C31CE55EC103123C9A	73	91
2	290DA66E	14E66AD09215290DA66EE66A0312D092	71	89
3	3206C64D	14D46C6023153206C64DD46C03126023	70	91
4	FA8FA015	14510AF8AF15FA8FA015510A0312F8AF	71	89
5	0B7E7C22	1422C7E7B0150B7E7C2222C70312E7B0	71	87
		Rata-rata	71.2	89.4

Tabel 3 menunjukkan hasil enkripsi dan dekripsi pada *smart card* menggunakan *key* 128 bit yang diuji cobakan pada lima kartu yang berbeda. Berdasarkan Tabel 3 tersebut, diketahui bahwa proses pembacaan data membutuhkan waktu lebih lama dibandingkan dengan proses penulisan data, yaitu waktu rata-rata sebesar 71.2 ms untuk penulisan data dan 89.4 ms untuk pembacaan data. Hal ini dikarenakan pada proses pembacaan terdapat proses dekripsi data yang membutuhkan waktu komputasi lebih tinggi.

Tabel 4. Hasil Pengujian Menggunakan *Key* 192 bit

No	UID	*Key*	Durasi Penulisan Data (ms)	Durasi Pembacaan Data (ms)
1	A9C31CE5	145EC13C9A155EC103123C9A5EC10312 3C9A145EC13C9A15	73	87
2	290DA66E	14E66AD09215E66A0312D092E66A0312 D09214E66AD09215	69	86
3	3206C64D	14D46C602315D46C03126023D46C03126 02314D46C602315	70	88
4	FA8FA015	14510AF8AF15510A0312F8AF510A0312 F8AF14510AF8AF15	70	87
5	0B7E7C22	1422C7E7B01522C70312E7B022C70312E 7B01422C7E7B015	72	92
		Rata-rata	70.8	88

Hasil pengujian pada Tabel 4 tetap menggunakan lima kartu yang sama seperti pada pengujian sebelumnya, hal yang membedakan adalah *key* yang digunakan yaitu mempunyai panjang 192 bit. Berdasarkan hasil pengujian pada Tabel 4 tersebut, diperoleh waktu rata-rata penulisan data sebesar 70.8 ms dan pembacaan data sebesar 88 ms.

 ISSN:2302-8149
e-ISSN:2540-9719

Tabel 5. Hasil Pengujian Menggunakan *Key* 256 bit

No	UID	*Key*	Durasi Penulisan Data (ms)	Durasi Pembacaan Data (ms)
1	A9C31CE5	145EC13C9A155EC103123C9A145EC13C5 EC103125EC103123C9A145EC13C9A15	74	91
2	290DA66E	14E66AD09215E66A0312D09214E66AD0E 66A0312E66A0312D09214E66AD09215	73	89
3	3206C64D	14D46C602315D46C0312602314D46C60D 46C0312D46C0312602314D46C602315	70	88
4	FA8FA015	14510AF8AF15510A0312F8AF14510AF85 10A0312510A0312F8AF14510AF8AF15	72	87
5	0B7E7C22	1422C7E7B01522C70312E7B01422C7E722 C7031222C70312E7B01422C7E7B015	71	87
		Rata-rata	72	88.4

Sama halnya dengan Tabel 3 dan Tabel 4, Tabel 5 merupakan hasil pengujian algoritma dengan menggunakan lima kartu yang berbeda dan *key* sepanjang 256 bit. Waktu rata-rata yang dibutuhkan untuk melakukan penulisan data sebesar 72 ms dan pembacaan data sebesar 88.4 ms.

Dengan menggunakan lima kartu yang telah digunakan sebelumnya, skenario pengujian juga dilakukan untuk melakukan proses penulisan data dan pembacaan data tanpa adanya enkripsi dan dekripsi. Hasil pengujian ini ditunjukkan pada Tabel 6.

Tabel 6. Hasil Pengujian Tanpa Enkripsi dan Dekripsi

No	UID	Durasi Penulisan Data (ms)	Durasi Pembacaan Data (ms)
1	A9C31CE5	71	87
2	290DA66E	70	87
3	3206C64D	69	87
4	FA8FA015	73	87
5	0B7E7C22	71	88
	Rata-rata	70.8	87.2

Pada Tabel 6, hasil pengujian menunjukkan bahwa waktu rata-rata yang diperlukan untuk melakukan penulisan dan pembacaan data tanpa proses enkripsi dan dekripsi cenderung lebih kecil dibandingkan dengan waktu rata-rata yang diperlukan ketika terdapat proses enkripsi dan dekripsi, yaitu sebesar 70.8 ms untuk penulisan data dan 87.2 ms untuk pembacaan data. Walaupun demikian, perbedaan waktu yang diperlukan untuk proses baca dan tulis data ketika menggunakan enkripsi tidak terlalu jauh berbeda dengan proses baca dan tulis data tanpa enkripsi, yaitu sekitar 1 sampai dengan 2 ms. Hal ini tentunya tidak sebanding dengan keuntungan yang diberikan oleh penggunaan proses enkripsi data menggunakan algoritma AES. Selain dilakukan pengamanan pada data, mekanisme pengamanan lainnya juga dilakukan di sisi *smart card* dengan memanfaatkan UID. Dengan demikian, antara satu kartu dengan kartu lainnya akan mempunyai *key* yang berbeda, serta proses penulisan dan pembacaan data tidak membutuhkan waktu yang terlalu lama.

5 Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan, proses enkripsi dan dekripsi data pada *smart card* menggunakan algoritma AES terbukti dapat diandalkan karena terdapat mekanisme penggunaan *key* yang bersifat dinamis, yaitu pembentukan *key* berdasarkan UID dari setiap *smart card*. Dengan demikian, meskipun AES termasuk algoritma yang bersifat *symmetric*, namun adanya *key* dinamis ini menjadikan *key* yang digunakan setiap kartu menjadi berbeda-beda dan sulit untuk ditebak. Tentunya mekanisme ini menjadi sangat bermanfaat ketika *smart card* digunakan untuk aplikasi-aplikasi yang di dalamnya terdapat informasi yang bersifat rahasia. Penambahan enkripsi data menggunakan AES

menjadikan data lebih aman, namun waktu yang diperlukan untuk proses penulisan dan pembacaan data pada *smart card* menjadi semakin bertambah. Hal ini dikarenakan setelah dilakukan enkripsi, panjang data yang awalnya 40 byte (plainteks) akan bertambah menjadi 48 byte (cipherteks) dan menyebabkan durasi penulisan dan pembacaan data menjadi sedikit lebih lama dibandingkan dengan tanpa adanya proses enkripsi. Waktu rata-rata yang diperlukan untuk melakukan penulisan dan pembacaan data dengan adanya enkripsi sekitar 2 ms lebih lama dibandingkan dengan tanpa adanya enkripsi. Selisih waktu tersebut tidaklah berarti jika dibandingkan dengan keamanan yang diperoleh ketika digunakan mekanisme pengamanan data menggunakan algoritma AES dan *key* dinamis berdasarkan UID setiap *smart card*. Setelah proses enkripsi data, ukuran plainteks biasanya akan cenderung membesar ketika menjadi cipherteks, sehingga dapat mempengaruhi waktu komputasi apabila ukuran data yang disimpan di dalam *smart card* cukup besar. Oleh karena itu, pada penelitian berikutnya, mekanisme kompresi menggunakan algoritma Huffman akan dilakukan untuk memperkecil ukuran cipherteks tanpa mengubah isi data yang tersimpan di dalam *smart card*.

Referensi

[1] H. Taherdoost, "Appraising the Smart Card Technology Adoption: Case of Application in University Environment," *Procedia Eng.*, vol. 181, pp. 1049–1057, 2017.

[2] D. Priyasta, W. Cesar, Y. Susanti, and J. Junde, "Java Card Approach to Emulate The Indonesian National Electronic ID Smart Cards," *Sci. J. Informatics*, vol. 5, no. 2, pp. 224–234, 2018.

[3] M. Akbar and I. Effendy, "Implementasi Aplikasi Kehadiran Perkuliahan di Kelas Menggunakan Pembaca RFID pada E-KTP," *J. Ilm. Matrik*, vol. 19, no. 2, pp. 151–160, 2017.

[4] L. Miliani, M. S. Purwanegara, and M. T. D. Indriani, "Adoption Behavior of E-Money Usage," *Inf. Manag. Bus. Rev.*, vol. 5, no. 7, pp. 369–378, 2013.

[5] Pranoto and S. S. Salsabila, "Eksistensi Kartu Kredit dengan Adanya Electronic Money (E-Money) Sebagai Alat Pembayaran yang Sah," *J. Priv. Law*, vol. 6, no. 1, pp. 24–33, 2018.

[6] R. E. Rahmawati and M. R. Maika, "Penerapan Model UTAUT terkait akseptasi mahasiswa terhadap Cashless Payment di masa Pandemi COVID-19," *J. Ekon. Mod.*, vol. 17, no. 1, pp. 1–14, 2021.

[7] B. W. Harimurti, W. Kurniawan, and H. Nurwarsito, "Sistem Pengelolaan Parkir Dengan NFC," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 2, no. 6, pp. 2038–2045, 2018.

[8] D. I. Putra and W. Syahputra, "Sistem Pembayaran Parkir Menggunakan Near Field Communication Berbasis Android dan Teknologi Internet of Things," *J. Nas. Teknol. dan Sist. Inf.*, vol. 3, no. 1, pp. 153–164, 2017.

[9] H. Setiadi, Y. Priyandari, and S. I. Cahyono, "Implementation of Parking System Based on Radio Frequency Identification (RFID) at the Faculty of Engineering Sebelas Maret University," *ITSMART J. Ilm. Teknol. dan Inf.*, vol. 6, no. 1, pp. 39–44, 2017.

[10] S. A. Utomo, D. Utomo, and B. W. Yohanes, "Sistem e-money berbasis Contactless Smartcard dengan Teknologi RFID," *Techné J. Ilm. Elektrotek.*, vol. 15, no. 1, pp. 67–75, 2016.

[11] Ratnadewi, R. P. Adhie, Y. Hutama, J. Christian, and D. Wijaya, "Implementation and performance analysis of AES-128 cryptography method in an NFC-based communication system," *World Trans. Eng. Technol. Educ.*, vol. 15, no. 2, pp. 178–183, 2017.

[12] B. Padmavathi and S. R. Kumari, "A Survey on Performance Analysis of DES, AES and RSA Algorithm along with LSB Substitution Technique," *Int. J. Sci. Res.*, vol. 2, no. 4, pp. 170–174, 2013.

[13] Laurentinus, H. A. Pradana, D. Y. Sylfania, and F. P. Juniawan, "Perbandingan kinerja RSA dan AES terhadap kompresi pesan SMS menggunakan algoritme Huffman," *J. Teknol. dan Sist. Komput.*, vol. 8, no. 3, pp. 171–177, 2020.

[14] Noprianto and V. N. Wijayaningrum, "End to End Enkripsi Menggunakan Advanced Encryption Standard pada Perangkat Internet of Things," *J. Sist. Inf. dan Bisnis Cerdas*, vol. 14, no. 2, pp. 98–107, 2021.

[15] A. M. Sison, B. T. Tanguilig, B. D. Gerardo, and Y. C. Byun, "Implementation of Improved DES Algorithm in Securing Smart Card Data," in *Computer Applications for Software Engineering, Disaster Recovery, and Business Continuity*, Berlin, Heidelberg: Springer, 2012,

 ISSN:2302-8149
e-ISSN:2540-9719

pp. 252–263.
[16] N. Aleisa, "A comparison of the 3DES and AES encryption standards," *Int. J. Secur. its Appl.*, vol. 9, no. 7, pp. 241–246, 2015.
[17] H. A. Darwito, M. Yuliana, and R. Soelistijorini, "Implementasi Algoritme 3DES pada Sistem Sharing Electronic Health Record (EHR) Berbasis Cloud," *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 6, no. 3, pp. 284–290, 2017.
[18] F. Maqsood, M. Ahmed, M. M. Ali, and M. A. Shah, "Cryptography: A Comparative Analysis for Modern Techniques," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 6, pp. 442–448, 2017.
[19] H. O. Alanazi, B. B. Zaidan, A. A. Zaidan, H. A. Jalab, M. Shabbir, and Y. Al-Nabhani, "New Comparative Study Between DES, 3DES and AES within Nine Factors," *J. Comput.*, vol. 2, no. 3, pp. 152–157, 2010.
[20] A. Y. Ananta, Noprianto, and V. N. Wijayaningrum, "Desain Sistem Smart Attendance Menggunakan Kombinasi Smart Card dan Sidik Jari," *Sistemasi*, vol. 9, no. 3, pp. 480–492, 2020.