Pengamanan *Mnemonic Phrase* Menggunakan *Modified Advanced Encryption Standart*

M. Adharis Adlani[1], Ricky Eka Putra[2]
[1 2]jurusan Teknik Informatika Fakultas Teknik Universitas Negeri Surabaya
[1]m.adharis.18072@mhs.unesa.ac.id
[2]rickyeka@unesa.ac.id

Abstrak - Keamanan merupakan tindakan mencegah terjadinya bahaya, ancaman, maupun serangan, terutama keamanan penyimpanan asset *cryptocurrency* pada aplikasi *mobile wallet* kripto smartphone. Keamanan berperan penting melindungi data *wallet* dari serangan pencurian asset. Hasil obervasi yang dilakukan banyak pengguna menyimpan kata pemulihan *(mnemonic phrase)* dompet mereka pada layanan cloud seperti *google drive* dan *one drive* hal ini sangat rentan peretasan data apabila akun email sudah terbobol kemudian hacker mengambil data yang berharga untuk keuntungan diri sendiri. Para ilmuwan komputer menciptakan suatu algoritma kemanan kriptografi untuk mengamankan data dari pencurian. Algoritma AES (Advanced Encryption Standart) merupakan algoritma kriptografi *cipher block symmetric* dimana untuk memperoleh data yang sudah dienkripsi menggunakan kunci rahasia atau cipher key pengguna harus memasukkan kunci yang sama ketika melakukan proses penyandian data (enkripsi). AES dibagi menjadi tiga berdasarkan panjang kunci, yaitu AES-128 memiliki panjang kunci 128bit, AES-192 memiliki panjang kunci 192bit, dan AES-256 memiliki panjang kunci 256bit. Sistem yang akan dibuat pada penilitian ini yaitu memodifikasi putaran pada algoritma AES standar yang memiliki 10 putaran menjadi 16 putaran. Modifikasi yang dilakukan dengan menambah jumlah putaran algoritma membuat sistem lebih aman dan kuat dari serangan serta membutuhkan waktu komputasi lebih lama bagi hacker untuk memecahkan enkripsi. Pengujian yang dilakukan yaitu membandingkan hasil *ciphertext*, waktu dan *avalanche effect*. Hasil pengujian menunjukkan perbedaaan output *ciphertext* pada file yang sama setelah enkripsi, kemudian waktu yang dibutuhkan pada algoritma AES modifikasi dalam proses enkripsi dan dekripsi lebih lama dibanding AES standar. Sedangkan hasil uji pada *avalanche effect* menunjukkan perubahan bit yang terjadi pada AES standar sebesar 48,6% lalu pada AES modifikasi sebesar 55,6%. Suatu *avalanche effect* mempunyai hasil baik jika perubahan bit terjadi sebesar 45-60% (separuh atau lebih). Semakin banyak bit yang berubah mengakibatkan algoritma kriptografi semakin sulit untuk dipecahkan.

Kata Kunci: kriptografi, AES (*Advanced Encryption Standart*), pengamanan file, *cryptocurrency, modified aes,*

I. PENDAHULUAN

Sistem transaksi saat ini sudah berkembang pesat, begitu juga dengan gaya investasi masa kini seperti *cryptocurrency*. *Cryptocurrency* merupakan mata uang virtual yang dicetak oleh program komputer menggunakan teknologi kriptografi dengan perhitungan sistematis untuk menyusun sandi atau kode untuk melindungi dari adanya pemalsuan (Wicaksono Sakti Yudo, 2018). *Cryptocurrency* bisa dijadikan sebagai alat investasi, pembayaran transaksi pembelian online, dan ditukarkan menjadi mata uang lain seperti Dollar, Yen, Rupiah dan mata uang lainnya. Pada saat menyimpan asset mata uang digital dibutuhkannya dompet/wallet digital agar memudahkan saat melakukan transaksi, aplikasi yang umum digunakan seperti *metamask*, *trustwallet,* dan *safepal wallet*. Langkah pertama setelah menginstall aplikasi dompet digital adalah membuat *mnemonic phrase* yang sudah disediakan oleh aplikasi digital wallet. Secara umum *Mnemonic Phrase* adalah susunan kata baru untuk membantu mengingat berbagai jenis informasi yang memuat data tertentu (Barcroft Joe, 2012). *Mnemonic phrase* adalah kata pemulihan yang harus disimpan secara rahasia agar nantinya jika ingin masuk pada wallet digital menggunakan perangkat berbeda bisa dilakukan dengan mudah dengan menulis susunan kata yang sesuai dan tidak diketahui oleh orang lain agar tidak kehilangan asset digital. Pengguna wajib menyimpannya dengan hati-hati ditempat yang aman idealnya di lingkungan yang tidak terhubung ke internet. Berdasarkan pengamatan yang dilakukan banyak pengguna yang menyimpan *mnemonic phrase* mereka pada layanan cloud hal ini tentu tidak aman terlebih lagi layanan cloud tidak sepenuhnya aman dari cela peretasan pencurian data pribadi. Dibuatnya penelitian ini membantu mengamankan asset digital dari pencurian data. Dengan memanfaatkan kriptografi, yaitu ilmu untuk melindungi pengiriman data dengan mengubah menjadi kode yang hanya bisa dibaca oleh penerima yang memiliki kunci untuk mengubah kode tersebut menjadi pesan terbaca (Pabokory N. F. dkk. 2015). Salah satunya menggunakan teknik enkripsi MAES *(Modified Advanced Encryption Standart).* MAES adalah modifikasi dari algoritma AES yang menggunakan kunci kriptografi 128, 192 , dan 256 bit untuk mengenkripsi dan mendekripsi data dalam blok 128 bit (Telegarapu P. dkk. 2011). Alasan menggunakan metode ini karena ditetapkan sebagai standart algoritma kriptografi oleh pemerintah Amerika Serikat secara resmi pada

tanggal 22 mei 2002. Usulan sistem yang akan dibuat cara kerjanya pengguna memasukkan pesan yang akan dienkripsi lalu menentukan juga kunci yang akan dibuat sebagai pengaman agar nantinya pesan yang sudah dienkripsi tidak dapat didekripsi dengan mudah jika tidak memiliki kunci yang sudah dibuat tadi, dengan catatan pengguna menyimpan kunci dengan baik dan aman (sebisa mungkin menghindari tempat yang terhubung ke jaringan internet).

Penggunaan kriptografi sudah dilakukan sejak zaman yunaini kuno untuk mengamankan pesan agar tidak bisa dibaca oleh orang lain selain penerima yang bisa. Kriptografi berasal dari kata *crypto* dan *grapho* yang artinya menulis dalam Bahasa yunaini. Kriptografi adalah teknik perhitungan yang berhubungan dengan keamanan mulai dari kerahasiaan informasi, integritas data, otentikasi entitas dan otentikasi keaslian data (Talbot John & Welsh Dominic. 2006). Kriptografi bukan hanya sekadar jasa keamanan informasi, melainkan juga memiliki teknik yang bisa dipelajari bertujuan untuk memberikan pengamanan berlebih pada data. Alur proses pengamanan lebih dikenal dengan enkripsi yaitu mengubah data asli *(plain text)* menjadi data yang tidak bisa dibaca *(cipher text)* dengan menyandikan kode kunci *(cipher key)* sebagai pengaman agar tidak mudah dibaca, sedangkan dekripsi adalah mengubah data yang tidak bisa dibaca *(cipher text)* menjadi data yang bisa dibaca *(plain text)* dengan memasukkan kunci *(cipher key)* yang sama pada saat proses penyandian data.

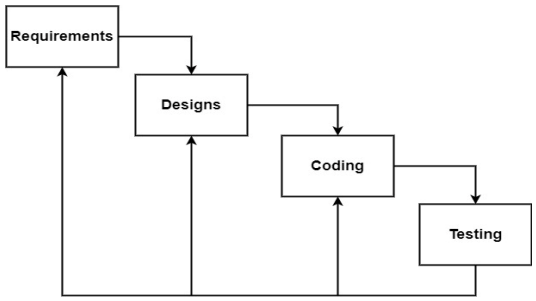
Salah satu algoritma kriptografi yaitu AES *Advanced Encryption Standart* yang memanfaatkan teknik blok simetris dalam proses pengamanan data . Pengembang dari algoritma bernama Dr. Vincent Rijmen dan Dr. Joan Daemen berasal dari Belgia pada tahun 1997. Sebagai kandidat AES, mereka berdua mengajukan algoritma ini dan berhasil menjadikannya proposal terpilih bagi AES oleh NIST (National Institute of Standard and Technology) pada tanggal 26 November 2001 (Daemen, J., & Rijmen, V. (n.d.), NIST).

Tanggal 26 mei 2002 menteri perdagangan amerika serikat pada saat itu Donald Evans menyetujui AES menjadi standar algoritma enkripsi sebagai metode dalam pengamanan data atau pesan yang efektif digunakan serta dapat diakses oleh pemerintah dan public. AES dapat diimplementasikan dan berjalan dengan baik melalui *software* maupun *hardware.* Alasan dibuatnya standart algoritma kriptografi yang baru dibuat untuk menggantikan algoritma DES *(Data Encryption Standart)* yang diklaim sudah tidak terjamin lagi keamanannya kemudian digantikan oleh AES *(Advanced Encryption Standart)*. AES memiliki 4 jenis transformasi pada enkripsi yaitu *AddRoundKey, SubBytes, ShiftRows, dan MixColumns.* Sedangkan pada proses dekripsi merupakan transformasi dari kebalikannya yaitu : *InvShiftRows, InvSubBytes,* dan *InvMixColumns.*

Implementasi yang dilakukan pada penelitian kali ini adalah mengamankan kata pemulihan *(mnemonic phrase)* pada aplikasi dompet *cryptocurrency* dari pencurian asset. Perancangan aplikasi kali ini diimplementasikan dalam bahasa java menggunakan platform mobile android studio dengan memodifikasi putaran pada AES untuk mendapatkan keamanan tambahan sehingga sulit untuk dipecahkan dari pada algoritma AES standart yang telah ada. Penelitian yang dilakukan oleh Puneet Kumar dan Shashi B. Rana menghasilkan peningkatan kemanan pada proses enkripsi atau dekripsi dengan memodifikasi putaran/iterasi. Penelitian yang telah dilakukan dengan cara mengubah AES standar menjadi 16 putaran untuk meningkatkan sistem kemanan (Kumaar, P., & Rana, B. S. 2015). Penelitian kali ini mengembangkan aplikasi enkripsi untuk mengamankan file gambar dan dokumen berbasis teks maupun data dengan mengimplementasikan penelitian yang sudah ada.

II. METODE PENELITIAN

Proses penelitian menggunakan model pengembangan *waterfall* sebagai rancangan awal untuk mengambil proses dasar seperti spesifikasi, desain sistem, koding, dan pengujian sebagai presentasi dari bagian fase proses. Waterfall adalah teknik morfologi yang digunakan untuk merancang suatu sistem agar pengembangan dapat berjalan dengan cepat dan efektif (Golodetz dkk. 2014). Tahapan alur penelitian ditunjukkan seperti gambar 1 berikut.



Gbr 1. Metode penelitian

Proses tahapan penelitian yang akan dilakukan yaitu:

1. Analisis Kebutuhan *(Requirements)*
   Tahap awal dari menganalisa kebutuhan apa saja yang akan diterapkan dalam merancang system yang akan dibangun, sekaligus mempertimbangkan fitur apa saja yang akan dibuat.

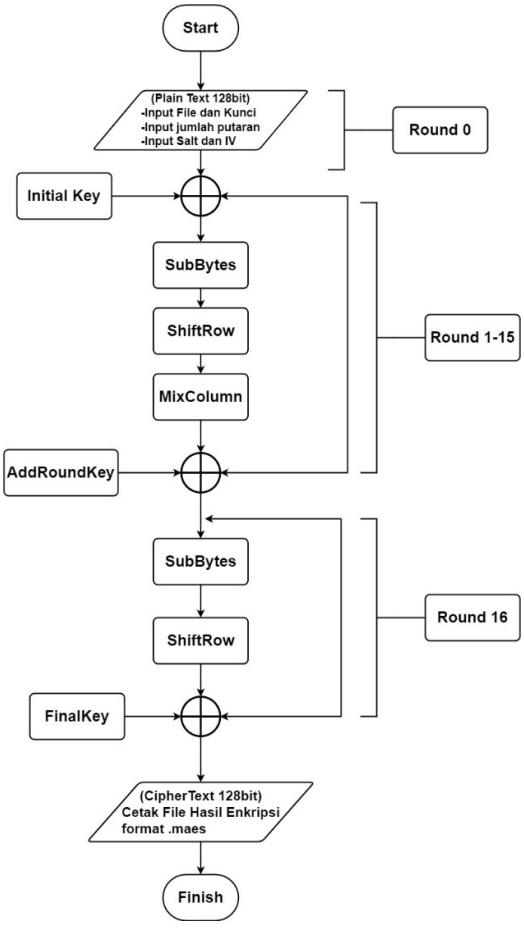
2. Desain *(Design)*
   Proses tahap ini yaitu meninjau hasil dari analisis kebutuhan lalu diimplementasikan ke dalam bentuk desain sistem yang akan dibuat, tujuannya agar mempermudah pengguna dalam menjalankan sebuah program/aplikasi. Dengan menyempurnakan desain sebagus mungkin sesuai dengan kebutuhan dan dapat dipahami oleh pengguna.

3. Pengkodean *(Coding)*
   Merupakan tahap implementasi algoritma kedalam bahasa pemrograman sesuai dari rancangan

kerja sistem, lalu menggabungkan kedalam desain antarmuka pengguna menjadi bentuk aplikasi bebasis mobile.

4. Pengujian *(Testing)*
   Tahap akhir dari proses penelitian yaitu melakukan pengujian terhadap system yang telah dirancang sesuai dari analisa kebutuhan awal. Fungsi diterapkannya pengujian bertujuan untuk mengetahui hasil dari perbedaan algoritma yang sudah dimodifikasi dengan algoritma asli dalam tingkat keamanan.

Alur sistem yang akan dibuat pada penelitian ini yaitu memodifikasi algoritma AES dengan menambahkan jumlah putaran sesuai dengan kebutuhan pengguna sebagai upaya untuk meningkatkan keamanan data/file agar tidak dimudah dipecahkan atau dibaca. Sistem aplikasi yang akan dibuat berjalan pada android berbasis mobile menggunakan bahasa pemograman java. Alur diagram ditunjukkan pada gambar 2.

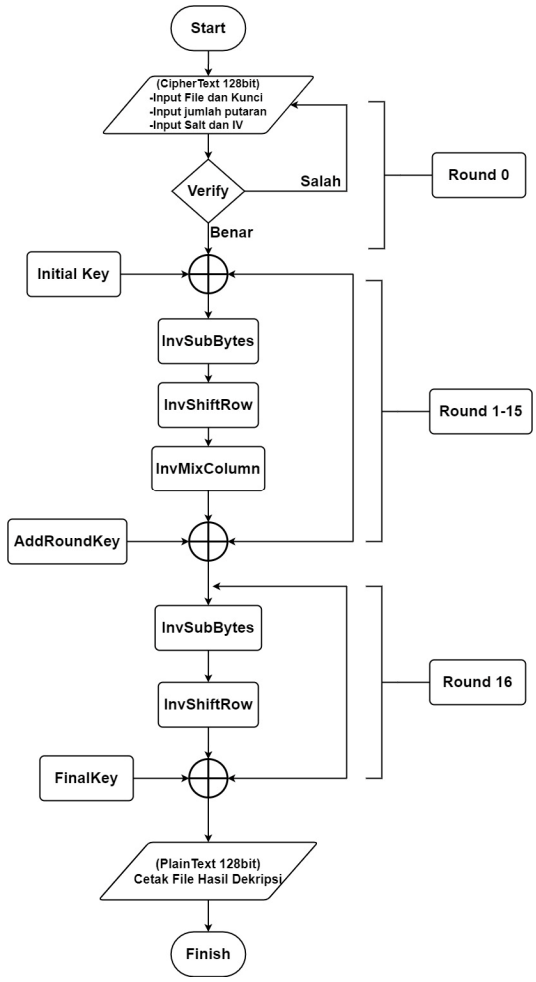


Gbr2. Proses enkripsi modifikasi AES

Alur proses enkripsi dari modifikasi AES adalah sebagai berikut:

1. Pengguna terlebih dahulu memilih file yang ada pada penyimpanan lokal perangkat. File yang masuk kemudian ditampung ke dalam blok data yang berbentuk array 2 dimensi yaitu state array. Kemudian data dikonversi terlebih dahulu ke dalam bit menggunakan kode ASCII. Lalu kode ASCII dikonversikan ke dalam bentuk heksadesimal.
2. Selanjutnya state array melakukan operasi XOR menggunakan cipher key (kunci) yang telah dimasukkan dan dikonversi ke dalam bentuk array kunci. Cipher key yang pertama diberi nama initial key. Sedangkan proses ini dinamakan Addroundkey.
3. Selanjutnya proses substitusi bytes (SubBytes) yaitu hasil dari operasi XOR disubstitusikan dengan tabel Rijndael S-Box.
4. Kemudian dilanjutkan menggunakan proses ShiftRows yaitu pergeseran pada setiap elemen blok yang bekerja pada setiap baris, dimana baris pertama tidak mengalami pergeseran, baris ke-2 melakukan pergeseran 1 byte, baris ke-3 melakukan pergeseran 2 byte, dan baris ke-4 melakukan pergeseran 3 byte.
5. Selanjutnya yaitu dijalankan proses transformasi MixColumns yaitu masing-masing elemen dari cipher block dikalikan dengan matriks yang telah ada. Pengalian menggunakan simbol dot sama halnya dengan perkalian matriks biasa, kemudian hasil dari perkalian ditampung pada sebuah cipher block menu.
6. Setelah proses MixColumns selesai, selanjutnya dilakukan fungsi AddRoundKey yaitu melakukan operasi XOR state array dengan round key.
7. Jika proses AddRoundKey selesai maka dilakukan pengulangan kembali proses SubBytes dan seterusnya sampai 15 putaran.
8. Setelah sampai putaran terakhir atau putaran ke-16 dan sudah melakukan ShiftRow, dilanjutkan dengan proses operasi XOR antara hasil state array dengan array kunci terakhir atau final key. .
9. Proses terakhir yaitu data yang keluar berupa ciphertext file berformat .maes disimpan pada direktori file yang sama dengan lokasi file awal tersebut disimpan.

Proses selanjutnya yaitu alur dekripsi ditunjukkan pada gambar 3.



Gbr 3. Proses dekripsi modifikasi AES

Alur proses dekripsi dari modifikasi AES yakni:

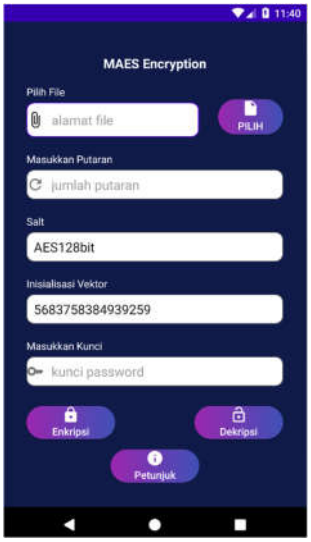
1. Data yang sudah dienkripsi menjalani transformasi AddRoundKey yakni perpaduan antara kunci ronde ke-16 melalui state maupun blok array yang mengandung hasil ciphertext enkripsi melalui perhitungan XOR.
2. Sesudah proses AddRoundKey dilangsungkan maka proses berikutnya yakni transformasi Inverse Shiftrows yang yakni kebalikan dari transformasi Shiftrows.
3. Selanjutnya dilakukan transformasi Inverse SubBytes yang yakni transformasi yang terakhir atas putaran pertama maupun pra-ronde. Transformasi ini dilangsungkan proses substitusi atas state hasil transformasi yang terdahulu memakai tabel Inverse SubBytes.
4. Berikutnya masuk putaran selanjutnya maupun ronde ke-1 yang berlangsung transformasi AddRoundKey yakni perpaduan antara kunci ronde ke-15 melalui state hasil putaran terdahulu.
5. proses transformasi berikutnya yakni Inverse MixColumns yakni melangsungkan perkalian matriks antara state hasil AddRoundKey melalui matriks yang sudah tergambar guna proses dekripsi.
6. Sesudah transformasi Inverse MixColumns dilangsungkan, kemudian melangsungkan transformasi Inverse ShiftRows serta Inverse SubBytes.
7. Kemudian proses selanjutnya yaitu mengulangi lagi proses transformasi invers Shiftrows,inv SubBytes, dan inv mix column sampai round (putaran) 15.
8. Proses putaran terakhir, hanya melakukan tranformasi AddRoundKey dimana state yang dihasilkan pada round (putaran) 15 dicampur dengan kunci round ke 0 atau kunci cipher.
9. Setelah itu, mencetak hasil file plaintext atau file asli menandakan proses dekripsi berhasil.

Alur proses enkripsi serta dekripsi yang tergambar pada gambar 2 dan 3 yang perlu diperhatikan pada saat sebelum melakukan enkripsi, pengguna harus mengingat kode (angka dan huruf) pada 4 kolom masukan yaitu putaran, salt, inisialisasi vektor dan kunci cipher. Jika salah satu pengguna lupa kode pada kolom masukan tersebut maka file yang sudah dienkripsi akan sulit untuk dipecahkan atau didekripsi kembali. Mengingat ini adalah algoritma modifikasi AES jika dibandingkan dengan algoritma AES standar pengguna hanya perlu mengingat kunci cipher pada enkripsi mereka.

III. HASIL DAN PEMBAHASAN

A. *Implementasi Program*

Penelitian diterapkan pada aplikasi mobile sebagai kebutuhan fungsional. Pengembangan aplikasi menggunakan bahasa pemograman java yang ada pada android studio. Implementasi dilakukan pada mobile dikarenakan banyak pengguna lebih mudah mengakses aplikasi dari perangkat smartphone mereka daripada mengakses lewat web dikarenakan harus mengetik alamat link dahulu yang akan dituju. Alasan selanjutnya masalah keamanan, proses enkripsi lebih baik jika offline dan tidak terhubung ke internet untuk menghindari error. Gambar 4 yakni tampilan utama aplikasi enkripsi android dimana terdapat beberapa fitur modifikasi yang bisa digunakan oleh pengguna diantaranya mengatur jumlah putaran dan mengatur inisialisasi vector menggunakan angka acak sebanyak 16 byte. Selanjutnya terdapat menu lain yaitu pilih file, Salt, dan masukkan kunci, serta terdapat total empat tombol yaitu enkripsi, petunjuk, dekripsi, dan pilih file *(input)*. Fungsi kolom Salt pengguna dapat mengubah kode yang sesuai dengan kebutuhan ini bertujuan untuk memberi bit acak kedalam input saat proses derivasi kunci agar tidak terjadi kesamaan data (Krisanty, T., ITB) .



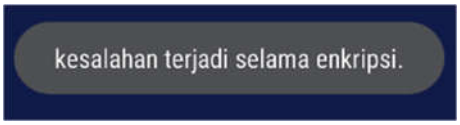
Gbr 4. Tampilan utama program

Berikut adalah rangkaian yang dilakukan pengguna untuk menjalankan proses enkripsi dan dekripsi:

1. Pengguna memilih file yang akan dienkripsi dengan mengklik tombol pilih lalu diarahkan ke direktori lokal android.
2. Pengguna memasukkan jumlah putaran yang akan digunakan pada proses enkripsi bertujuan untuk meningkatkan keamanan dari enkripsi standart dimana hanya berjumlah 10 putaran. Hal yang perlu diperhatikan ketika melakukan dekripsi pengguna harus mengingat jumlah putaran yang sama saat proses enkripsi agar file yang sudah dienkripsi berhasil didekripsi dengan baik.
3. Pengguna memasukkan kode bebas (huruf atau angka) pada kolom Salt, salt berfungsi memberikan bit acak kedalam input saat proses derivasi kunci untuk memperkuat data yang dienkripsi.
4. Selanjutnya yaitu pengguna memasukkan angka acak sebanyak 16 atau 16 byte (berupa angka) pada kolom inisialisai vektor proses ini opsional tergantung dari kebutuhan pengguna ingin mengubah atau menggunakan masukan bawaan. Peru diperhatikan ketika proses dekripsi pengguna harus memasukkan angka yang sama saat proses enkripsi.
5. Pengguna harus memasukkan password/kunci pada kolom kunci sesuai dengan keinginan. Password harus disimpan dengan aman tanpa diketahui orang lain sebisa mungkin disimpan lingkungan yang tidak terhubung jaringan internet.
6. Pengguna memilih opsi proses antara enkripsi dan dekripsi kemudian klik
7. Jika proses enkripsi berhasil maka akan muncul pesan pop up yang berisi nama file telah diubah ke dalam format .maes yaitu format file enkripsi seperti pada gambar 5.

Gbr 5. Notifikasi enkripsi berhasil

Jika kondisi file yang akan dienkripsi error atau tidak bisa diproses, besar kemungkinan terjadi saat eksekusi program dimulai. Faktor yang mempengaruhi error adalah nilai inisialisi vektor harus diisi sebanyak 16 byte jika kurang dari angka tersebut maka bisa dipastikan proses enkripsi akan error pada gambar 6.



Gbr 6. Notifikasi enkripsi error

Adapun tampilan dari halaman ketika proses enkripsi selesai, pengguna akan diarahkan ke halaman petunjuk penggunaan aplikasi seperti gambar 7.

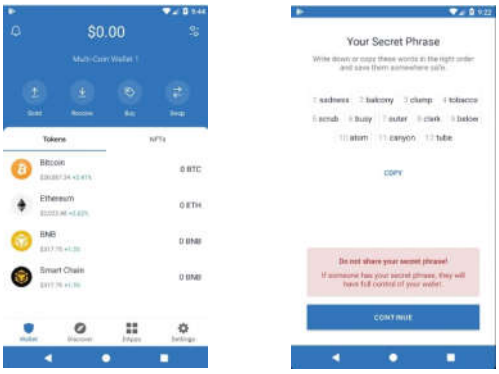
Gbr 7. Halaman petunjuk

Ketika proses enkripsi selesai file yang telah dienkripsi akan disimpan ke dalam direktori dimana file itu berada dengan format ekstensi .maes. Pengguna diharapkan mengingat kunci, jumlah putaran dan inisiasi vektor (opsional) dikarenakan hasil yang dicetak hanyalah file hasil enkripsi tidak termasuk data kunci.

B. *Hasil Pengujian*

Proses pengujian meliputi pengujian hasil *ciphertext*, waktu dan *avalanche effect* dimana tujuannya untuk mengetahui perbandingan enkripsi serta dekripsi antara algoritma AES standar 10 putaran melalui AES modifikasi 16 putaran. Pengujian dilakukan melalui beragam macam dokumen. serta ukuran file yang tidak sejenis. Hasil uji diatas digunakan sebagai acuan pembanding guna mengkaji apakah sistem yang sudah dibuat bisa beroprasi dengan baik dengan sistem asli sebelumnya.

Pengujian keamanan dilakukan pada aplikasi dompet digital *cryptocurrency* trustwallet dengan mengambil sampel *mnemonic phrase* gambar 8 sebelah kanan. Jenis masukan file yang diamankan yaitu teks dan tangkapan gambar berformat jpg & png.



Gbr 8. Aplikasi trustwallet dan mnemonic phrase

Enkripsi teks diproses dengan menyalin teks sampel pada gambar 8 sebelah kanan, kemudian enkripsi dilakukan pada file berformat .txt dan .docx dikarenakan banyak pengguna menyimpan arsip atau dokumen menggunakan format tersebut pada layanan cloud drive.

Proses pengujian dilakukan pada emulator android dengan spesifikasi seperti berikut:

- Prosesor quadcore (4 cpu)
- Ram 1,5gb
- Resolusi 1080x1920
- Penyimpanan internal 8gb
- Versi android 7.1 (nougat)

1) *Pengujian hasil ciphertext*

Pengujian dilakukan bertujuan untuk mengetahui apakah file yang sudah dienkripsi hasilnya sama atau berbeda dengan algoritma AES standar dan AES modifikasi (Nurrahmi, Setyaningsih, & Herawati, AKPRIND). Berikut adalah hasil enkripsi cipher text menggunakan nilai kunci, salt, dan inisialisasi vektor yang sama namun berbeda putaran. Tabel 1 menampilkan pengujian yang dilakukan pada file berformat txt . Tabel 2 menunjukkan pengujian pada file docx . Sedangkan tabel 3 pengujian dilakukan pada file jpg. Pengujian terakhir dilakukan pada file gambar berformat png ditunjukkan pada tabel 4.

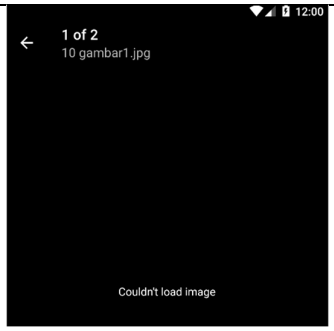
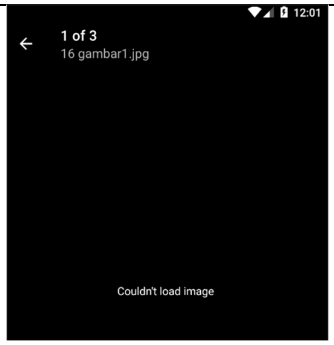
TABEL 1
HASIL CIPHERTEXT .TXT

Plaintext	kata pemulihan.txt donor fantasy must mimic patient supply garment void yard identify skin pupil
Ciphertext 10 putaran	10kata pemulihan.txt
Ciphertext 16 putaran	16kata pemulihan.txt

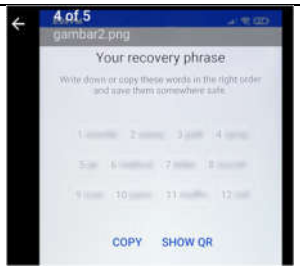
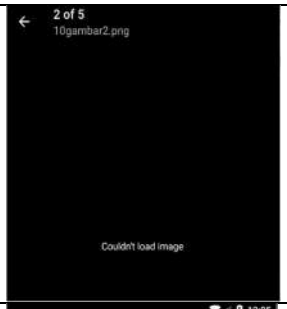
TABEL 2
HASIL CIPHERTEXT .DOCX

Plaintext	backup phrase.docx 1. Phrase wallet 1 hire, swap, soldier, fiction, kingdom, shove, truly, captain, code, search, cart, correct. 2. Phrase wallet 2 blouse, cool, rigid, where, typical, ramp, side, rich, appear, kit, angle, method 3. Phrase wallet 3 build benefit travel master certain shy walnut diet renew donate parent boa 4. Phrase wallet 4 ready, useful, office, click, rabbit, veteran, oxygen, zone, amazing, unusual, rubber, behind.
Ciphertext 10 putaran	backup phrase 10.docx.maes
Ciphertext 16 putaran	backup phrase 16.docx.maes

TABEL 3
HASIL CIPHERTEXT .JPG

Plaintext	
Ciphertext 10 putaran	
Ciphertext 16 putaran	

TABEL 4
HASIL CIPHERTEXT .PNG

Plaintext	
Ciphertext 10 putaran	
Ciphertext 16 putaran	

Hasil pengujian pada keempat tabel yang dilakukan pada 4 file yang berbeda menggunakan *cipher key,* kode *salt*, dan inisialisasi vektor yang sama namun beda putaran, pada dokumen teks seperti txt, dan docx menghasilkan *cipher text* yang berbeda pada tabel 1 dan 2. Kemudian pada tabel 3 dan 4 pengujian pada file gambar berformat jpg & png menunjukkan hasil ciphertext 10 putaran dan 16 putaran memiliki kesamaan keluaran *(output)* yaitu hasil gambar yang sudah dienkripsi tidak bisa dibaca oleh sistem. Hal ini membuktikan bahwa jika peretas ingin membaca file maka harus menebak berapa putaran file yang telah dienkripsi saat awal file enkripsi dibuat.

2) *Hasil Pengujian Waktu*

Proses pengujian waktu untuk mengetahui perbandingan performa pada algoritma enkripsi standar dan modifikasi (Kumaar, P., & Rana, B. S.,2015).

TABEL 5
PENGUJIAN WAKTU

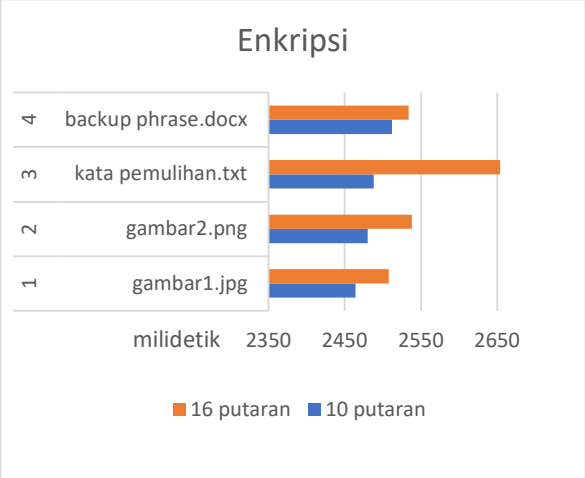
NO	Nama File	Ukuran (mb)	10 putaran		16 putaran	
			E (ms)	D (ms)	E (ms)	D (ms)
1	gambar1.jpg	0,02	2464	2466	2508	2564
2	gambar2.png	0,9	2480	2492	2538	2522
3	kata pemulihan.txt	0,01	2488	2550	2654	2602
4	backup phrase.docx	0,013	2512	2464	2534	2570

Keterangan:
E = Enkripsi D = Dekripsi ms = milidetik

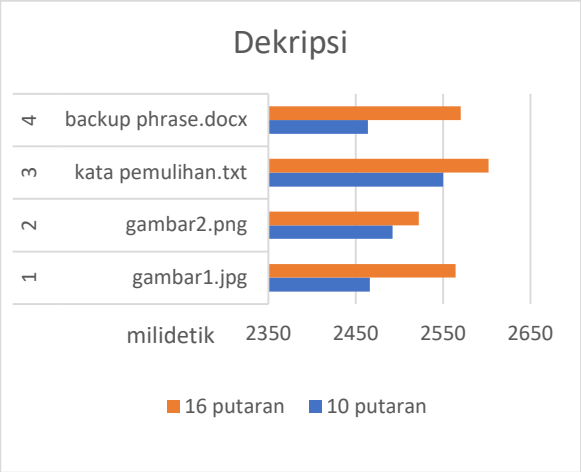
Hasil pengujian dilakukan pada empat file dengan format yang berbeda yaitu jpg, png, txt, dan docx. Pengkajian

dilangsungkan total 5 kali pada masing-masing file dokumen melalui *cipher key* yang sama. Waktu yang didapat saat pengkajian beragam, maka dari itu dibuatlah pengujian sebanyak 5 kali selanjutnya diambil rata-rata waktu yang hasilnya terdapat dalam tabel 5.

Berikut adalah perbandingan antara AES modifikasi yang sudah dibuat melalui AES standart yang sudah tergambar atas grafik dibawah:



Gbr 9. Perbandingan grafik enkripsi



Gbr 10. Perbandingan grafik dekripsi

Gambar 9 dan 10 variabel sebelah kiri yakni sebutan file yang diuji, selanjutnya bagian bawah yakni range nilai dalam milidetik yang dipakai. Hasil perbandingan gambar 9 serta 10 proses enkripsi serta dekripsi memakai algoritma AES modifikasi lebih lama dibanding menggunakan algoritma AES standar (Kumaar, P., & Rana, B. S. 2015). Hal ini menunjukkan melalui penggunaan AES modifikasi membuat peretas semakin sulit untuk membuka file yang telah dienkripsi karena membutuhkan waktu lebih lama lagi.

3) *Hasil Pengujian Avalanche Effect*

Proses pengujian dilakukan ketika enkripsi teks terhadap bit menggunakan *Avalanche Effect. Avalanche Effect* bisa dipakai selaku matrik guna mengkaji keamanan serta kinerja pada algoritma enkripsi. Pengujian berfokus pada analisis yang dilakukan yaitu perubahan kecil bit (satu bit) ASCII (kode teks pada komputer) baik pada *plaintext* maupun *ciphertext* akan menyebabkan perubahana signifikan hasil dari enkripsi *ciphertext* (Sugiyanto & Hapsari Kembang R. 2016). Rumus untuk menghitung *Avalanche Effect* sebagai berikut:

$$Avalanche\ Effect = \frac{Perubahan\ bit}{Total\ bit}\ X\ 100\%$$

Umumnya bit yang berada di *ciphertext* akan mengalamai perubahan dari jumlah bit plaintext. *Avalanche Effect* mempunyai hasil baik bila perubahan bit menghasilkan antara 45-60%. Semakin banyak perubahan bit yang terjadi maka akan semakin sulit dan membutuhkan waktu yang lama algoritma kriptografi untuk dipecahkan (Sugiyanto & Hapsari,2016).

Percobaan 1

Percobaan pertama yaitu mengganti masukan teks *(plaintext)* karakter terakhir yaitu "jtifunesasby2022" menjadi "jtifunesasby2022". Tabel 6 menunjukkan hasil perbedaan *Avalanche Effect* dan jumlah perubahan bit. Algoritma AES standar mempunyai nilai *Avalanche Effect* sebesar 47,6% sedangkan AES modifikasi mempunyai nilai sebesar 57,8%.

TABEL 6
PERUBAHAN KARAKTER PLAINTEXT TERAKHIR

	Algoritma Enkripsi	
	AES	Modifikasi AES
Plaintext	jtifunesasby2022	jtifunesasby2022
Plaintext diedit	jtifunesasby2021	jtifunesasby2021
Kunci	selamatpagi	selamatpagi
Perubahan bit	61	71
Total bit	128	128
Avalanche effect	47,60%	57,80%

Percobaan 2

Percobaan kedua yaitu mengganti kunci (*cipherkey)* pada karakter terakhir yaitu "jayalahunesaku!@" diubah menjadi

"jayalahunesaku!#".Tabel 7 menunjukkan hasil perbedaan nilai *avalanche effect,* nilai algoritma AES standar sebesar 53,9% sedangkan AES modifikasi mempunyai nilai sebesar 58,5%.

TABEL 7
PERUBAHAN KUNCI KARAKTER TERAKHIR

	Algoritma Enkripsi	
	AES	Modifikasi AES
Plaintext	jtifunesasby2022	jtifunesasby2022
Kunci	jayalahunesaku!@	jayalahunesaku!@
Kunci diedit	jayalahunesaku!#	jayalahunesaku!#
Perubahan bit	69	75
Total bit	128	128
Avalanche effect	53,90%	58,50%

Percobaan 3

Percobaan ketiga yaitu mengganti masukan *plaintext* karakter yang berada ditengah yaitu "jtifunesasby2022" menjadi "jtifunesasba2022". Hasil tabel 8 terlihat perbedaan nilai dan jumlah bit pada masing-masing algoritma. AES standar memiliki nilai *avalanche effect* sebesar 44,5% sedangkan AES modifikasi memiliki nilai sebesar 50,7%.

TABEL 8
PERUBAHAN KARAKTER TENGAH PLAINTEXT

	Algoritma Enkripsi	
	AES	Modifikasi AES
Plaintext	jtifunesasby2022	jtifunesasby2022
Plaintext diedit	jtifunesasba2022	jtifunesasba2022
Kunci	jayalahunesaku!@	jayalahunesaku!@
Perubahan bit	57	65
Total bit	128	128
Avalanche effect	44,50%	50,70%

Hasil dari pengujian ketiga tabel mulai dari percobaan 1 sampai 3 diperoleh nilai rata-rata *avalanche effect* dari algoritma AES standar yakni:

$$\frac{47{,}6 + 53{,}9 + 44{,}5}{3} \; X \; 100\% = 48{,}6\%$$

Sedangkan algoritma modifikasi AES didapatkan nilai rata-rata *avalanche effect* sebagai berikut:

$$\frac{57{,}8 + 58{,}5 + 50{,}7}{3} \; X \; 100\% = 55{,}6\%$$

Perhitungan nilai rata-rata *avalanche effect*, didapat nilai dari algoritma modifikasi AES melebihi algoritma AES standar. Bisa diambil simpulan kalau algoritma AES modifikasi lebih baik dalam pengamanan data dari pada AES standar.

IV. KESIMPULAN

Hasil penelitian yang dilakukan yaitu pengujian waktu, *avalanche effect* dan menganalisis hasil enkripsi *chiper text* berbeda putaran, dapat disimpulkan kalau modifikasi algoritma AES pada proses enkripsi serta dekripsi melalui penambahan total putaran dapat mempengaruhi tingkat keamanan sistem yang sudah ada. Berikut adalah kesimpulan yang diperoleh dari hasil penelitian yakni:

1. Modifikasi algoritma AES bisa dilakukan pada platform mobile sesuai dengan referensi yang dibutuhkan.
2. Aplikasi yang dibuat bisa mengubah nilai masukan pada Putaran, Salt dan Inisialisasi Vektor dapat mempengaruhi proses enkripsi dan dekripsi apabila pengguna lupa nilai masukan tersebut maka file tidak dapat didekripsi dengan benar.
3. Modifikasi putaran algoritma AES membutuhkan waktu proses lebih lama daripada AES standar, ini membuktikan bahwa untuk memecahkan file terenkripsi membutuhkan waktu lebih lama bagi *hacker* dan kripto analis.
4. Pengujian *avalanche effect* algoritma modifikasi AES memiliki nilai sebesar 55,6% sedangkan algoritma AES standar memiliki nilai sebesar 48,6%. Hal ini membuktikan modifikasi AES lebih baik dalam mengamankan data dari pada AES standar.

V. SARAN

Aplikasi ini dapat dikembangkan lebih lanjut lagi, misal dengan memperbaiki beberapa bug yang masih ada terkendala pada dukungan versi android terbaru dimana tidak bisa membaca masukan file yang akan dienkripsi, selanjutnya dengan mengembangkan aplikasi ini pada sistem operasi IOS agar bisa menjangkau lebih banyak lagi pengguna *mobile* yang ingin mengamankan asset digital *cryptocurrency* mereka agar lebih aman dari pencurian.

UCAPAN TERIMAKASIH

Ucapan syukur serta terimakasih penulis disampaikan kepada:

1. Allah SWT atas segala rahmat serta hidayah, sehinggan penulis bisa menuntaskan penelitian ini.

2. Bapak Jamiin dan Ibu Mukholifah selaku orang tua saya yang selalu mendukung serta menyemangati.
3. Bapak Dr. Ricky Eka Putra, S.kom., M.Kom. selaku dosen pembimbing dengan sabar membimbing penelitian ini dari awal hingga akhir.
4. Seluruh teman-teman prodi TI angkatan 2018 yang selalu mendukung serta menyemangati.

DAFTAR PUSTAKA

[1] Kumaar, P., & Rana, B. S. (2015). Development of Modified AES Algorithm for Data Security. Jurnal Optik, 2341-2345.

[2] Sugiyanto & Hapsari Kembang R. (2016). Pengembangan Algoritma Advanced Encryption Standart pada sistem keamanan berbasis android menggunakan algoritma Vigenere. Jurnal ultimatics, Institut Teknologi Adhi Tama Surabaya.

[3] Wicaksono Sakti Yudo, 2018. Keabsahan Transaksi menggunakan system cryptocurrency di Indonesia. Fakultas Hukum Universitas 17 Agustus 1945 Surabaya.

[4] Barcroft Joe, 2012. Mnemonics. Wiley Online Library. https://onlinelibrary.wiley.com/doi/abs/10.1002/9781405198431.wbeal0767 (diakses 30 januari 2022).

[5] Pabokory N. F. dkk. 2015. Implementasi Kriptografi Pengamanan Data pada pesan Teks, isi file Dokumen, dan File dokumen Menggunakan Algoritma Advanced Encryption Standart. FMIPA, Universitas Mulawarman Vol. 10, No. 1.

[6] Telegarapu P. dkk. 2011. Design and Analysis of Multimedia Communication System. MIT, Anna University, Chennai.

[7] Talbot John & Welsh Dominic. 2006. Complexity and cryptography: an introduction. Cambridge University Press, New York.

[8] Daemen, J., & Rijmen, V. (n.d.). Computer Security Resource Center. Retrieved from NIST: https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-andguidelines/documents/aes-development/rijndael-ammended.pdf

[9] Rahmatullah A. dkk. 2016. Kriptografi Advanced Encryption Standard (AES) Untuk Penyandian File Dokumen. FMIPA Universitas Islam Bandung.

[10] Krisanty Tania. Studi Mengenai Salt. Prodi Teknik Informatika, Institut Teknologi Bandung.

[11] Nurrahmi, Setyaningsih, & Herawati. Keamanan File dokumen menggunakan algoritma advanced encryption standart pada aplikasi berbasis android. Institute Sains & Teknologi AKPRIND, Yogyakarta.

[12] Galas M. E. & Gerardo B. D. 2019. Implementing Randomized Salt on Round Key for Corrected Block Tiny Encryption Algorithm (XXTEA). IEEE 11th International Conference on Communication Software and Networks (diakses 10 mei 2022).

[13] Pirzada dkk 2020. Modification of Initialization Vector for Parallel CMAC Algorithm. School of Cyber Science and Technology Beihang University Beijing, China.

[14] Abikoye dkk. 2019. Modified Advanced Encryption Standart Algorithm for Information Security. MDPI journal: Volume 11, Issue 12.

[15] Golodetz dkk. 2014. Two Tree-based Methods for The Waterfall. Department of Computer Science, University of Oxford, United Kingdom.