

PENYANDIAN DATA DENGAN ALGORITMA KRIPTOGRAFI NOEKEON

I Made Ari Jaya N (232 02 002)

Magister Teknologi Informasi
Institut Teknologi Bandung

ABSTRAKSI

Keamanan data merupakan hal yang sangat penting dalam menjaga kerahasiaan informasi terutama yang berisi informasi sensitif yang hanya boleh diketahui isinya oleh pihak yang berhak saja, apalagi jika pengirimannya dilakukan melalui jaringan publik, apabila data tersebut tidak diamankan terlebih dahulu, akan sangat mudah disadap dan diketahui isi informasinya oleh pihak-pihak yang tidak berhak.

Salah satu cara yang digunakan untuk pengamanan data adalah menggunakan sistem kriptografi yaitu dengan menyandikan isi informasi (plaintext) tersebut menjadi isi yang tidak dipahami melalui proses enkripsi (encipher), dan untuk memperoleh kembali informasi yang asli, dilakukan proses dekripsi (decipher), disertai dengan menggunakan kunci yang benar. Namun sejalan dengan perkembangan ilmu penyandian atau kriptografi, usaha-usaha untuk memperoleh kunci tersebut dapat dilakukan oleh siapa saja, termasuk pihak yang tidak sah untuk memiliki informasi tersebut. Oleh karena itu, penelitian tentang kriptografi akan selalu berkembang untuk memperoleh algoritma kriptografi yang makin kuat, sehingga usaha-usaha untuk memecah kode kriptografi secara tidak sah menjadi lebih sulit.

Melalui tugas ini akan dibahas mengenai algoritma kriptografi NOEKEON, yang diajukan kepada proyek Nessie (New European Schemes for Signatures, Integrity, and Encryption) dalam rangka memperkuat posisi industri negara-negara Eropa di bidang kriptografi.

Kata Kunci: Kriptografi, *Block Cipher*, Noekeon.

I Dasar Teori

I.1 Pengertian Kriptografi

Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti keabsahan, integritas data, serta autentikasi data. Kriptografi tidak berarti hanya memberikan keamanan informasi saja, namun lebih ke arah teknik-tekniknya.

Ada empat tujuan mendasar dari ilmu kriptografi ini yaitu :

1. Kerahasiaan, adalah layanan yang digunakan untuk menjaga isi dari informasi dari siapapun kecuali yang memiliki otoritas.
2. Integritas data, adalah berhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.
3. Autentikasi, adalah berhubungan dengan identifikasi, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan melalui kanal harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain.
4. Non-repudiasi, yang berarti begitu pesan terkirim, maka tidak akan dapat dibatalkan.

I.2 Dasar Matematis

Dasar matematis yang mendasari proses enkripsi dan dekripsi adalah relasi antara dua himpunan yaitu yang berisi elemen *plaintext* dan yang berisi elemen *ciphertext*. Enkripsi dan dekripsi merupakan fungsi transformasi antara himpunan-himpunan tersebut. Apabila elemen-elemen *plaintext* dinotasikan dengan P , elemen-elemen *ciphertext* dinotasikan dengan C , sedang untuk proses enkripsi dinotasikan dengan E , dekripsi dengan notasi D , maka secara matematis proses kriptografi dapat dinyatakan sebagai berikut :

$$\text{Enkripsi : } E(P) = C \quad (1.1)$$

$$\text{Dekripsi : } D(C) = P \text{ atau } D(E(P)) = P \quad (1.2)$$

Pada skema enkripsi konvensional atau *symmetric-key*, digunakan sebuah kunci untuk melakukan proses enkripsi dan dekripsinya. Kunci tersebut dinotasikan dengan K . Sehingga proses kriptografinya adalah sebagai berikut:

$$\text{Enkripsi : } E_K(P) = C \quad (1.3)$$

$$\text{Dekripsi : } D_K(C) = P \text{ atau } D_K(E_K(P)) = P \quad (1.4)$$

Sedangkan pada sistem *asymmetric-key* digunakan kunci umum (*public key*) untuk enkripsi dan kunci pribadi (*private key*) untuk proses dekripsinya. Sehingga kedua proses tersebut dinyatakan dengan :

$$\text{Enkripsi : } E_{PK}(P) = C \quad (1.5)$$

$$\text{Dekripsi : } D_{SK}(C) = P \text{ atau } D_{SK}(E_{PK}(P)) = P \quad (1.6)$$

I.3 Teknik Kriptografi

Secara umum dikenal dua teknik dalam kriptografi, yaitu *symmetric-key* dan *asymmetric-key* (*public-key*).

II.3.1 *Symmetric-key*

Skema enkripsi akan disebut *symmetric-key* apabila pasangan kunci untuk proses enkripsi dan dekripsinya adalah sama. Pada skema enkripsi *symmetric-key* dibedakan menjadi dua kelas, yaitu *block-cipher* dan *stream-cipher*.

Block-cipher adalah skema enkripsi yang akan membagi-bagi *plaintext* yang akan dikirimkan menjadi string-string (disebut blok) dengan panjang t , dan mengenkripsinya per-blok. Pada umumnya, *block-cipher* memproses *plaintext* dengan blok yang relatif panjang lebih dari 64 bit, untuk mempersulit penggunaan pola-pola serangan yang ada untuk membongkar kunci. Sedangkan skema *stream-cipher* pada dasarnya juga *block-cipher*, hanya dengan panjang bloknnya adalah satu bit.

I.3.2 *Asymmetric-key*

Skema ini adalah algoritma yang menggunakan kunci yang berbeda untuk proses enkripsi dan dekripsinya. Skema ini disebut juga sebagai sistem kriptografi *Public-key* karena kunci untuk enkripsi dibuat secara umum (*public-key*) atau dapat diketahui siapa saja, tapi untuk proses dekripsinya dibuat satu hanya oleh yang berwenang untuk mendekripsinya, disebut *private-key*.

Keuntungan skema model ini, untuk berkorespondensi secara rahasia dengan banyak pihak tidak diperlukan kunci rahasia sebanyak jumlah pihak tersebut, cukup membuat dua buah kunci, yaitu *public-key* bagi para koresponden untuk mengenkripsi pesan, dan *private-key* untuk mendekripsi pesan. Berbeda dengan skema *symmetric-key*, jumlah kunci yang dibuat adalah sebanyak jumlah pihak yang diajak berkorespondensi.

I.4 Kriptografi blok cipher

I.4.1 Konsep Dasar

Blok cipher merupakan sebuah fungsi yang memetakan n -bit blok-blok *plaintext* ke n -bit blok-blok *ciphertext*, dengan n adalah panjang blok. Blok cipher umumnya memproses *plaintext* ke dalam blok-blok yang cukup besar ($n \geq 64$). Beberapa teknik blok cipher modern diantaranya :

I.4.1.1 Cipher Berulang

Pada teknik cipher berulang (*iterated cipher*), blok *plaintext* mengalami pengulangan fungsi transformasi beberapa kali untuk mendapatkan blok *ciphertext*. Fungsi transformasi pada umumnya merupakan gabungan proses substitusi, permutasi, kompresi, atau ekspansi terhadap blok *plaintext*. Sebuah kunci pada setiap putaran (*round key*) akan dikombinasikan dengan *plaintext*. Parameter dalam cipher ini adalah jumlah putaran r , besar blok n , dan besar kunci k . Sub-kunci K_i pada setiap putaran diperoleh dari penurunan kunci input K .

I.4.1.2 Fiestel Cipher

Fiestel cipher beroperasi terhadap panjang blok data tetap sepanjang n (genap), kemudian membagi 2 blok tersebut dengan panjang masing-masing $n/2$, yang dinotasikan dengan L dan R . Fiestel cipher menerapkan metode cipher berulang dengan masukan pada putaran ke- i yang didapat dari keluaran sebelumnya. Secara matematis dapat dinyatakan sebagai berikut:

$$L_i = R_{i-1} \quad (1.7)$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i); \quad i = 1, 2, 3, \dots, r \quad (1.8)$$

K_i adalah kunci untuk putaran ke- i dan f adalah fungsi transformasi.

Blok *plaintext* adalah gabungan L dan R awal atau secara formal *plaintext* dinyatakan dengan (L_0, R_0) . Sedangkan blok *ciphertext* didapatkan dari L dan R hasil putaran terakhir setelah terlebih dahulu dipertukarkan atau secara formal *ciphertext* dinyatakan dengan (R_r, L_r) .

I.4.1.3 Avalanche

Pada blok cipher, perubahan satu buah bit dapat menghasilkan perubahan lebih dari satu bit setelah satu putaran, lebih banyak lagi bit berubah untuk putaran berikutnya. Hasil perubahan tersebut dinamakan sebagai *avalanche effect*. Sebuah algoritma kriptografi memenuhi kriteria *avalanche effect* apabila satu buah bit input mengalami perubahan, maka probabilitas semua bit berubah adalah setengahnya. *Avalanche effect* merupakan salah satu karakteristik yang menjadi acuan untuk menentukan baik atau tidaknya sebuah algoritma kriptografi.

I.4.2 Mode Operasi

Ada beberapa mode operasi yang digunakan dalam kriptografi. Beberapa diantaranya adalah :

1. Electronic codebook (ECB)

Pada mode ini, blok-blok *plaintext* (x) yang identik (menggunakan kunci yang sama) akan menghasilkan *ciphertext* (c) yang identik pula. Secara matematis dinyatakan :

$$\text{Enkripsi : } c_j \leftarrow E_K(x_j); \quad 1 \leq j \leq t \quad (1.9)$$

$$\text{Dekripsi : } x_j \leftarrow E_K^{-1}(c_j); \quad 1 \leq j \leq t \quad (1.10)$$

2. Cipher-block chaining (CBC)

Pada prosesnya, mode ini melibatkan penggunaan *initializing vector* (IV), yang menyebabkan blok-blok *ciphertext* yang identik apabila dienkripsi menggunakan kunci dan IV yang sama. Berubahnya IV , kunci, atau blok *plaintext* pertama akan menghasilkan *ciphertext* yang berbeda. Secara matematis dinyatakan :

$$\text{Enkripsi : } c_0 \leftarrow IV, \text{ untuk } 1 \leq j \leq t, c_j \leftarrow E_K(c_{j-1} \oplus x_j). \quad (1.11)$$

$$\text{Dekripsi : } c_0 \leftarrow IV, \text{ untuk } 1 \leq j \leq t, x_j \leftarrow c_j - 1 \oplus E_K^{-1}(c_j). \quad (1.12)$$

3. Cipher feedback (CFB)

Jika pada mode CBC, *plaintext* sebesar n -bit diproses dalam sekali waktu (menggunakan sebuah n -bit cipher blok). Beberapa aplikasi mengharuskan r -bit *plaintext* untuk dienkripsi terlebih dahulu dan ditransmisikan bebas delay, untuk $r < n$ (biasanya $r = 1$ atau $r = 8$). Dalam kasus ini CFB digunakan. Dalam mode ini juga melibatkan penggunaan *initializing vector*.

4. Output feedback (OFB)

Mode operasi ini digunakan apabila kesalahan propagasi sama sekali harus dihindari. Hampir mirip dengan CFB, dan juga memungkinkan enkripsi menggunakan besar blok yang bervariasi.

I.4.3 Kunci Lemah dan Kunci Setengah Lemah

Dalam kriptografi dikenal dengan istilah kunci lemah (*weak-key*) dan kunci setengah lemah (*semi weak-key*). Kunci Lemah adalah kunci yang apabila mengenkripsi suatu *plaintext* kemudian dienkripsi lagi menggunakan kunci yang sama maka *ciphertext*-nya adalah *plaintext* itu sendiri. Sedangkan yang disebut kunci setengah lemah adalah sepasang kunci yang mempunyai sifat jika sebuah *plaintext* dienkripsi dengan suatu kunci, akan dapat didekripsi dengan kunci yang lain.

I.5 Enkripsi dan Dekripsi

I.5.1 Enkripsi

Proses utama dalam suatu algoritma kriptografi adalah enkripsi dan dekripsi. Enkripsi merubah sebuah *plaintext* ke dalam bentuk *ciphertext*. Pada mode ECB (*Electronic Codebook*), sebuah blok pada *plaintext* dienkripsi kedalam sebuah blok *ciphertext* dengan panjang blok yang sama. Secara matematis, proses enkripsi telah diberikan pada persamaan (1.3).

Blok cipher memiliki sifat bahwa setiap blok harus memiliki panjang yang sama (misal 128 bit). Namun apabila pesan yang dienkripsi memiliki panjang blok terakhir yang tidak tepat 128 bit, maka diperlukan mekanisme *padding* yaitu penambahan bit-bit *dummies* untuk menggenapi menjadi panjang blok yang sesuai, biasanya *padding* dilakukan pada blok terakhir *plaintext*.

I.5.2 Padding

Padding pada blok terakhir bisa dilakukan dengan berbagai macam cara, misal penambahan bit-bit tertentu. Salah satu contoh penerapan *padding* dengan cara menambahkan jumlah total *padding* sebagai byte terakhir pada blok terakhir *plaintext*. Misal panjang blok adalah 128 bit (16 byte), dan pada blok terakhir terdiri dari 88 bit (11 byte). Sehingga jumlah *padding* yang diperlukan adalah 5 byte, yaitu dengan menambahkan angka nol sebanyak 4 byte, kemudian

menambahkan angka 5 sebanyak satu byte. Cara lain dapat menggunakan penambahan karakter *end-of-file* pada byte terakhir lalu diberi *padding* setelahnya.

I.5.3 Dekripsi

Dekripsi merupakan proses kebalikan dari proses enkripsi, merubah *ciphertext* kembali kedalam bentuk *plaintext*. Proses dekripsi telah diberikan pada persamaan (1.4). Untuk menghilangkan *padding* yang diberikan saat proses enkripsi, dilakukan berdasarkan informasi jumlah *padding* yaitu angka pada byte terakhir.

I.6 Algoritma Kriptografi Noekeon

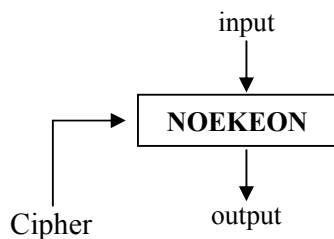
NOEKEON merupakan cipher blok berulang dengan panjang blok dan panjang kuncinya masing-masing 128 bit, yang terdiri dari aplikasi transformasi *round* sederhana yang berulang, diikuti dengan sebuah transformasi output.

NOEKEON memiliki 16 putaran (N_r) iterasi, dalam setiap putarannya dilakukan empat buah transformasi yaitu *theta*, *Shift offset* yang terdiri dari dua buah transformasi P_{i1} dan P_{i2} , dan *gamma*.

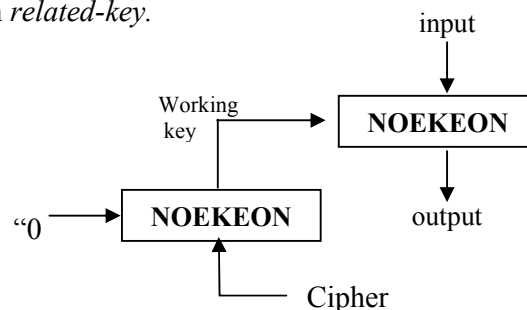
I.6.1 Penjadwalan Kunci

Penjadwalan kunci dilakukan dengan mengkonversi kunci utama (*cipher key*) 128-bit menjadi sebuah *working-key* 128-bit. Karena sifat NOEKEON yang simetri, maka setiap *round*-nya, menggunakan *working-key* yang sama.

Dalam Noekeon, ada mode saat penjadwalan kunci tidak dilakukan, disebut mode *direct-key*, artinya *working-key* adalah *cipher-key* itu sendiri. Mode yang kedua adalah mode *indirect-key* yang melakukan proses penjadwalan kunci untuk mengeliminasi pola serangan *related-key*.



Gambar 2.1 Mode Direct-key
[4]



Gambar 2.2 Mode indirect-key
[4]

Pada mode *indirect-key*, sebelum kunci diaplikasikan terhadap pesan pada operasi *theta*, kunci dirubah dahulu menjadi sebuah kunci yang lain dengan tetap menggunakan fungsi yang sama dalam Noekeon. Baru kemudian, kunci tersebut diaplikasikan terhadap pesan pada operasi *theta* dan seterusnya sebanyak 16 putaran.

I.6.2 State

Setiap transformasi *round* dioperasikan pada sebuah state yang terdiri dari empat buah 32-bit word yaitu $a[0]$ sampai $a[3]$.

I.6.3 Theta

Theta adalah pemetaan linear yang menggunakan *working-key* k dan dilakukan operasi pada *state* a . Pada tahap ini, terdiri dari 12 langkah operasi. Langkah yang pertama adalah operasi *xor* antara word a_0 dan a_2 . Selanjutnya, hasil operasi itu dilakukan dua buah pergeseran bit, yaitu kekanan sebanyak 8 bit, dan kekiri sebanyak 8 bit, lalu hasil pergeseran tersebut di-*xor* dengan hasil langkah pertama. Langkah berikutnya, yaitu proses perubahan word a_1 dan a_3 , dengan meng-*xor*-kan a_1 dengan langkah kedua. Setelah itu, keempat word dari *plaintext* masing-masing di-*xor*-kan dengan keempat buah word kunci, dan akan menghasilkan word $[a_0, a_1, a_2, a_3]$ baru. Dari word baru tersebut, a_1 dan a_3 , di-*xor*, dan hasilnya dilakukan dua buah pergeseran 8 bit masing-masing kekanan dan kekiri. Terakhir, a_0 dan a_2 masing-masing akan di-*xor* dengan hasil pergeseran tersebut. Dari proses *Theta* ini akan dihasilkan word $[a_0, a_1, a_2, a_3]$ yang baru, untuk dilakukan proses selanjutnya.

I.6.4 Shift Offset

Pergeseran ini terdiri dari dua yaitu Pi_1 dan Pi_2 yang keduanya saling berkebalikan. Masing-masing terdiri dari empat offset dengan modulo 8 yang berbeda (a_0, a_1, a_2, a_3), untuk a_0 tidak dilakukan pergeseran. Secara jelas digambarkan sebagai berikut:



Gambar 2.3 Visualisasi Shift Offset

Pada gambar diatas, susunan empat buah word mulai teratas adalah a_3 , kemudian a_2, a_1 , dan a_0 . Pada proses Pi_1 , a_1 digeser 1 bit ke kiri, a_2 5 bit ke kiri, dan a_3 2 bit ke kiri. Pada proses Pi_2 , merupakan kebalikan Pi_1 , jadi a_1 digeser 1 bit ke kanan, a_2 5 bit ke kanan, dan a_3 2 bit ke kanan. Sedang a_0 tidak terjadi pergeseran.

I.6.5 Gamma

Gamma merupakan pemetaan involusi non-linear. Transformasi non-linear dilakukan dengan tiga langkah yaitu:

- Transformasi non-linear sederhana
- Transformasi linear sederhana
- Transformasi non-linear sederhana

Pada tahap ini, Noekeon akan menghasilkan word-word a_0 , a_1 , dan a_2 yang baru. *Gamma* akan menghasilkan *substitution-box (S-box)* bagi Noekeon. S-box ini berupa word terdiri dari 4 buah word 32 bit yang masing-masing box-nya adalah 4 bit setiap word $[a_0, a_1, a_2, a_3]$ secara berurut.

I.6.6 Round Constant

Untuk menghilangkan sifat kelinearan pada setiap putaran Noekeon, dilakukan operasi *round constants*. Pada dasarnya operasi ini adalah sebuah shift register (mod $0x80$, untuk $state[0]$) yang dilakukan terhadap 8 bit terbawah dalam 32-bit word *state* awal.

I.6.7 Enkripsi dan Dekripsi

I.6.7.1 Tahap Enkripsi

Tahap ini diawali dengan adanya masukan dari pemakai berupa teks yaitu untuk *plaintext* dan untuk kuncinya. Teks yang masih berbentuk karakter tersebut selanjutnya akan direpresentasikan kedalam bentuk deretan bit, kemudian akan dibentuk blok dengan panjang 128 bit, yang masing-masing blok untuk *plaintext* dan kuncinya akan dibagi menjadi 4 buah word masing-masing 32 bit, yaitu $[a_0, a_1, a_2, a_3]$ untuk *plaintext* dan $[k_0, k_1, k_2, k_3]$ untuk kuncinya. Bila jumlah bit dalam satu blok tersebut kurang dari 128 bit, maka akan dikenakan dengan menambahkan bit "0". Kedelapan word tersebut yang nantinya akan diproses dalam prosedur algoritma Noekeon.

I.6.7.2 Tahap Dekripsi

Keunggulan algoritma Noekeon terletak pada kesederhanaan kode program atau sirkuit perangkat kerasnya. Kode atau sirkuit yang sama, digunakan baik pada proses enkripsi dan dekripsinya, hanya penerapan pada *theta* yang berbeda. Pada proses enkripsi, *theta* adalah *theta* (k , a), sedangkan pada proses dekripsinya, menjadi *theta* (NullVektor, a). Dengan kata lain, kebalikan dari *theta* adalah *theta* itu sendiri, namun dengan pengaplikasian *null vektor* sebagai *working-key*.

II PERANCANGAN SIMULASI ALGORITMA KRIPTOGRAFI NOEKEON

II.1 Analisa Sistem

Pada simulasi algoritma kriptografi Noekeon terdiri dari dua tahapan besar, yaitu tahap enkripsi dan tahap dekripsi, yang masing-masing, satu putarannya (*round*) terdiri dari empat prosedur, yaitu prosedur *theta*, prosedur *Pi1*, prosedur *gamma*, dan prosedur *Pi2*. Pada implementasinya, data sebagai masukan, akan dibagi-bagi menjadi beberapa blok yang masing-masing blok sepanjang 128 bit yang disebut *state*, baik sebagai *plaintext* maupun sebagai

kunci, kemudian blok-blok tersebut akan dilakukan operasi dengan empat prosedur diatas tadi sehingga akan mneghasilkan *ciphertext* yang diharapkan.

Tujuan utama pada simulasi ini adalah memberikan penjelasan yang cukup baik dan mudah dipahami bagaimana proses atau langkah kriptografi algoritma Noekeon, mulai dari masukan data yang diberikan oleh pengguna, sampai terjadinya perubahan data tersebut menjadi bentuk yang telah terenkripsi, kemudian kembali lagi menjadi data semula.

II.2 Batasan Perancangan Program Simulasi

Dalam proses perancangan program simulasi, akan ada pembatasan-pembatasan masalah yang diutamakan untuk memudahkan pengamatan langkah-langkah kriptografi pada algoritma ini, yaitu :

1. Simulasi dibuat dengan menggunakan bahasa pemrograman Borland C++ Builder 6.
2. Masukan dari pengguna berupa string, baik untuk kunci kriptografi maupun teks yang aka dienkripsi.
3. Jumlah blok yang akan diamati yaitu satu buah blok, mengingat masukan berupa string juga memiliki panjang yang terbatas.

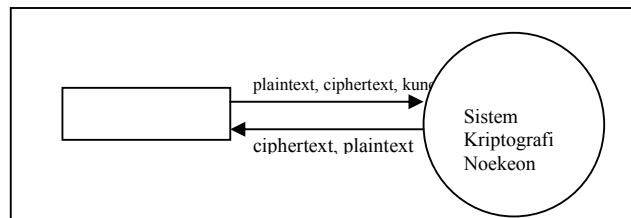
II.3 Perancangan Sistem

II.3.1 Model Simulasi

Model simulasi yang akan dirancang adalah dengan adanya masukan dari pengguna, kemudian visualisasi proses sistem, dan keluaran yang ditampilkan bagi pengguna. Oleh karena itu, akan dibuat dalam bentuk perangkat lunak yang sederhana, namun meliputi seluruh proses dalam enkripsi dan dekripsinya. Dalam menjelaskan tentang sistem yang dirancang, diberikan alur data sistem dalam bentuk diagram konteks dan Diagram Aliran Data, serta akan lebih dirinci ke dalam level diagram berikutnya.

II.3.1.1 Diagram Konteks

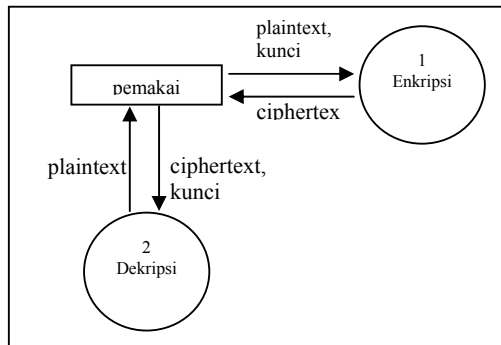
Perancangan dimulai dengan pembuatan diagram konteks, berupa penggambaran sistem penerapan algoritma Noekeon secara garis besar. Berikut ini diagram konteks sistem simulasi yang akan dirancang :



Gambar 2.1 Diagram Konteks

Dari diagram konteks tersebut, diturunkan Diagram Aliran Data (DAD) level 0 untuk penjabaran sistem yang terdiri dari :

- Proses dekripsi
- Proses enkripsi



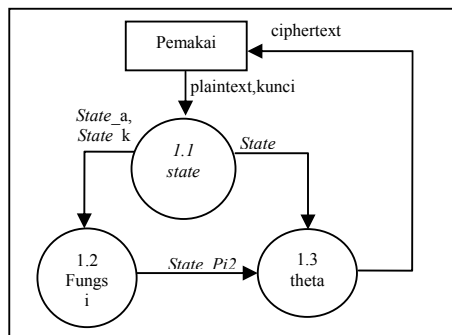
Gambar 2.2 Diagram Aliran

Untuk kedua proses enkripsi dan dekripsi yang masing-masing putarannya terdiri dari θ , $Pi1$, γ , dan $Pi2$, dapat lebih dirinci nantinya dalam Diagram Aliran Data yang diturunkan dari DAD level 0.

II.3.1.2 Diagram Aliran Data

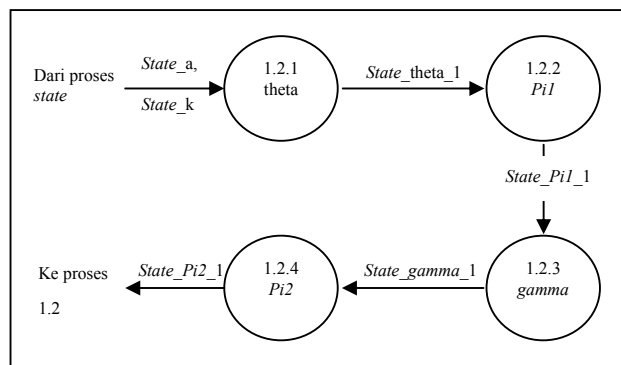
II.3.1.2.1 Proses Enkripsi

Dari DAD level 0 seperti gambar 2.2 sebelumnya, dapat diturunkan menjadi DAD level 1 proses 1 yang merincikan proses dekripsi pada algoritma Nokeon .



Gambar 2.3 DAD level 1 proses 1

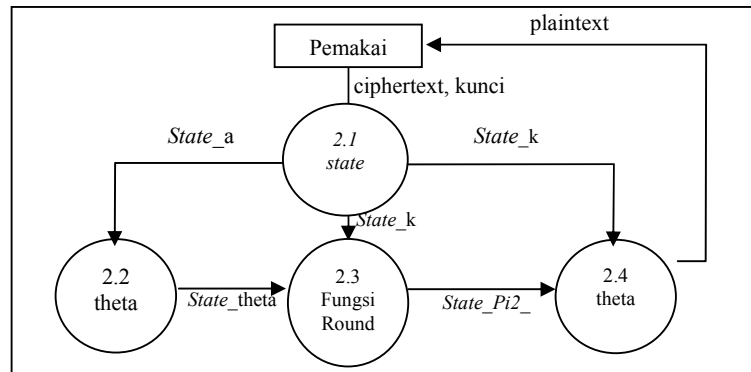
Untuk menjelaskan fungsi *Round*, diturunkan dari DAD level 1 ke DAD level 2 sebagai berikut :



Gambar 2.4 DAD level 2 proses 1.2

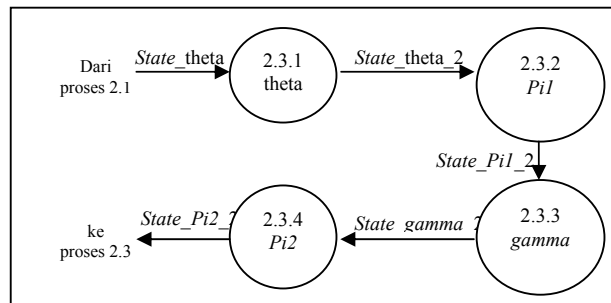
II.3.1.2.2 Proses Dekripsi

Dari DAD level 0 diatas dapat diturunkan kembali untuk menjelaskan proses dekripsi menjadi DAD level 1 proses 2 berikut ini :



Gambar 2.5 DAD level 1 proses 2

Fungsi *Round* yang terjadi pada proses dekripsi sama dengan fungsi *Round* yang terjadi pada proses enkripsi.



Gambar 2.6 DAD level 2

II.3.1.3 Kamus Data

- Plaintext merupakan masukan dan keluaran dari proses sistem.
Plaintext = 0 {karakter ASCII} 16
- Ciphertext merupakan masukan dan keluaran dari proses sistem.
Ciphertext = 0 {karakter ASCII} 16
- State_a merupakan perubahan bentuk masukan (plaintext atau ciphertext) ke dalam bentuk state yaitu 4 buah 32 bit word.
State_a = 0 {karakter ASCII} 16
- State_k merupakan perubahan bentuk kunci ke dalam bentuk state yaitu 4 buah 32 bit word.
State_k = 0 {karakter ASCII} 16
- State_theta_1 merupakan state setelah hasil theta pada proses enkripsi.
State_theta_1 = 0 {karakter ASCII} 16
- State_theta_2 merupakan state setelah hasil theta pada proses dekripsi.
State_theta_2 = 0 {karakter ASCII} 16
- State_Pi1_1 merupakan state setelah hasil Pi1 pada proses enkripsi.
State_Pi1_1 = 0 {karakter ASCII} 16

- State_Pi1_2 merupakan state setelah hasil Pi2 pada proses dekripsi.
State_Pi1_2 = 0 {karakter ASCII} 16
- State_gamma_1 merupakan state setelah hasil Pi1 pada proses enkripsi.
State_gamma_1 = 0 {karakter ASCII} 16
- State_gamma_2 merupakan state setelah hasil Pi1 pada proses dekripsi.
State_gamma_2 = 0 {karakter ASCII} 16
- State_Pi2_1 merupakan state setelah hasil Pi2 pada proses enkripsi.
State_Pi2_1 = 0 {karakter ASCII} 16
- State_Pi2_2 merupakan state setelah hasil Pi2 pada proses dekripsi.
State_Pi2_2 = 0 {karakter ASCII} 16

II.3.1.4 Spesifikasi Proses

Penggunaan algoritma Noekeon pada simulasi sistem kriptografi seperti telah disebutkan, menerapkan beberapa proses yang saling berhubungan sehingga akan membentuk suatu sistem utuh yang diharapkan.

Nomor Proses	1.1, 2.1
Masukan	Teks dari pengguna. Berupa plaintext, ciphertext, atau kunci
Keluaran	State_a, state_k
Logika	128 bit masukan data, dibagi kedalam empat buah word, masing-masing 32 bit.

Nomor Proses	1.2, 2.3
Masukan	State_a, state_k, state_theta
Keluaran	State_Pi2_1, State_Pi2_2
Logika	Theta (k, a) Pi1 (a) Gamma (a) Pi2 (a)

Nomor Proses	1.3, 1.2.1, 2.2, 2.4
Masukan	State_Pi2_1, State_a, State_k, State_a, State_Pi2_2
Keluaran	ciphertext, State_theta_1, State_theta, plaintext
Logika	$\begin{aligned} & \text{temp} = a0 \oplus a2 \\ & \text{temp} = \text{temp} \oplus (\text{temp} \lll 8) \oplus (\text{temp} \ggg 8) \\ & a1 = a1 \oplus \text{temp} \\ & a3 = a3 \oplus \text{temp} \\ & a0 = a0 \oplus k0 \\ & a1 = a1 \oplus k1 \\ & a2 = a2 \oplus k2 \\ & a3 = a3 \oplus k3 \\ & \text{temp} = a1 \oplus a3 \\ & \text{temp} = \text{temp} \oplus (\text{temp} \lll 8) \oplus (\text{temp} \ggg 8) \\ & a0 = a0 \oplus \text{temp} \\ & a2 = a2 \oplus \text{temp} \end{aligned}$

Nomor Proses	1.2.2, 2.3.2
Masukan	State_theta_1, State_theta_2
Keluaran	State_Pi1_1, State_Pi1_2
Logika	$\begin{aligned} a1 &= a1 \lll 1 \\ a2 &= a2 \lll 5 \\ a3 &= a3 \lll 2 \end{aligned}$

Nomor Proses	1.2.3, 2.3.3
Masukan	State_Pi1_1, State_Pi1_2
Keluaran	State_gamma_1, State_gamma_2
Logika	$a1 = a1 \oplus \neg (a3 \vee a2)$ $a0 = a0 \oplus (a2 \wedge a1)$ $temp = a3$ $a3 = a0$ $a0 = temp$ $a2 = a2 \oplus a0 \oplus a1 \oplus a3$ $a1 = a1 \oplus \neg (a3 \vee a2)$ $a0 = a0 \oplus (a2 \wedge a1)$

Nomor Proses	1.2.4, 2.3.4
Masukan	State_gamma_1, State_gamma_2
Keluaran	State_Pi2_1, State_Pi2_2
Logika	$a1 = a1 \ggg 1$ $a2 = a2 \ggg 5$ $a3 = a3 \ggg 2$

II.4 Implementasi Sistem

Implementasi perangkat lunak simulasi ini dilakukan dengan membuat antarmuka yang sederhana dalam mempermudah pemberian masukan serta memberikan visualisasi yang baik terhadap proses sistem yang terjadi. Antarmuka ini terdiri atas kontrol masukan dan keluaran, menu pilihan, serta kontrol visualisasi.

III HASIL PERANCANGAN DAN ANALISA UNJUK KERJA ALGORITMA NOEKEON

Tampilan muka untuk menu hasil perancangan simulasi dapat dilihat pada “*pogram simulasi*” yang disertakan pada laporan ini (MADE_ARI.exe)

III.1 Hasil Simulasi

Untuk memulai simulasi, terdapat pilihan proses yaitu enkripsi atau dekripsi, serta masukan berupa string dan kunci yang dipakai. Untuk melihat hasil simulasi maka dapat dilakukan dengan menjalankan program.

III.1.1 Proses Enkripsi

Pada proses ini dibatasi masukan berupa 16 buah karakter bertipe string, karena akan lebih mudah mengamati perubahannya saat dienkripsi. Kunci yang dipakai dibatasi 16 karakter. Proses enkripsi dan langkah-langkahnya dapat dilihat langsung pada program.

III.1.2 Proses Dekripsi

Proses dekripsi dilakukan guna memperoleh pesan yang dienkripsi, sehingga *ciphertext* yang dihasilkan dapat terbaca kembali. Sebagai masukan adalah *ciphertext*, sedangkan kunci yang digunakan tetap sama dengan kunci yang digunakan pada proses enkripsi. Proses dekripsi dapat dilihat langsung pada program.

III.2 Proses File

Pada submenu file, berfungsi untuk dekripsi dan enkripsi arsip. Pada sub menu ini tidak diperlihatkan proses kerja algoritma, hanya menunjukkan kemampuan algoritma Noekeon dalam mengenkripsi dan mendekripsi suatu file.

III.3 Analisa Unjuk Kerja Algoritma Noekeon

III.3.1 Struktur Cipher

Struktur *cipher* pada algoritma Noekeon ditekankan pada kesederhanaan transformasi, yaitu komposisi desainnya yang hanya terdiri dari tiga buah transformasi linear *theta*, *Pi1*, dan *Pi2*, serta satu buah transformasi non-linear *gamma*. Operasinya diimplementasikan hanya menggunakan operasi-operasi *bit-wise* serta operasi pergeseran bit. Namun struktur yang dirancang, diharapkan memberikan keamanan data yang cukup baik.

Suatu *cipher* dikatakan memiliki peluang keamanan yang sempurna (*perfect secrecy*), jika antara *plaintext* dan *ciphertext* tidak memiliki keterhubungan satu sama lain. Untuk menganalisa ada tidaknya keterhubungan tersebut pada algoritma Noekeon, dilakukan pengujian terhadap sebuah *plaintext*, namun dilakukan enkripsi dengan beberapa kunci yang berbeda.

III.3.2 Proses Enkripsi dan Dekripsi

Struktur enkripsi dan dekripsi pada Noekeon dirancang menggunakan kode maupun rangkaian yang sama. Sehingga akan menghasilkan kesederhanaan dalam implementasinya.

III.3.3 Penjadwalan Kunci

Penjadwalan kunci pada Noekeon digunakan pada mode *indirect-key*. Metode ini akan lebih memperkuat algoritma terhadap usaha kriptanalisis, terutama dalam usaha untuk mendapatkan kunci yang dipakai. Penjadwalan kunci dilakukan melalui proses yang sama dengan algoritma Noekeon itu sendiri, yaitu melalui tahap *Theta*, *Pi1*, *gamma*, dan *Pi2*, dan dilakukan putaran sebanyak 16 kali. Namun *working-key* yang digunakan adalah *NullVektor*. Sehingga kunci yang akan menjadi kunci enkripsi mempunyai kekuatan yang sama dengan *cipher* Noekeon itu sendiri.

III.3.4 Sifat Simetris, Kunci lemah, dan Kunci Setengah Lemah

Sifat simetri setiap putaran pada suatu algoritma kriptografi memberikan peluang seorang kriptanalisis untuk membongkar suatu *ciphertext* dengan mengaplikasikan suatu teknik serangan tertentu. Pada Noekeon ini dapat terjadi apabila dalam setiap transformasinya tidak menambahkan *round constant*. Seperti telah dijelaskan, sifat simetris pada Noekeon dihilangkan dengan penggunaan *round constant* yang berbeda pada setiap putarannya. Sehingga secara praktis akan menghilangkan kemungkinan timbulnya kunci lemah dan kunci setengah lemah, seperti yang dimiliki oleh DES.

III.3.5 Efek Avalanche

Suatu algoritma kriptografi memenuhi kriteria *Strict Avalanche Criterion* (SAC) apabila rata-rata perubahan bit keluaran terhadap berubahnya satu bit pada masukan setidaknya adalah 50% . Pada simulasi ini sudah memenuhi kriteria *Strict Avalanche Criterion*, karena rata-rata perubahan bit keluaran terhadap berubahnya satu bit pada masukan lebih besar dari 50%, yaitu : 52.78 % untuk perubahan pada plain text dan 52.37 % untuk perubahan pada kunci (dari hasil percobaan).

III.3.6 Kesalahan Propagasi

Dimungkinkan terjadi kesalahan baik pada proses enkripsi maupun saat proses pertukaran data melalui media tertentu yang disebabkan oleh adanya interferensi baik oleh gelombang frekuensi maupun yang dilakukan oleh penyadap, sehingga terjadi perubahan informasi atau bit-bit tertentu pada *ciphertext*. Untuk melihat sejauh mana kesalahan tersebut berpengaruh terhadap *plaintext* hasil dekripsi, pada analisa ini dilakukan terhadap satu blok data yang dipresentasikan dengan sebuah file sebesar 128 bit. Data hasil enkripsi, *ciphertext* yang diperoleh dilakukan perubahan beberapa bit pada blok pertama (16 karakter ASCII) sebagai simulasi terjadinya kesalahan. Kemudian *ciphertext* tersebut didekripsi kembali untuk memperoleh *plaintext*.

Dari Hasil yang diperoleh, ternyata *plaintext* yang diperoleh kembali mengalami kerusakan pada blok pertama saja. Hal ini terjadi karena pada algoritma Noekeon yang diimplementasikan ini menggunakan mode ECB, yaitu masing-masing blok dioperasikan secara independen. Kesalahan yang terjadi pada sebuah blok *plaintext*, tidak akan mempengaruhi kepada blok lainnya saat dilakukan enkripsi. Sehingga saat dilakukan proses dekripsi, kesalahan yang terjadi hanya pada blok yang bersangkutan saja.

III.3.7 Kekuatan Terhadap Jenis Serangan *Brute-Force*

Serangan bertipe *brute-force* adalah dengan menerapkan percobaan setiap kemungkinan kunci yang ada satu per satu sampai diperoleh *plaintext* yang diharapkan. Waktu yang diperlukan untuk memperoleh kunci yang diharapkan, selalu berbanding lurus dengan panjang bit kunci yang dimiliki oleh algoritma kriptografi, dan makin cepat prosesor yang dipakai, makin cepat waktu yang pula dibutuhkan untuk dapat memperoleh kunci tersebut.

Dengan panjang kunci 128 bit, berdasarkan perkiraan pakar kriptografi dengan mengacu pada hukum **Moore** maka algoritma Noekeon akan bertahan mampu terhadap serangan bertipe *brute-force* selama ± 100 tahun.

IV KESIMPULAN

Dari hasil analisa pada tugas ini, dapat diambil kesimpulan sebagai berikut :

1. Pada tugas ini berhasil membuat program simulasi algoritma kriptografi Noekeon dengan bahasa pemrograman Borland C++6 dan berjalan dengan baik sehingga dapat menerangkan cara kerjanya.
2. Pencampuran antara *state* dari kunci *state* dan *plaintext* pada proses theta akan menghasilkan nilai *state* yang jauh berbeda dari *state* awal, dengan diikuti pula proses pergeseran bit dan operasi gamma, kemudian dilakukan perputaran 16 kali, menghasilkan enkripsi data yang kuat.
3. Semua kunci lemah dan kunci setengah lemah yang dimiliki oleh DES, tidak berpengaruh apabila diterapkan pada algoritma Noekeon, sehingga pada algoritma ini tidak ada pembatasan kunci yang digunakan ataupun pemanfaat kunci-kunci tersebut untuk melakukan eksploitasi terhadap *ciphertext*. Tidak tampak kelemahan algoritma Noekeon dibanding algoritma DES yang mengacu pada analisa kunci lemah dan setengah lemah
4. Efek *avalanche* yang diperoleh memenuhi kriteria yang cukup baik sesuai kriteria *Strict Avalanche Criterion (SAC)* yaitu pada perubahan satu bit kunci ataupun *plaintext* menyebabkan lebih dari separuh perubahan bit pada *ciphertext*..
5. Kesalahan atau kerusakan yang terjadi pada sebuah blok *plaintext*, tidak akan mempengaruhi kepada blok lainnya saat dilakukan enkripsi.
6. Dengan panjang kunci 128 bit, algoritma Noekeon akan bertahan mampu terhadap serangan bertipe *brute-force* selama ± 100 tahun.

DAFTAR PUSTAKA

- [1] Brickell Ernest F., Odlyzko, Andrew M.. *Cryptanalysis: A Survey of Recent Result*.
- [2] Daemen, Joan, Peeters, Michaël. Van Assche, Gilles. and Rijmen, Vincent. *NOEKEON Block Cipher, Nessie Proposal*. October 27, 2000.
- [3] Daemen, Joan, Peeters, Michaël. Van Assche, Gilles. and Rijmen, Vincent. *NOEKEON, Slide*. September 13, 2000.
- [4] Daemen, Joan, *Propagation and Correlation, Chapter 5 of Cipher and Hash Function Design*, distributed as PDF file as annex to AES submission Rijndael.
- [5] Daemen, Joan. L. Knudsen. Rijmen, Vincent. *The Block Cipher Square*, Paper.
- [6] Feistel, H. *Cryptography and Computer Privacy*. Scientific American, 1973.
- [7] Menezes, P. Van Oorschot, and S. Vanstone. *The Handbook of Applied Cryptography*, CRC Press, 1996.
- [8] N.P. Smart, *Physical Side-Channel Attacks On Cryptographic Systems*.
- [9] P. Michael. *Secure Data Networking*, Artech House, Inc. 1993.
- [10] Schneier, B. *Applied cryptography Second Edition*. John Wiley & Sons, Inc. 1996.
- [11] St Dennis, Tom, *Block Cipher, an Introduction to Modern Cryptanalysis*, Algonquin College, 2001.
- [12] Mohammad Sbastian Widodo, “Simulasi Algoritma Kriptografi Noekoen dalam Penyandian Data “, Tugas Akhir STT Telkom, 2002
- [13] St Dennis, Tom, *Differential Cryptanalysis of NOEKEON – DRAFT*, Algonquin College, 2000.
- [14] Webster, A. and S. Tavares. *On the Design of S-Boxes*. Advances in Cryptology, CRYPTO 1985.