Analisis Perbandingan Algoritma AES dan GOST

Yudhistira - 13508105
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
if18105@students.if.itb.ac.id

***Abstract*— Dalam dunia kriptografi, banyak sekali algoritma yang digunakan untuk mengenkripsi. Dua di antara algoritma tersebut adalah algoritma AES yang dikenal juga dengan algoritma Rijndael, dan algoritma GOST.**

AES dan GOST merupakan varian dari algoritma *block cipher*. Algoritma AES merupakan algoritma *block cipher* dengan ukuran blok 128-bit, dan panjang kunci yang bisa diubah-ubah, yaitu 128-bit, 192-bit, dan 256-bit. Sedangkan algoritma GOST merupakan algoritma *block cipher* dengan ukuran blok 64-bit dan panjang kunci 256-bit.

Kedua algoritma tersebut merupakan algoritma yang sangat kuat. Namun, masing-masing memiliki kekurangan dan kelebihan. Untuk menentukan algoritma mana yang lebih baik, dibutuhkan analisis perbandingan antar kedua algoritma.

Dalam tulisan ini, penulis akan mencoba melakukan studi analisis perbandingan antar kedua algoritma. Analisis akan dilakukan dari perbandingan spesifikasi, kompleksitas algoritma, dan kriptanalisis yang dilakukan pada kedua algoritma.

Kata Kunci*— GOST, AES, Analisis, *Block Cipher

I. PENDAHULUAN

Dewasa ini, perkembangan Teknologi Informasi telah membuat informasi menjadi sangat mudah didapatkan. Hanya dengan mengetikkan kata kunci yang diinginkan di situs pencarian, kita akan dengan mudah mendapatkan banyak *link* menuju berbagai informasi terkait dengan kata kunci yang diketikkan. Informasi yang muncul bisa puluhan, ratusan, bahkan ribuan. Semua orang merasa nyaman dan termudahkan dengan kemajuan Teknologi Informasi yang begitu pesat.

Namun demikian, seiring dengan perkembangan Teknologi Informasi yang begitu pesat, muncul isu yang kurang mengenakkan bagi sebagian pengguna internet, terutama bagi orang-orang yang ingin melindungi datanya agar tidak dicuri oleh orang lain, dan orang yang tidak ingin pesan rahasianya diketahui oleh pihak yang tidak seharusnya mengetahuinya. Ya, isu yang kurang mengenakkan itu adalah isu keamanan informasi yang semakin melemah.

Perkembangan Teknologi Informasi yang begitu pesat membuat perkembangan teknik pencurian data pun menjadi sangat bervariasi dan berbahaya bagi para pengguna jasa Teknologi Informasi. Orang-orang yang tidak bertaggung jawab bisa dengan mudah memperoleh informasi yang tidak seharusnya mereka ketahui dengan beberapa teknik tertentu. Karena isu inilah, kemudian muncul ilmu kriptografi. Dengan kriptografi, seseorang dapat melindungi data atau informasi miliknya. Hanya dengan seizinnya seseorang dapat membuka dokumen tersebut.

Algortima kriptografi yang pertama kali muncul adalah algoritma kriptografi klasik. Algoritma kriptografi klasik merupakan algoritma yang berbasis karakter. Cara kerja algoritma ini adalah dengan menggeser karakter ke kiri atau ke kanan beberapa karakter (substitusi), atau dengan mengubah posisi huruf di dalam *plaintext* (transposisi). Termasuk ke dalam algoritma ini adalah *Caesar Cipher, Vigenere Cipher, Playfair Cipher, Affine Cipher, Hill Cipher,* dan masih banyak lagi.

Algoritma kriptografi klasik ternyata sangat mudah dipecahkan. Bahkan, algoritma *Caesar Cipher* 26 karakter dapat dipecahkan dengan melakukan *brute-force* attack pada ke-26 kemungkinan kunci. Karena itulah kemudian muncul algoritma kriptografi modern.

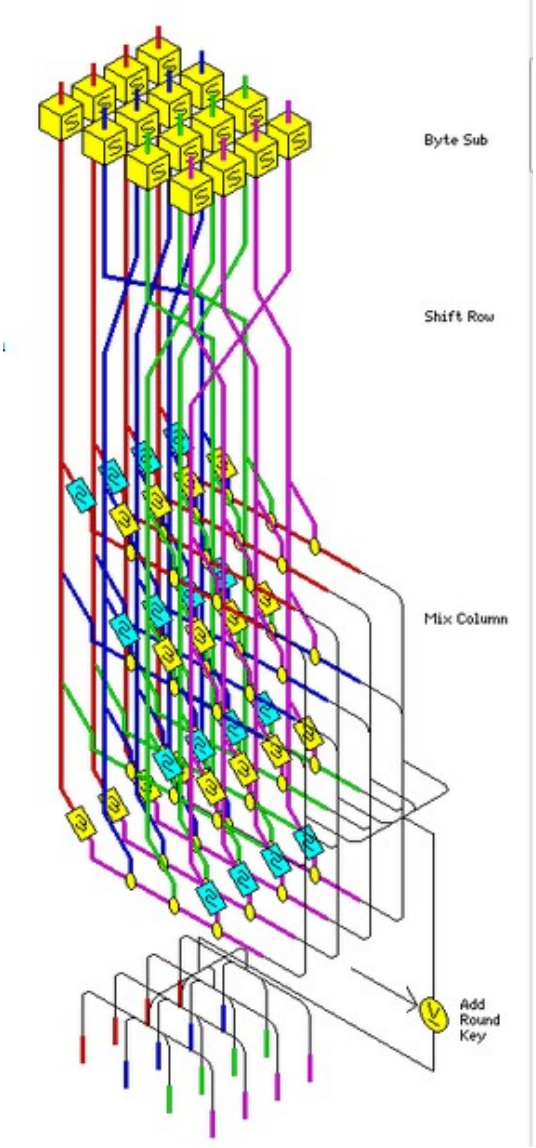
Berbeda dengan algoritma kriptografi klasik yang berbasis, algoritma kriptografi modern merupakan algoritma yang berbasis bit. Pada algoritma ini, baik kunci, *plaintext,* maupun *ciphertext* tidak diproses per karakter. Kunci, *plaintext,* dan *ciphertext* dalam algoritma kriptografi modern diproses dalam rangkaian bitnya. Algoritma kriptografi modern sering menggunakan operator *xor* dalam proses enkripsinya. Munculnya algoritma kriptografi modern tidak semata-mata mengeliminasi algoritma kriptografi klasik. Dalam algoritma kriptografi modern, terdapat prinsip-prinsip yang dipakai pada algoritma kriptografi klasik, seperti substitusi dan transposisi. Namun, dalam algoritma kriptografi modern, proses substitusi dan transposisinya lebih rumit, sehingga sulit untuk dipecahkan.

Algoritma kriptografi modern ada 2 jenis. Jenis pertama adalah *Stream Cipher*, yang beroperasi pada *bit* tunggal. *Stream Cipher* melakukan enkripsi dengan cara bit per bit. Contoh algoritma *Stream Cipher* adalah *Vernam Cipher.* Jenis kedua adalah *Block Cipher,* yang beroperasi pada *block-bit*. *Block Cipher* melakukan enkripsi dengan cara blok per blok. Contoh algoritma *Block Cipher* adalah DES, AES, GOST, LOKI, RC2, FEAL, dan sebagainya.

Dari sekian banyak algoritma *Block Cipher,* algoritma yang paling populer adalah algoritma AES yang dikenal juga dengan nama algoritma Rijndael. Algoritma ini sangat sulit untuk dipecahkan. Namun demikian, bukan berarti AES merupakan algoritma terkuat yang kini ada dalam dunia kriptografi. Dalam tulisan ini, penulis akan melakukan analisis perbandingan antara Algoritma AES dan Algoritma GOST.

II. Algoritma AES (Advanced Encryption Standard)

AES sebetulnya bukan merupakan algoritma. Namun, orang biasa menyebut AES sebagai Algoritma AES. AES adalah sebuah standar enkripsi data. Algoritma yang dijadikan AES adalah algoritma Rijndael yang ditemukan oleh Vincent Rijmen dan Joan Daemen. Mereka berdua memenangkan kompetisi pemilihan AES yang diselenggarakan oleh *National Institute of Standards and Technology (NIST)*. Nama Rijndael itu sendiri berasal dari bagian depan nama belakang kedua penemunya, ditambah huruf "l" yang merupakan huruf ketiga dari kata Belgia yang merupakan negara asal kedua ilmuwan tersebut.

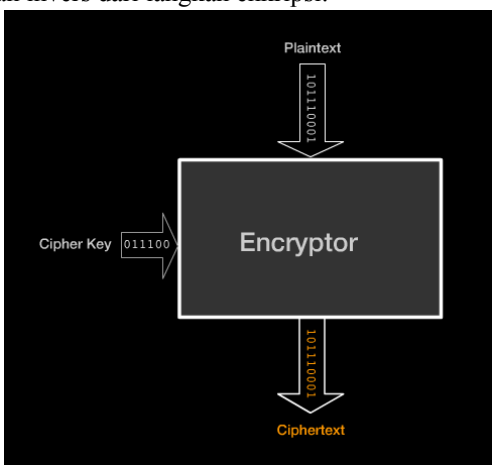


Gambar 1: Visualisasi Algoritma AES

AES pertama kali digunakan secara luas oleh pemerintah Amerika Serikat sebagai pengganti algoritma DES. Algoritma Rijndael merupakan algoritma kunci simetri, yang berarti, ia menggunakan kunci yang sama untuk melakukan enkripsi maupun dekripsi. Berbeda dengan pendahulunya, algoritma DES, algoritma AES tidak berorientasi bit, AES beroperasi dalam orientasi *byte*. AES juga tidak menggunakan Jaringan Feistel dalam proses enkripsinya.

Algoritma AES memiliki panjang blok 128 bit dan mendukung 3 variasi panjang kunci, yaitu 128 bit, 192 bit, dan 256 bit. Berdasarkan ketiga panjang kunci tersebut, AES dibagi menjadi tiga, yaitu AES-128, AES-192, dan AES-256. Kebanyakan orang memakai AES-128 atau AES-256. AES-192 jarang digunakan oleh para pemrogram.

AES beroperasi berdasarka prinsip desain yang diketahui sebagai *jaringan substitusi-permutasi.* Dengan desain ini, AES dapat diimplementasikan dengan efektif baik pada *hardware* maupun *software.* Algoritma AES dispesifikasikan sebagai jumlah perulangan dari transformasi loop yang mengkonversi *plaintext* menjadi *ciphertext.* Masing-masing *loop* terdiri atas beberapa proses, termasuk satu langkah yang bergantung pada kunci enkripsinya. Untuk mendekripsi, dilakukan langkah-langkah invers dari langkah enkripsi.



Gambar 2: Gambaran Enkripsi

Berikut adalah deskripsi dari algoritma AES.

1. Ekspansi Kunci – kunci diturunkan dari cipher dengan menggunakan *Rijndael's key schedule.* Pertama, kunci dibagi menjadi 8, masing-masing 32 bit. Kemudian, masing-masing sub-kunci diubah ke dalam heksadesimal, kemudian dirotasi. Contoh:

Kunci awal : 5F 6D 1B 2E.
Hasil rotasi : 6D 1B 2E 5F.
Lalu, dilakukan prosedur Rcon. Setelah itu, dilakukan pembuatan S-box. Setelah itu, dilakukan penjadwalan kunci.

2. *Initial Round*

Pada AES-256, langkah-langkah berikut diulang sebanyak 14 kali.

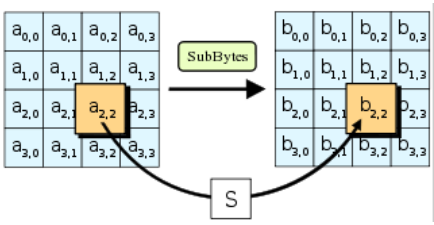
- *AddRoundKey*

Masing-masing byte pada state dikombinasikan dengan kunci yang sudah dijadwalkan dengan menggunakan *bitwise xor.*

3. *Rounds*

- *SubBytes*

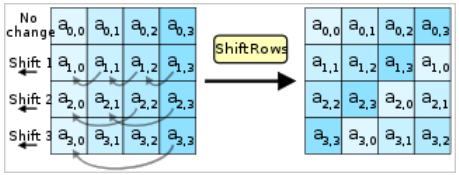
Pada langkah ini, masing-masing byte dalam matriks blok di-*update* dengan menggunakan S-box. Untuk menghindari serangan yang berdasar pada persamaan aljabar sederhana, S-box dibuat dengan mengkombinasikan fungsi invers dengan transformasi *affine.*



Gambar 3: Langkah *SubByte*

- ShiftRows

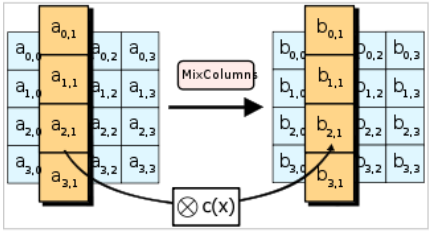
Langkah ShiftRows dilakukan pada baris tiap-tiap state. Baris pertama tidak digeser. Pergeseran hanya untuk baris kedua, dan seterusnya. Masing-masing byte pada baris kedua digeser satu ke kiri. Lalu, baris ketiga digeser sebanyak 2 kali. Dan baris keempat digeser sebanyak 3 kali.



Gambar 4: *ShiftRows*

- *MixColumns*

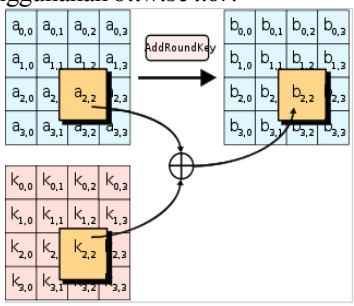
Pada *MixColumn,* empat byte dari masing-masing kolom dikombinasikan dengan menggunakan transformasi lanjar yang bisa diinvers. *MixColumns* mengambil 4 byte *input*, dan mengeluarkan 4 byte *output,* di mana setiap input byte mempengaruhi keempat *output* yang lain. Pada operasi ini, masing-masing kolom dikalikan dengan matriks tertentu. Multiplikasi didefinisikan sebagai: Multiplikasi dengan 1 berarti tidak ada perubahan pada blok. Multiplikasi dengan 2 berarti melakukan *shift* byte ke kiri dan multiplikasi dengan 3 berarti *shift* byte ke kiri, lalu lakukan *xor* pada nilai awal yang belum di-*shift*.



Gambar 5: *MixColumns*

- *AddRoundKey*

Pada langkah ini, subkunci dikombinasikan dengan state terkini. Pada masing-masing *round,* sebuah subkunci diturunkan dari kunci utama dengan menggunakan penjadwalan kunci. Masing-masing subkunci ditambahkan dengan mengkombinasikan masing-masing byte dari subkey dengan menggunakan *bitwise xor.*



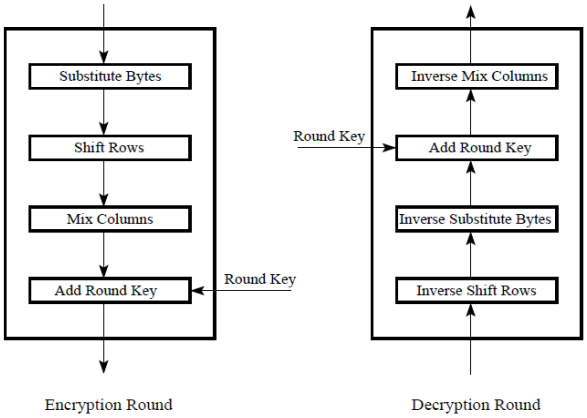
Gambar 6: *AddRoundKey*

4. *Final Round*

Pada *round* ini, dilakukan operasi yang sama seperti pada *Round,* namun tanpa *MixColumns.*

Untuk melakukan dekripsi algoritma AES, dilakukan langkah invers dari langkah yang telah dilakukan pada proses enkripsi. *Round* dari deskripsi adalah sebagai berikut:

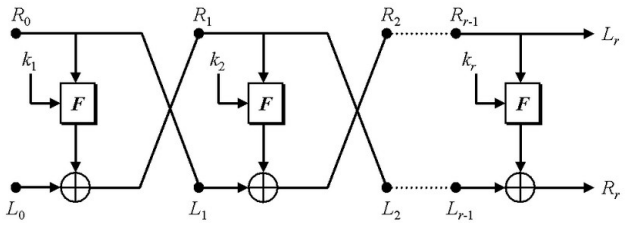
1. *Inverse ShiftRows*
2. *Inverse SubBytes*
3. *AddRoundKey*
4. *Inverse MixColumn*



Gambar 7: *Enkripsi dan Dekripsi*

III. ALGORITMA GOST

Algoritma GOSt merupakan Algoritma buatan Rusia yang merupakan tandingan dari algoritma DES buatan Amerika Serikat. Secara struktural, algoritma ini sangat mirip dengan algoritma DES. Seperti halnya algoritma DES, Algoritma GOST menggunakan jaringan Feistel dalam struktur enkripsinya.

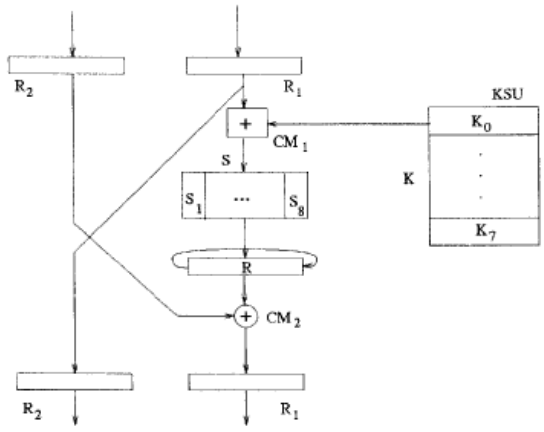


Gambar 8: Jaringan Feistel

Algoritma ini memiliki ukuran blok pesan 64-bit. Berbeda dengan AES yang memiliki ukuran blok pesan 128-bit. Algoritma ini juga memiliki jumlah *round* yang lebih banyak dari AES-256, yaitu 32 putaran. Setiap putaran menggunakan 8 buah kunci internal yang dijadwalkan penggunaannya.

Fungsi yang digunakan pada tiap *round* dalam algoritma GOSt sangat sederhana. Kita tinggal menambahkan subkunci yang di-mod dengan 2^{32}. Lalu hasilnya ditaruh ke S-box, dan rotasikan hasil tersebut ke kiri sebanyak 11 bit. Hasilnya akan dipakai di *round* berikutnya. Begitu seterusnya sampai 32 *round.*

Penjadwalan kunci yang dilakukan oleh GOSt pun sangat sederhana. Pertama, bagi kunci utama menjadi 8 buah subkunci berukuran 32-bit. Masing-masing subkunci. Masing-masing subkunci digunakan empat kali di dalam algoritma. Dua puluh empat *round* pertama menggunakan kata kunci sesuai urutan. Sedangkan 8 *round* terakhir menggunakan kunci dengan urutan terbalik.



Gambar 9: Satu *round* algoritma GOST

IV. Kriptanalisis Pada Algoritma AES dan GOST

Algoritma AES dan GOST merupakan algoritma yang sangat kuat. Ketangguhan kedua algoritma ini mengundang para kriptografer untuk mencari lubang dalam kedua algoritma tersebut. Di bawah ini merupakan metode-metode yang pernah dicoba untuk memecahkan kedua algoritma

A. Kriptanalisis Algoritma AES

Algoritma AES sudah banyak dicoba untuk dipecahkan. Salah satunya dilakukan oleh Alex Biryukov dan Dmitry Khovratovich. Kedua ilmuwan dari Universitas Luxemburg tersebut mencoba meng-kriptanalisis AES-256 dengan berbagai serangan terhadap kunci.

Chabaud dan Joux melakukan kriptanalisis dengan melakukan *local collisions.* Idenya adalah untuk memasukkan sebuah fungsi ke dalam sebuah *round* yang akan menyebabkan gangguan, lalu ada fungsi lain yang dimasukkan yang akan membetulkannya. Pola yang terbentuk dari pemasukkan fungsi ini akan menyebar karena penjadwalan pesan menyebabkan lebih banyak gangguan di *round* yang lain. Tujuan dari pemasukan *collisions* ini adalah sampai gangguannya menyedikit sesedikit mungkin untuk mengurangi kompleksitas yang diperlukan pada serangan.

Pada serangan terhadap kunci, kedua ilmuwan Rusia itu menginjeksi fungsi pembeda pada kunci, tidak hanya pada *plaintext* seperti pada kriptanalisis diferensial. Namun, penyerang tidak dapat mengontrol kunci tersebut dan serangan akan bekerja untuk beberapa pasang kunci dengan perbedaan tertentu yang dibuat oleh fungsi pembeda.

Lalu metode kedua yang dilakukan kedua ilmuwan Luxemburg itu adalah *Boomerang Attack.* Serangan ketiga adalah serangan terhadap *round* 7 dari algoritma AES-256 tersebut.

Dari tiga serangan tersebut, belum ada yang bisa menjebol algoritma AES-256. Metode-metode di atas bukan merupakan ancaman bagi algoritma AES-256.

Kriptanalisis berikutnya adalah dengan metode *Improved Related-Key Impossible Differential Attack* oleh Hadi Soleimany, Alireza Sharifi, dan Mohammadreza Aref dari Sharif University of Technology, Iran. Ketiga ilmuwan ini mencoba menyerang algoritma AES-256 dengan menyerang *round* 8 dari algoritma ini. Namun, percobaan ini tetap belum menimbulkan efek yang berarti bagi keamanan algoritma AES-256.

Serangan terakhir adalah *Key Recovery Attack* oleh Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, dan Adi Shamir, yang merupakan gabungan dari ilmuwan-ilmuwan Eropa. Keempad ilmuwan ini mencoba metode yang bisa dipraktikkan dengan menyerang 10 *round* algoritma AES. Algoritma ini memiliki kompleksitas serangan 2^{38}.

B. Kriptanalisis Algoritma GOST

Algoritma GOST lebih kuat daripada algoritma DES. Setelah Algoritma DES terpecahkan, orang mulai mencoba memecahkan algoritma GOST. Berbagai cara telah ditempuh untuk menembus pertahanan algoritma GOST. Beberapa di antaranya akan dibahas di bawah ini.

Nicolas T. Courtois dari University College London melalui papernya mencoba memecahkan algoritma GOST. Ia mencoba metode Kriptanalisis

Aljabar dan Serangan dengan data kompleksitas yang rendah. Selain itu, ia juga mencoba serangan dengan terlebih dahulu mencari plaintext yang diketahui (2^{32} dan 2^{64}). Selain itu, ia juga mencoba *in-the-middle* attack. Seluruh cara yang ia coba belum bisa menembus algoritma GOST karena seluruhnya terlalu lama bahkan apabila dilakukan dengan komputer tercepat di dunia.

Nicolas T. Courtois juga mencoba metode diferensial bekerjasama dengan Michal Misztal. Dengan metode ini, dekripsi algoritma GOST belum bisa dilakukan, namun kompleksitas serangannya menurun. Lebih baik dari penelitian Courtois sebelumnya yang kompleksitasnya besar sekali.

V. Kompleksitas Serangan Terhadap Algoritma AES dan GOST

Algoritma AES dan GOST merupakan dua algoritma yang sangat kuat. Masing-masing memiliki kelebihan dan kekurangan masing-masing.

Algoritma AES memiliki kelebihan berupa penjadwalan kunci yang lebih sulit, panjang blok yang 128-bit, dan struktur *round* yang sangat *complicated.* Di sisi lain, algoritma GOST memiliki keunggulan *round* yang mencapai 32 buah, dan panjang kunci yang 256-bit.

Kekurangan yang dimiliki AES adalah jumlah *round*nya yang hanya 14 buah sehingga mungkin akan ada komputer supercepat yang mampu mendekripsinya dengan *exhaustive search*.

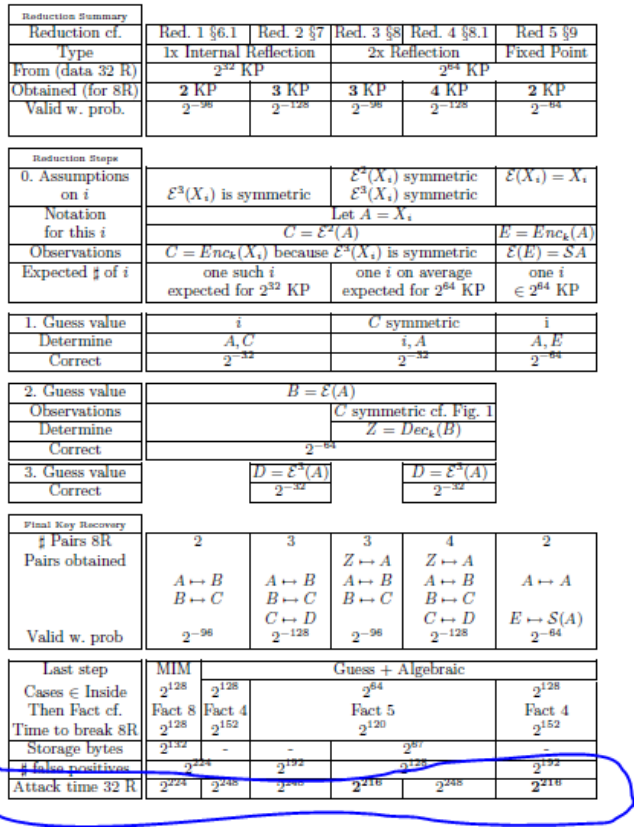
Algoritma GOST pun memiliki kekurangan. Operasi pada *round*nya sangat sederhana. Sehingga mungkin saja ada kriptanalisis yang menggunakan metode aljabar tertentu yang bisa mendekripsinya.

Dari serangan-serangan yang dilakukan pada kedua algoritma, hasilnya cukup di luar dugaan. Berikut disajikan tabel kompleksitas serangan yang dilakukan terhadap algoritma AES.

Tipe	Kompleksitas Serangan
Serangan Diferensial	2^{53}
Partial Sums	2^{119}
Impossible Differential	2^{89}
Meet in the middle	2^{32}
Impossible Differential (2)	2^{119}

Tabel 1: Tabel Kompleksitas Serangan terhadap AES

Sedangkan tabel di bawah ini merupakan statistik serangan terhadap GOST.



Kedua tabel itu merupakan hasil penelitian para ilmuwan kriptanalisis yang berjuang untuk menemukan celah dalam kedua algoritma.

VI. Kesimpulan

Baik algoritma AES maupun GOST, keduanya memiliki kelebihan dan kekurangannya sendiri. Algoritma AES memiliki kelebihan pada panjang blok dan struktur *round* yang sangat rumit. Sedangkan Algoritma GOST memiliki kelebihan berupa *round* yang panjang dan panjang kunci yang *default,* sebesar 256-bit. Kekurangan algoritma AES adalah jumlah *round* yang sedikit. Sedangkan kekurangan algoritma GOST adalah struktur *round*nya yang terlalu sederhana. Dari statistik kompleksitas serangannya, algoritma GOST mendapa serangan paling baik sebesar 2^{216} serangan. Sedangkan algoritma AES 2^{32} serangan. Hal ini bukan berarti algoritma GOST lebih baik. Ini mungkin terjadi karena algoritma GOST lebih jarang dicoba untuk di kriptanalisis. Faktanya, kedua algoritma sangat sulit untuk dikriptanalisis. Keduanya belum ada yang bisa memecahkan. Kesimpulannya, kedua algoritma merupakan algoritma yang sangat aman untuk enkripsi data. Tinggal pilih saja. Apabila anda suka dengan struktur yang rumit dengan panjang kunci yang tidak terlalu panjang, gunakanlah algoritma AES. Namun apabila anda lebih suka struktur yang sederhana dengan panjang kunci yang sangat panjang, gunakanlah GOST.

Daftar Pustaka

[1] Hadi Soleimany, Alireza Sharifi, Mohammadreza Aref: Improved Related-Key Impossible Differential Attacks on 8-Round AES-256. https://ece.uwaterloo.ca/~a9sharif/papers/6.pdf

[2] Alex Biryukov, et.all: Key Recovery Attacks of Practical Complexity on AES Variants With Up To 10 rounds. http://eprint.iacr.org/2009/374.pdf

[3] Nicolas T. Courtois: Algebraic Complexity Reduction and Cryptanalysis of GOST. http://www.nicolascourtois.com/papers/gostac11.pdf

[4] Nicolas T. Courtois: Differential Cryptanalysis of GOST. http://eprint.iacr.org/2011/312.pdf

[5] http://en.wikipedia.org/wiki/Advanced_Encryption_Standard , diakses pada 18 Mei 2012.

[6] http://www.rsa.com/rsalabs/faq/images/feistel.gif diakses pada 19 Mei 2012

[7] http://www.quadibloc.com/crypto/co040401.htm diakses pada 18 Mei 2012

[8] https://engineering.purdue.edu/kak/compsec/NewLectures/Lecture8.pdf diakses pada 18 Mei 2012

[9] http://en.wikipedia.org/wiki/GOST_(block_cipher) diakses pada 19 Mei 2012

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 20 Maret 2012

ttd

Yudhistira
13508105