Pengembangan dan Implementasi Algoritma Tiger

I Nyoman Prama Pradnyana - 13509032
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
prama.pradnyana@itb.ac.id

***Abstract*—Fungsi hash adalah salah satu aplikasi kriptografi yang sangat penting. Fungsi hash dapat melakukan pengamanan data-data penting yang rentan termanipulasi. Fungsi hash ini sudah banyak digunakan di berbagai bidang antara lain pada keamanan jaringan. Contoh fungsi hash yang terkenal antara lain MD dan SHA. MD dan SHA merupakan fungsi hash yang tergolong kuat karena untuk melakukan kriptnalisis diperlukan waktu yang sangat lama. Namun pada tahun 1990 ditemukanlah teknik pemecahan algoritma tersbut. Menanggapi hal itu, muncul banyak pertanyaan bagaimana cara membuat fungsi hash yang kuat. Munculah gagasan fungsi hash baru yang lebih kuat dan cepat yang diberi nama algoritma Tiger. Kekuatan algoritma Tiger ini terletak pada mekanisme pembuatan hasil hash yang berbeda dari fungsi hash pada umumnya. Meskipun demikian, tentu masih terdapat kelemahan pada fungsi ini. Pada makalah ini dilakukan analisis dan pengembangan fungsi hash yang bertujuan untuk mengetahui kelemahan algoritma Tiger dan dapat mengembangkan algoritma ini sehingga algoritma Tiger ini menjadi lebih kuat.**

Index Terms- Fungsi Hash Tiger

I. PENDAHULUAN

Saat ini pengiriman data sudah menjadi hal yang sangat sering dilakukan. Pengiriman data yang paling sering dilakukan adalah pengiriman dengan melalui media internet. Pengiriman data melalui internet memiliki berbagai kelemahan salah satunya tentang integritas data. Data atau pesan saat dikirim bisa saja mengalami perubahan sehingga ketika sampai di tujuan pesan sudah berubah. Hal ini menjadikan pengiriman data melalui media internet masih rawan. Dengan menggunakan fungsi hash, integritas pesan dapat dicek apakah pesan tersebut asli atau sudah termanipulasi.

Saat ini fungsi hash telah menjadi kebutuhan dalam pengiriman pesan. Fungsi hash yang terkenal adalah MD dan SHA. Kedua algoritma ini terkenal karena kekuatan dan kecepatan fungsinya dibandingkan dengan fungsi hash lainnya. Kekuatan yang dimaksud adalah ketahanan fungsi tersebut terhadap serangan kriptnalisis sedangkan kecepatan yang dimaksud adalah lama waktu yang dibutuhkan untuk melakukan proses pembuatan nilai hash. MD dan SHA dikatakan memiliki ketahanan yang kuat karena untuk melakukan kriptnalisis diperlukan waktu yang sangat lama.

Meskipun hash memiliki ketahanan terhadap serangan, namun hash tetap masih bisa dilakukan penyerangan. Pada tahun 1990 ditemukan *collision* fungsi MD dan SHA. Serangan yang dilakukan dengan memanfaatkan kelemahan struktural fungsi hashnya. Hal ini membuat banyak orang meragukan fungsi hash karena fungsi hash lain pasti bisa diserang dan hanya tinggal menunggu waktu saja [ROS96].

Ketika terjadi keraguan dengan fungsi hash, muncul ide tentang pembuatan fungsi hash baru yang mengantisipasi serangan yang dilakukan terhadap MD dan SHA. Algoritma ini dinamakan algoritma Tiger yang diambil dari sifat algoritmanya yang kuat dan cepat. Di samping itu algoritma ini dirancang juga pada prosesor 64 bit yang yang tidak ada fungsi hash yang mampu digunakan sebelumnya.

Keunggulan algoritma ini dibandingkan dengan algoritma lain adalah algoritma Tiger menggunakan S-Box yang terdiri dari 8-64 bit. Dengan menggunakan S-Box ini, algoritma tiger terhidar dari sifat kelinieran nilai hash. Hal ini disebabkan S-Box dilakukan pada selurih bit tidak hanya pada bit paddingnya saja. Disamping itu algoritma tiger menghasilkan pergeseran yang kuat ketika dilakukan putaran round dan panjang nilai hash yang dihasilkan lebih panjang dari hash sebelumnya sehingga usaha penyerangan menjadi jauh lebih besar. Selain itu keunggulan algoritma Tiger terletak pada *key schedule* pada proses putarannya yang mencegah penyerang memaksakan input di putaran yang dilakukan.

Pada makalah ini akan dibahas mengenai keunggulan tiger dibandingkan dengan hash lainnya, melakukan analisis terhadap kemungkinan serangan dan melakukan pengembangan terhadap fungsi tiger sehingga fungsi tiger

menjadi lebih kuat. Makalah ini juga dibataskan pada pembahasan terhadap fungsi hash satu arah saja.

Melalui makalah ini diharapkan mampu memberikan gambaran mengenai penyerangan terhadap algoritma tiger dan pengembangan yang bisa dilakukan. Selain itu diharapkan makalah ini mampu menjadi refrensi dalam pengembangan algoritma Tiger kedepannya mengingat masih sangat sedikit orang yang melakukan analisis terhadap algoritma tiger ini karena kerumitan prosesnya.

II. DASAR TEORI

2.1 Penyerangan Fungsi Hash

Fungsi hash adalah fungsi yang menerima masukkan string yang panjangnya sembarang dan mengkonversinya menjadi string keluaran yang panjangnya tetap [RIN05]. Keunggulan fungsi hash adalah tingkat kesensitifannya yang tinggi sehingga jika suatu data telah diubah sedikit saja, maka nilai hash yang dihasilkan jauh berbeda. Fungsi hash ini sangat sering digunakan dalam pengiriman pesan. Pengiriman pesan yang dilakukan melalui jaringan tentu tidak mendapat jaminan keamanan data tersebut. Pesan yang terkena noise atau hack akan memberikan arti pesan yang berbeda pada penerimanya. Dengan menggunakan fungsi hash ini dapat dilakukan pengecekan terhadap integritas pesan tersebut apakah pesan tersebut pesan asli atau pesan yang sudah termanipulasi.

Fungsi hash sangat banyak jenisnya. Fungsi hash yang terkenal adalah MD dan SHA. Fungsi hash MD atau sering disebut dengan *Message Digest* adalah fungsi hash yang menggunakan bit pengganjal pada akhir pesan dan panjang pesan. MD mengolah 512 bit kemudian dibagi menjadi 16 sub blok berukuran 32bit. Keluarannya adalah 4 blok yang berukuran 32 bit dan kemudian digabungkan sehingga menghasilkan nilai hash yang panjangnya 128 bit. Sedangkan untuk SHA pesan kelipatan 512 bit dan pada blok terakhir dibuat sepanjang 448 bit ditambah dengan 64 bit yang menyatakan panjang pesan. Pesan kemudian diolah dan digabungkan dan meghasilkan panjang 160 bit. Karena SHA lebih panjang 32 bit dibanding MD, maka dapat dikatakan jika kedua fungsi sha tersebut diserang dengan bruteforce SHA memerlukan waktu yang lebih lama diabandingkan dengan MD.

Meskipun fungsi hash mampu menentukan integritas suatu pesan, namun fungsi hash yang ada masih memiliki beberapa kelemahan dan hingga saat ini sudah banyak fungsi hash yang telah mampu dipecahkan bahkan fungsi MD dan SHA yang merupakan fungsi hash yang tergolong kuat. Penyerangan ini dilakukan dengan memanfaatkan kelemahan dari fungsi hash. Beberapa hal yang dapat dimanfaatkan dalam melakukan penyerangan antara lain kelemahan struktural fungsi hash, kelemahan analitik dan eksploitasi keduanya[LAT06]. Dengan memanfaatkan kelemahan fungsi hash tersebut, maka akan didapat suatu pola yang bisa dikembangkan untuk memecahkan fungsi hash tersebut.

Terdapat beberapa teknik penyerangan yang dapat dilakukan dengan memanfaatkan kelemahan fungsi hash. Teknik tersebut dibedakan menjadi 5 jenis yaitu *Collision Attack*, *Preimage Attack*, *2nd Preimage Attack*, *k-Collision Attack*, dan *K-Way(2nd) Preimage Attack* [LAT06].

Collision Attack memanfaatkan kelemahan struktur dari fungsi hash dimana fungsi hash pada umumnya memiliki nilai hash yang panjangnya tetap. Dengan kata lain pasti terdapat dua buah pesan yang memiliki nilai hash yang sama. Jadi *Collision Attack* berprinsip menemukan dua buah *message* berbeda $M_1 \neq M_2$ dengan $H(M_1)=H(M_2)$. Untuk *Preimage Attack* menggunakan prinsip brute force dimana akan diberikan nilai acak $Y \varepsilon \{0,1\}^n$ dan kemudian dilakukan pencarian *message* M yang memenuhi $H(M) = Y$. *2nd Preimage Attack* hampir sama dengan *Collision Attack* dimana diberikan sebuah *message* M_1 dan menemukan message M_2 yang memenuhi $H(M_1) = H(M_2)$. *K-Collision Attack* mengembangkan prinsip *Collision Attack* dimana dicari sejumlah K *message* M_i yang berbeda dimana $H(M_1)=\ldots=H(M_K)$. *K-Way (2nd) Preimage Attack* merupakan gabungan dari *K-Collision* dan *(2nd)Preimage Attack*.

2.2 Algoritma Tiger

Algoritma tiger adalah fungsi hash baru yang dirancang oleh Ross Anderson dan Elli Biham pada tahun 1996. Fungsi ini dirancang sebagai fungsi hash alternatif ketika ditemukannya *collision* pada fungsi hash MD dan SHA. Ketika ditemukannya *collision* pada algoritma MD dan SHA, banyak orang meragukan kekuatan fungsi hash lain mengingat MD dan SHA yang sangat terpercaya sudah bisa ditemukan *collision*. Dari sinilah mulai terbentuk suatu ide untuk membuat algoritma Tiger ini.

Algoritma Tiger diambil dari sifat algoritmanya yang sangat kuat dan cepat. Algoritma ini juga bisa dijalankan tidak hanya pada prosesor 32 bit, namun bisa juga bekerja pada prosesor 16 dan 64 bit dengan sangat cepat. Prinsip kerja algortima hash ini sebenarnya mirip dengan SHA. Pertama dilakukan *padding message* dimana pesan diubah menjadi kelipatan 512 bit. Untuk blok terakhir dilakukan padding dengan menambahkan bit 1 dan dilanjutkan dengan bit 0 hingga panjang message menjadi 448. Kemudian message tersebut ditambahkan dengan 64 bit yang merepresentasikan panjang pesan.

Tahap kedua adalah melakukan inisialisasi penyangga MD. Fungsi Tiger membutuhkan tiga buah penyangga yang terdiri dari 64 bit. Total panjang penyangga adalah 192 bit. Penyangga yang digunakan adalah :

A = 0x0123456789ABCDEF
B = 0xFEDCBA9876543210
C = 0xF096A5B4C3B2E187

Tahap ketiga adalah dengan mengubah message 512 menjadi 8 *block message* yang panjangnya 64 bit ($X_0..X_7$). Kemudian masing-masing block message tersebut dimasukkan proses *round* dan *key-schedule*. Setelah dilakukan putaran proses ini, didapatkan nilai a, b, c yang bernilai total keseluruhan 192 bit.

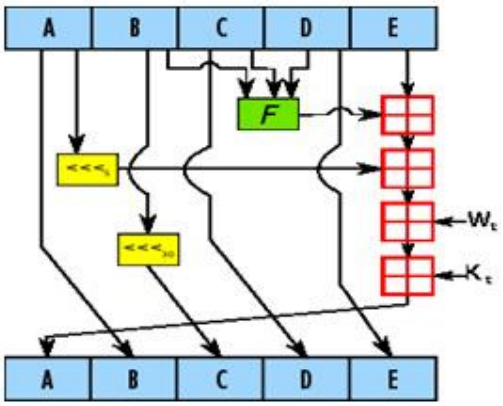
III. ANALISIS

3.1 Serangan pada fungsi hash

Berdasarkan penjelasan sebelumnya, terdapat kelemahan-kelemahan pada fungsi hash yang dapat dimanfaatkan dalam melakukan penyerangan. Sebagai contoh adalah penyerangan yang dilakukan pada fungsi SHA. Penyerangan yang dilakukan adalah dengan memproses dua pesan yang berbeda hingga ditemukan nilai hash yang sama. Ketika dilakukan kriptnalisis pada fungsi SHA, ditemukan *full collision* dengan memerlukan waktu pemecahan 80.000 jam. Jika telah mendapat dua message yang menghasilkan nilai hash yang berbeda, maka pesan asli dapat tergantikan dengan pesan yang palsu namun tetap memiliki nilai hash yang sama. Selain itu pada SHA juga berhasil dilakukan penyerangan *preimage* dimana ditemukan banyak pesan yang menghasilkan satu nilai SHA.

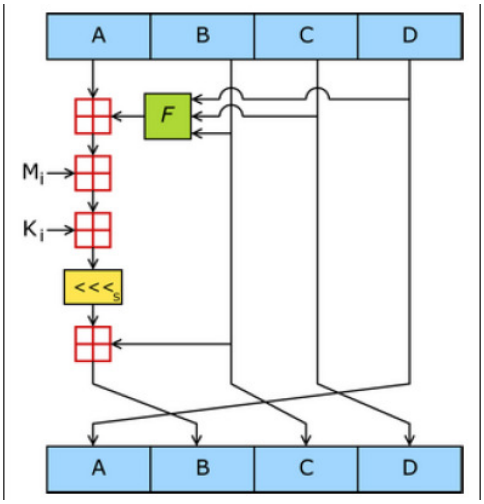
Dari penjelasan di atas, fungsi SHA yang tergolong kuat dari penyerangan masih bisa ditemukan *full collision*. Hal ini membuktikan fungsi SHA masih memiliki kelemahan baik dari struktur algoritmanya maupun struktur analitiknya. Jika dilihat dari sistem penyerangan yang dilakukan dengan menggunakan serangan *birthday attack*, dapat dikatakan pengguna dapat memaksakan sejumlah pesan ke dalam fungsi SHA ke dalam suatu proses putaran. Penyerangan ini terdiri dari beberapa tahap yang dilakukan terhadap beberapa putaran hingga didapatkan *collision* tiap putarannya. Hal ini dilakukan hingga ditemukan *full collision*.

Jika dilihat dari strukturnya, SHA menggunakan lima buah register yang digunakan untuk menghasilkan nilai hash. Pada prosesnya kelima register tersebut akan diproses menjadi lima register baru dan akhirnya menjadi nilai hash SHA. Dapat dilihat pada gambar 3.1 bahwa pesan hanya diproses pada register terakhir dan diubah menjadi register pertama. Sedangkan register lain seperti register A, C dan D tidak mengalami proses apa-apa dan hanya mengalami perubahan urutan dengan register lain. Hal ini bisa dimanfaatkan untuk melakukan langkah awal penyerangan dimana penyerang hanya perlu melakukan analisis terhadap register yang telah diproses dengan pesan sedangkan register lainnya bisa diabaikan. Jika sudah mendapatkan pola pada proses putaran pertama, maka sangat mungkin untuk mendapatkan pola yang kedua dan seterusnya hingga didapatkan full collision.



Gambar 3.1 Proses putaran SHA

Contoh lain adalah pada fungsi hash MD5. Algoritma MD5 juga memiliki kelemahan pada strukturnya. Dapat dilihat pada gambar dibawah dimana terdapat 3 register yag sama sekali tidak diproses dan hanya dilakukan proses pengubahan nilai register saja. Hal ini menyebabkan pencarian pola menjadi lebih mudah. Kriptnalisis hanya perlu melakuakn perubahan pada register pertamanya saja dan ketika mendaptakn nilai hasil proses yang sama, maka collision akan mudah ditemukan.



Gambar 3.2 Proses putaran MD5

Dari analisis struktur fungsi hash di atas, dapat dikatakan bahwa struktur pada fungsi hash SHA dan MD masih memiliki kelemahan yaitu pada putaran prosesnya. Putaran yang dilakukan pada fungsi SHA dan MD masih belum menjamin adanya perubahan yang terjadi setiap bitnya (*avalance effect*). Hal ini menyebabkan pencarian nilai pada setiap pola menjadi mudah karena hanya perlu melakukan analisis terhadap register yang terproses bersama *message*. Proses ini dilakukan terhadap seluruh putaran hingga ditemukan *full collision*-nya. Selain itu panjang nilai hash yang dihasilkan masih tergolong pendek. Semakin pendek panjang nilai hash, semakin mudah kriptanalis melakukan penyerangan.

3.2 Perbandingan Tiger dengan fungsi lain

Algoritma tiger adalah algoritma yang dibuat sebagai alternatif fungsi hash ketika ditemukannya *collision* pada fungsi hash MD dan SHA. Pembuatan algoritma ini sudah memperhitungkan berbagai tipe penyerangan yang digunakan dalam menyerang algoritma hash lainnya. Fungsi tiger pada dasarnya hampir sama dengan fungsi hash lain yang memiliki panjang yang tetap seperti SHA dll yaitu menggunakan padding message dan putaran proses register dan menggunakan prinsip *blok chipper*. Namun algoritma tiger ini memiliki beberapa kelebihan dibandingkan dengan fungsi hash pendahulunya. Jika dilihat dari strukturnya, algoritma tiger memiliki kelebihan antara lain :

1. Algoritma Tiger menggunakan fungsi S-Box 8-64 bit. S-Box ini lebih baik dibandingkan hanya dengan mengandalkan kombinasi logika. Dengan S-Box ini, fungsi tiger dikatakan terhindar dari sifat kelinieran hasil hash.
2. Adanya avalanche yang kuat pada setiap bit message setelah dilakukan putaran ke tiga. Avalanche adalah kondisi dimana bit-bit yang diganakan pada register saat putaran mengalami perubahan. Avalanche yang kuat menjamin adanya perubahan yang besar terhadap bit keluaran ketika terjadi perubahan sedikit bit masukkan.
3. Jika pada MD dapat diketahui blok yang intermediate yaitu pada blok yang dipadding, maka pada tiger blok intermediate diperpanjang sehingga proses kriptnalisisnya menjadi lebih rumit. Hal ini masih sama dengan SHA dimana padding message yang digunakan dilakukan penambahan hingga 512 bit.
4. Adanya proses key-schedule yang menjamin setiap perubahan bit yang diproses mempengaruhi seluruh bit pesan tidak seperti fungsi hash MD yang mana prosesnya hanya mempengaruhi 2 bit di setiap prosesnya. Serangan ini sering disebut dengan Dobbertin's differential attack. Semakin banyak pengaruh terhadap perubahan bit, maka semakin besar upaya yang harus dilakukan penyerang ketika melakukan kriptnalisis.
5. Perkalian register b di setiap putaran menyebabkan input S-Box menjadi berbeda, disamping itu terdapat perlakuan terhadap ketiga register yang digunakan sehingga seluruh register mengalami perubahan dan kriptnalis harus melakukan upaya yang lebih besar dibandingkan dengan algoritma hash sebelumnya. Hal ini dapat dilihat pada gambar 3.3 dimana terdapat perkalian register b, pengurangan pada register a dengan hasil proses *even* untuk S-Box genap, dan register c dilakukan proses xor dengan blok pesan X.
6. Adanya feedforward yang melindungi Tiger dari Preimage Attack. Feedforward diletakkan di bagian belakang dimana feedforward berfungsi untuk membaurkan hasil dengan mengkombinasikan nilai register paling awal dengan nilai register paling akhir. Hal ini lah yang nantinya menghindari Tiger dari serangan Preimage Attack yang menemukan sejumlah pesan untuk satu nilai hash yang sama.

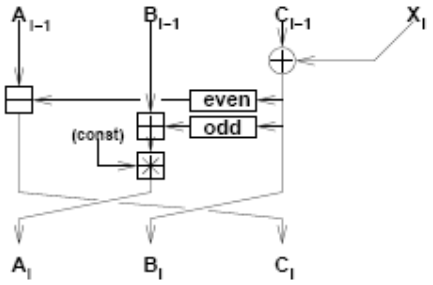


Fig. 1. The round function of Tiger

Gambar 3.3 Fungsi putaran pada Tiger

Disamping kelebihan tersebut, fungsi hash memiliki kelebihan yang belum dimiliki oleh fungsi hash sebelumnya yaitu kecepatan. Jika dilihat dari strukturnya yang rumit, menjadikan algoritma tiger menjadi algoritma yang lambat, namun pada kenyataanya algoritma tiger masih bisa menyaingi kecepatan SHA dalam melakukan proses hash. Hal ini disebabkan penggunaan kotak S-Box yang mempersingkat pendefinisian register ketika ingin menambah kerumitan dalam putaran algoritma tiger.

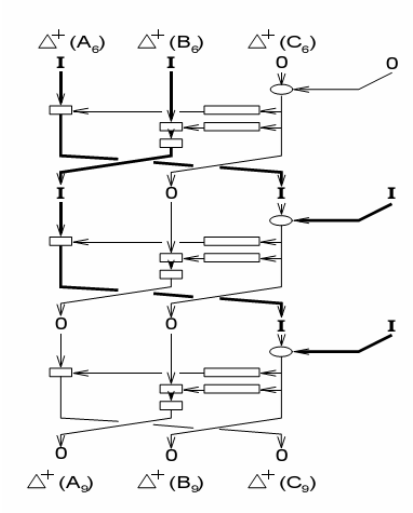
IV. PENGEMBANGAN

4.1 Analisis serangan yang mungkin untuk Tiger

Berdasarkan hasil analisis sebelumnya, diketahui bahwa algoritma Tiger memiliki struktur yang hampir sama dengan fungsi hash umumnya yaitu output yang dihasilkan cenderung tetap. Dengan kata lain algoritma ini sangat mungkin untuk ditemukan *collision*. Pada umumnya penyerangan lebih mudah dilakukan ketika panjang pesan bersifat tetap terlebih lagi jika nilai hasil hash pendek. Semakin pendek nilai hash, maka semakin sedikit waktu yang diperlukan untuk melakukan kriptnalisis. Selain itu algoritma tiger masih menggunakan struktur *blok chipper* sehingga dapat dilakukan penyerangan tiap bloknya. Dengan kata lain, penyerang dapat memecah tahap putaran tiger menjadi beberapa bagian dan mencoba tiap langkah untuk mendapatkan pola keluaran. Hal ini menyebabkan algoritma tiger juga masih tergolong rentan terkena serangan meskipun telah memiliki penanganan yang telah dibuat. Selain itu menggunakan blok chipper, penyerang dapat memaksakan memasukkan sejumlah pesan untuk mencari *collision* pada strukturnya. Kesamaan struktur algoritma tiger dengan fungsi hash lain yang sudah ditemukan *collision* membuat algoritma ini juga memiliki peluang yang besar untuk ditemukan *collision*.

Bisa dilihat pada gambar 4.1 dimana kesamaan struktur yang dimiliki Tiger yang menggunakan blok chipper. Terdapat beberapa kelemahan yang bisa dilihat dari struktur tersebut adalah penyerang dapat memecah proses putaran dan melakukan penyerangan terhadap proses tersebut. Hal ini memudahkan penyerang dalam melakukan kriptanalis. Disamping itu penyerang dapat

memaksa memasukkan pesan yang nantinya diproses bersama register. Hal ini memudahkan penyerang untuk mencari proses turunan nilai register dan bisa memasukkan berbagai pesan untuk dilakukan pencarian nilai register yang sama. Jika pesan tersebut telah mampu membuat suatu nilai register sama dengan pesan lain, maka register lain juga akan memiliki nilai yang sama sehingga sudah didapatkan collision pertama. Selain itu dari struktur sangat terlihat adanya alur proses yang jelas yaitu nilai akhir register dapat dicari perunutan prosesnya dan dilakukan penyerangan *meet-in-the-middle.* Untuk lebih jelas pada gambar 4.1 terlihat bahwa nilai C_8 dipengaruhi oleh nilai B_2. Dengan demikian sangat mudah mencari kelanjutan *collision* pada algoritma tiger ini.



Gambar 4.1 Struktur putaran Tiger

4.2 Perbaikan beberapa elemen Tiger

Berdasar penjelasan di atas, algoritma Tiger masih bisa dilakukan kriptnalisis terutama dilihat dari strukturnya. Dari analisis di atas maka algoritma tiger perlu adanya penambahan jumlah proses putaran pada algoritma ini yang awalnya ada 8, menjadi 16 putaran. Hal ini bertujuan untuk menghindari algoritma Tiger mengalami penyerangan *full collision.* Hal ini sangat penting dilakukan karena struktur Tiger yang berpola menyebabkan jika putaran 20 sudah mengalami collision, maka algoritma Tiger sudah dikatakan tidak aman karena hanya dengan menggunakan pola tersebut Tiger bisa didapatkan *full collision*-nya. Hal ini sempat menjadi permasalahan ketika disebutkan adanya penyerangan dengan karakteristik differential. Pengembangan yang akan dilakukan adalah melakuakan perulangan terhadap putaran menjadi 2 kali dengan perbedaan adanya persilangan antara posisi odd dan even. Hal ini bertujuan agar tidak bisa didaptkan pola yang sama untuk seluruh blok proses dan jika melakukan kriptnalisis diperlukan usaha yang berbeda setiap bloknya.

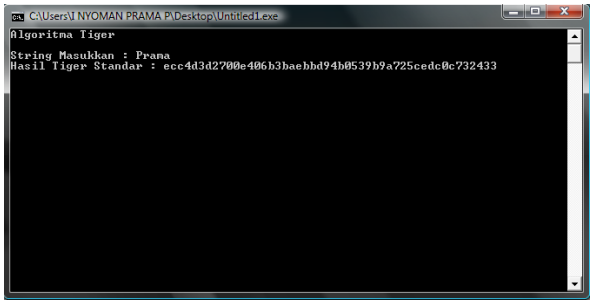
Perubahan kedua yang dilakukan adalah proses multiplikasi terhadap semua register dengan hasil modulo pesan dengan 8. Hal ini dilakukan karena algoritma Tiger saat ini belum mengubah seluruh bit secara penuh. Dengan adanya penambahan multplikasi terhadap register, dapat membuat perubahan pasti pada setiap bit register dan membuat perubahan pada seluruh bit register. Jika hanya memproses register dengan S-Box saja tentu akan menghasilkan nilai keluaran tetap untuk semua proses. Jika demikian halnya, maka penyerangan dengan memanfaatkan peluang munculnya nilai register yang berulang dapat terbaca oleh kriptnalis seperti yang terlihat pada gambar 4.1. Pada gambar terlihat bahwa register akan memiliki peluang memiliki nilai sama ketika tidak mengalami proses dengan pesan, dengan kata lain register yang diproses dengan menggunakan nilai yang tetap, akan menghasilkan nilai yang tetap juga dan menhasilkan suatu pola tertentu. Namun dengan adanya multiplikasi dengan konstanta yang bergantung pada pesan, hasil keluaran proses register juga kan berubah dan akan menghindari proses penyerangan *meet-in-the-middle* dimana jika penyerang ingin melakukan serangan terhadap suatu register, ia hanya perlu melakukan modifikasi terhadap turunan register sebelumnya. Namun dengan adanya modfikasi ini, penyerang tidak memiliki kebebasan seperti itu.

Perubahan yang ketiga adalah dilakukannya shift pada akhir proses masing-masing register. Hal ini bertujuan untuk mencegah adanya penyerangan dengan memanfaatkan output yang mudah terbaca oleh kriptanalis. Dengan adanya proses shift pada akhir proses tiap register, maka hasil keluaran sangat kecil peluangnya untuk membentuk pola. Dengan demikian kriptanalis yang misalkan telah mendapat collision pada salah satu register pada putaran pertama, harus mengubah hasil collision yang ia dapat di awal tadi jika melakukan penyerangan terhadap putaran yang kedua. Jadi dengan cara ini sangat kecil dilakukannya collision berantai.

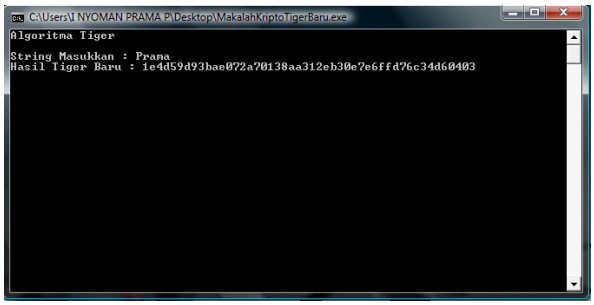
V. IMPLEMENTASI

Berikut ini adalah hasil implementasi fungsi hash tiger yang dibuat dalam bahasa C.

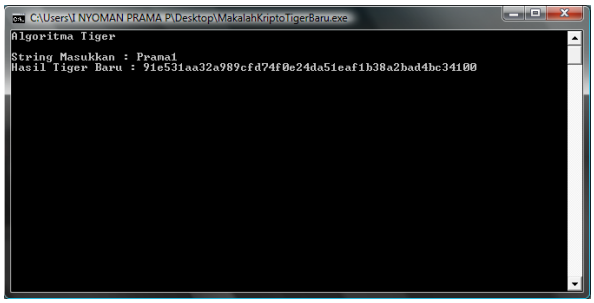
Pengujian 1



Gambar 5.1 Algoritma Standar Tiger

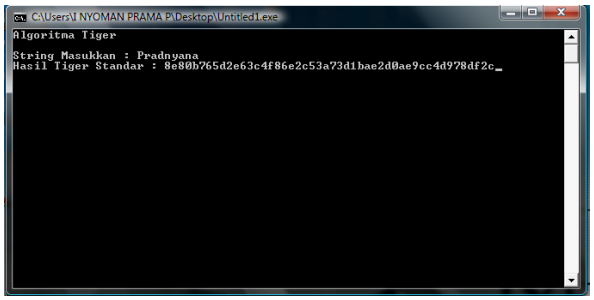


Gambar 5.2 Algoritma Tiger modifikasi

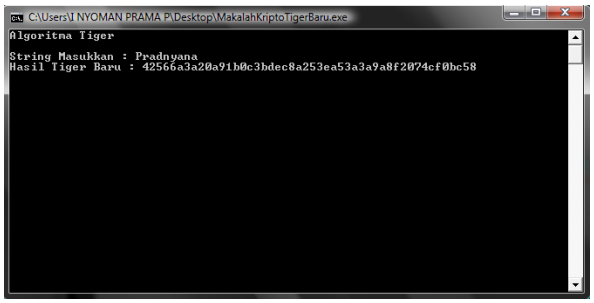


Gambar 5.3 Algoritma Tiger modifikasi

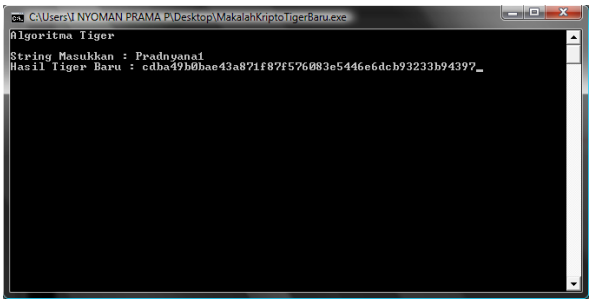
Pengujian 2



Gambar 5.4 Algoritma Standar Tiger



Gambar 5.5 Algoritma Tiger modifikasi



Gambar 5.6 Algoritma Tiger modifikasi

VI. PEMBAHASAN

Setelah dilakukan pengujian terhadap fungsi hash hasil modifikasi, terlihat bahwa Tiger hasil pengembangan memiliki nilai hasil hash yang berbeda namun memiliki struktur yang sama yaitu 192 bit. Terlihat juga bahwa ketika terjadi perubahan pada pesan masukkan, nilai hash yang dihasilkan jauh berbeda.

Meskipun sudah dilakukan perubahan, namun perubahan yang dilakukan masih belum sepenuhnya mampu menangani penyerangan. Algoritma ini sebaiknya terus dikembangkan mengingat perkembangan fasilitas teknologi yang semakin mendukung adanya penyerangan. Jika tidak disesuaikan dengan perkembangan tekonologi, maka akan sangat mudah untuk melakukan peyerangan iterasi terhadap algoritma tiger. Meskipun algoritma tiger masih perlu dikembangkan, algoritma tiger hingga saat ini masih dianjurkan untuk penggunaannya dengan aplikasi-aplikasi fungsi hash dan fungsi hash masih sangat aman untuk digunakan.

Selain itu perlu diketahui bahwa dengan ditemukannya *collision* pada penyerangan, masih belum dapat emngatakan bahwa fungsi hash tersebut tidak aman. Pada dasarnya Kriptnalisis hanya mampu menghasilkan *collision* namun tidak mampu mendapatkan kontrol terhadap pesan. Dengan kata lain, kriptanalis hanya mampu membuktikan adanya suatu *collision* namun tidak mampu memberikan suatu nilai pesan berarti. Penyerang tidak dapat memaksakan suatu pesan memiliki nilai hash yang sesuai dengan keinginan penyerang. Jadi meskipun penyerang telah mendapatkan collision pada fungsi hash, ia tetap tidak dapat memalsukan pesan asli dengan pesan palsu hasil collision karena pesan palsu tersebut bisa dipastikan tidak memiliki arti dan jika pesan asli digantikan dengan pesan palsu tersebut, penerima dapat langsung mengetahui bahwa pesan tersebut palsu.

VII. KESIMPULAN

Algoritma Tiger meskipun memiliki kekuatan yang lebih dibandingkan dengan fungsi hash sebelumnya, algoritma Tiger memiliki beberapa kelemahan terutama pada strukturnya.

Ada baiknya algoritma Tiger ini terus dikembangkan untuk mengantisispasi penyerangan dimana sudah semakin berkembang teknologi yang mendukung adanya penyerangan terutama dalam melakukan iterasi.

Jika kriptanalis berhasil menemukan collision pada algoritma Tiger ini, *message* hasil collision itu tidak dapat digunakan untuk memanipulasi pesan karena kriptnalisis pada dasarnya hanya mampu menemukan collision namun tidak memiliki kontrol terhadap pesan masukan.

REFERENCES

[RIN05] Munir, Rinaldi. (2005). *Diktat Kuliah IF5054 Kriptografi*. Bandung: Program Studi Teknik Informatika, Institut Teknologi Bandung.

[ROS96] Anderson, Ross. (1996). Proposal Tiger : A Fast New Hash Function. England : Cambridge University

[LAT06] Latuconsina, Roswan. (2006). Fungsi Hash Tiger.Bandung

[AGU09] Agung.(2009).Kriptografi. Jumat, 11 Mei 2012 http://the-jester-komputer.blogspot.com/

[ELI09] Biham,Eli.(2009) Jumat, 11 Mei 2012 http://www.cs.technion.ac.il/~biham/Reports/Tiger/

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, 11 Mei 2012

I Nyoman Prama Pradnyana
13509032