ANALISIS KINERJA ALGORITMA *HONEY ENCRYPTION* DAN ALGORITMA *BLOWFISH* PADA PROSES ENKRIPSI DAN DEKRIPSI

Sujacka Retno[(1)], Novia Hasdyna[(2)]
Program Studi Magister Teknik Informatika
Fakultas Ilmu Komputer dan Teknologi Informasi
Universitas Sumatera Utara
e-mail : [(1)]sujackaretno@students.usu.ac.id,
[(2)]novia_hasdyna@students.usu.ac.id

Abstrak

Algoritma *Honey Encryption* dan algoritma *Blowfish* merupakan dua buah algoritma kriptografi yang dapat digunakan dalam proses enkripsi dan dekripsi. *Honey Encryption* merupakan salah satu algoritma yang masih baru dalam ilmu kriptografi, untuk itu perlu dilakukan analisis kinerja algoritma tersebut pada proses enkripsi dan dekripsi. Berdasarkan studi perbandingan yang telah dilakukan diperoleh hasil bahwa jika ditinjau proses enkripsi dan dekripsinya, algoritma *Honey Encryption* jauh lebih efektif dan efisien dibandingkan dengan algoritma *Blowfish* dari segi keamanan dan tingkat kompleksitas enkripsi dan dekripsinya.

Kata Kunci : *Kriptografi, Algoritma Honey Encryption, Algoritma Blowfish, Enkripsi, Dekripsi.*

1. PENDAHULUAN

Perkembangan teknologi informasi telah menyebabkan perubahan dan cara pandang hidup manusia maupun suatu organisasi. Berbagai organisasi, perusahaan, atau pun pihak–pihak lain telah memanfaatkan teknologi basis data untuk menyimpan dan mengelola data organisasi atau perusahaannya. Dibutuhkan sebuah metode penyandian, ilmu sekaligus seni guna menjaga *file* yang disebut juga dengan kriptografi. Salah satu perangkat lunak kriptografi adalah *Pretty Good Privacy* (PGP) yang juga bisa digunakan secara online maupun offline. Selain dapat mengamankan file, perangkat lunak ini juga dapat memberikan tanda tangan digital (*digital signature*) yang mampu memenuhi tiga aspek keamanan yaitu integritas data, otentikasi, dan nirpenyangkalan.

Blowfish adalah cipher block berkunci yang didesain oleh Bruce Schneier pada tahun 1993 yang mencakup jumlah besar cipher dan

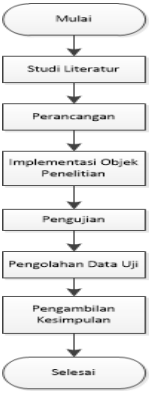
DOI: https://doi.org/10.29103/techsi.v10i2.858

enkripsi. *Blowfish* memberikan hasil enkripsi yang baik (sulit untuk dipecahkan) dan sampai saat ini belum ada kriptanalisis yang mengklaim telah dapat memecahkannya. Hal tersebut mungkin dikarenakan sistem enkripsi lain yang lebih banyak mendapat perhatian seperti sistem yang terbaru saat ini, *Honey Encryption*. Teknik *Honey Encryption* dikembangkan oleh Ari Juels, mantan kepala ilmuwan RSA, dan Thomas Ristenpart dari *University of Wisconsin*. *Honey Encryption* paling cocok dalam situasi di mana data terenkripsi diperoleh dari kata sandi. Untuk mengoptimalkan performa maksimal kedua algoritma (*Honey Encryption* dan *Blowfish*), ukuran data akan digunakan sebagai pembanding terhadap waktu proses algoritmA karena panjang data adalah sangat penting untuk aplikasi teks, dan perlu diketahui untuk aplikasi yang mendukung plaintext yang panjang seperti aplikasi email atau yang membatasi panjang plaintext seperti aplikasi SMS, manakah enkripsi yang lebih baik performanya.

Hasil dari perbandingan kedua algoritma dapat digunakan sebagai acuan data dua algoritma enkripsi yang populer saat ini, juga dapat digunakan programmer untuk memilih algoritma enkripsi mana yang sesuai dengan aplikasi yang dikembangkan, selain itu dapat juga digunakan sebagai bahan referensi penelitian yang berkaitan dengan algoritma enkripsi *Honey Encryption* dan *Blowfish*.

2. METODE PENELITIAN

Metode yang digunakan dalam penelitian ini meliputi studi literatur, perancangan, implementasi obyek penelitian, pengujian, pengolahan data uji, dan terakhir pengambilan kesimpulan.



Gambar 1. Metode Penelitian

DOI: https://doi.org/10.29103/techsi.v10i2.858

3. HASIL DAN PEMBAHASAN

3.1 Skema Enkripsi dan Dekripsi Algoritma *Honey Encryption*

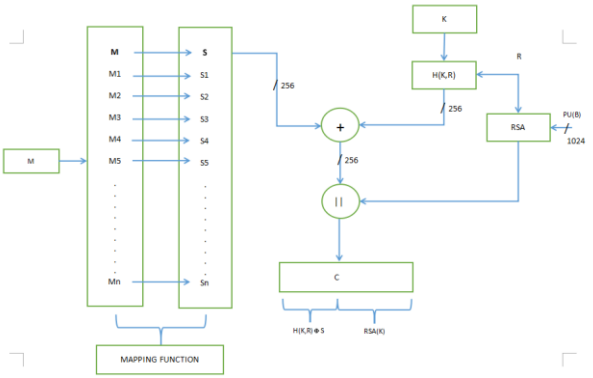
3.1.1 Enkripsi

Umpamakan A dan B adalah dua pihak yang berkomunikasi, di mana A ingin mengirim pesan M ke B. Dalam proses enkripsi, pengguna akan memberikan pesan (M) dan kunci simetrik (k). Pesan ditempatkan di ruang pesan yang akan dipetakan ke nilai *hash* dari pesan (S) yang dihasilkan menggunakan logika *SHA256*. Ruang juga berisi beberapa string valid yang dipilih secara acak (M1, M2, M3, ..) yang dipetakan ke nilai *seed* (S1, S2, S3,…).

Nilai kunci di*hash*kan menggunakan SHA256 dengan nilai (R) yang dihasilkan secara acak. Nilai *seed* ini dienkripsi dengan kunci public dari penerima B dan digabungkan dengan nilai *xor* dari nilai yang dipetakan pesan S dan nilai *hash* kunci dan (R).

$$C= H(K,R)\oplus S \;||\; RSA(Pub\ ,R)$$

Gambar di bawah ini mewakili keseluruhan proses enkripsi yang dilakukan oleh pengirim.

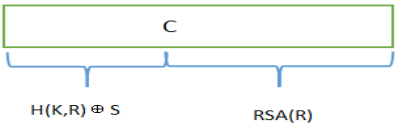


Gambar 2. Proses Enkripsi Honey Encryption

3.1.2 Dekripsi

Dalam *ciphertext* yang dihasilkan oleh pengirim pertama 256 bit memiliki nilai xor, nilai *hash* kunci dan string acak yang terenkripsi RSA. Pertama di bagian penerima, sebagian RSA diambil dari teks sandi dan didekripsi dengan kunci publiknyauntuk mendapatkan string acak R.

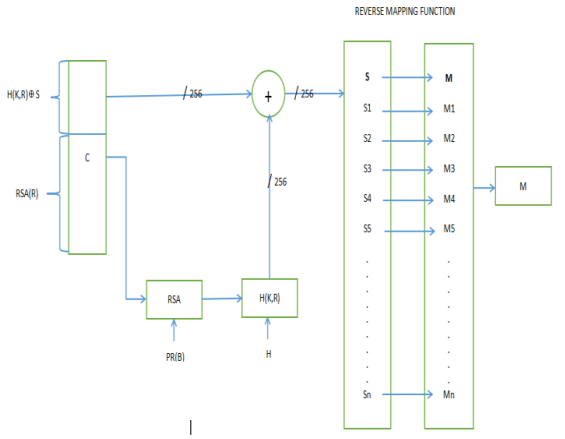
DOI: https://doi.org/10.29103/techsi.v10i2.858



Kemudian penerima akan meng*generate*kan nilai *hash* dengan menggunakan kunci simetrik K dan mendekripsi string acak R. Lalu, nilai yang dihasilkan adalah nilai *xor* dengan 256 bit pertama dari ciphertext. Hasilnya nanti akan menghasilkan nilai yang akan dipetakan secara *reverse* untuk mendapatkan pesan yang dihasilkan.

$$H(K,R)\oplus S\oplus H(K,RSA(PRb,RSA(Pub\ ,R)))=S$$

Gambar di bawah ini mewakili keseluruhan proses dekripsi pesan.



Gambar 3. Proses Dekripsi Algoritma *Honey Encryption*

3.1.3 Ketentuan Khusus

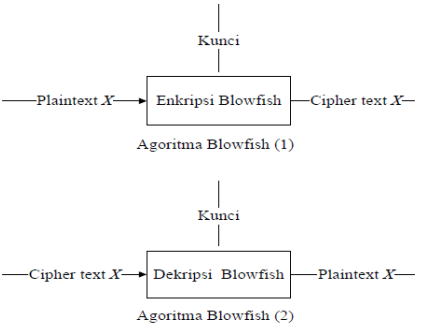
a. Nilai (R) harus mengandung minimal 10 karakter yang mencakup huruf besar, huruf kecil dan bilangan bulat.
b. Dalam perhitungan nilai *hash*, total jumlah putaran minimal 100000.
c. Kata sandi yang sering digunakan seperti '12345', 'password', dll digunakan untuk nilai kunci K1, K2, K3,… sehingga mudah untuk memanipulasi *cryptanalist*.

DOI: https://doi.org/10.29103/techsi.v10i2.858

d. Jika pengirim menggunakan kunci K yang berada dalam nilai kunci K1, k2, K3, .. maka nilai *seed* yang diperoleh dengan nilai K_i harus diperbarui dengan nilai yang diperoleh oleh K.

3.2 Skema Proses Enkripsi dan Dekripsi Algoritma *Blowfish*

Algoritma *Blowfish* menggunakan kunci simetris dalam proses enkripsi dan dekripsi. Misalkan kunci yang digunakan untuk proses enkripsi adalah ABC. Maka untuk melakukan dekripsi harus menggunakan kunci yang sama yaitu ABC, agar didapati hasil yang sama sebelum dienkripsi. Adapun skema proses enkripsi dan dekripsi pada algoritma Blowfish, dapat di lihat pada gambar dibawah ini.



Gambar 4. Proses Enkripsi dan Dekripsi Algoritma *Blowfish*

Adapun alur algoritma enkripsi dengan metode Blowfish dijelaskan sebagai berikut :

1. Bentuk inisial P-array sebanyak 18 buah (P0,P1,.....P17) masing-masing bernilai 32-bit. Array P terdiri dari delapan belas kunci 32-bit subkunci P0,P1,.....,P17.
2. Bentuk S-box sebanyak 4 buah masing-masing bernilai 32-bit yang memiliki masukan 256. Empat 32-bit S-box masing-masing mempunyai 256 masukan:

   S1,0,S1,1,.....,S1,255
   S2,0,S2,1,.....,S2,255
   S3,0,S3,1,.....,S3,255
   S4,0,S4,1,.....,S4,255
3. Plaintext yang akan dienkripsi diasumsikan sebagai masukan, Plaintext tersebut diambil sebanyak 64-bit, dan apabila kurang dari 64-bit maka kita tambahkan bit-nya, supaya dalam operasi nanti sesuai dengan datanya.

DOI: https://doi.org/10.29103/techsi.v10i2.858

4. Hasil pengambilan tadi dibagi 2, 32-bit pertama disebut XL, 32-bit yang kedua disebut XR.
5. Selanjutnya lakukan operasi XL = XL xor Pi dan XR = F(XL) xor XR
6. Hasil dari operrasi diatas ditukar XL menjadi XR dan XR menjadi XL.
7. Lakukan sebanyak 16 kali, perulangan yang ke-16 lakukan lagi proses penukaran XL dan XR.
8. Pada proses ke-17 lakukan operasi untuk XR = XR xor P16 dan XL = XL xor P17.
9. Proses terakhir, satukan kembali XL dan XR sehingga menjadi 64-bit kembali.

4. KESIMPULAN

1. Aplikasi pengamanan data menggunakan algoritma *Honey Encryption* dan *Blowfish* mempunyai dua teknik pembacaan yaitu teknik enkripsi (mengubah *file* asli menjadi *file* yang tidak dapat dibaca) dan teknik dekripsi (mengubah *file* yang tidak dapat dibaca menjadi *file* asli).
2. Aplikasi pengamanan algoritma *Honey Encryption* mempunyai kalimat sandi yang mempadukan sandi yang umum digunakan dan dipadukan dengan huruf besar dan kecil dibedakan, agar sulit ditebak oleh para *cryptanalyst*.
3. *Honey Encryption* lebih unggul dalam tingkat keamanan data dibandingkan dengan *Blowfish* karena memiliki tingkat yang lebih kompleks sehingga memungkinkan para *cryptanalyst* terkelabuhi dengan hasil yang didapatkan ketika mencoba menyerangnya.

DAFTAR PUSTAKA

Nahri Syeda Noorunnisa, and Dr. Khan Rahat Afreen, "Review on Honey Encryption Technique", IJSR, Volume 5 Issue 2, February 2016, 1683-1686.

Rifkie Primartha, "Penerapan Enkripsi dan Dekripsi File Menggunakan Algoritma Data Encryption Standard (DES)" JSI, Volume 3, No. 2, Oktober 2011, 371-387.

Santhi Baskaran, S.V.L Sarat Chandra, P.Venkatesh, E.Silambarasan, and M. Dinesh, "Implementation of Enhanced Honey Encryption for IoT Security", IJNTR, Volume-3, Issue-3, March 2017, 87-89.

DOI: https://doi.org/10.29103/techsi.v10i2.858

Taufiqur, Rahman Muhammad., Aryo Pinandito, dan Eko Sakti Pramukantoro, "Perbandingan Performansi Algoritme Kriptografi Advanced Encryption Standard (AES) dan Blowfish pada Text di Platform Android" Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Volume 1, No 12, Desember 2017, 1551-1559.

DOI: https://doi.org/10.29103/techsi.v10i2.858