PERANCANGAN DAN IMPLEMENTASI *SECURE CLOUD* DENGAN MENGGUNAKAN *DIFFIE-HELLMAN KEY EXCHANGE* DAN *TWOFISH CRYPTOGRAPHY ALGORITHM*

DESIGN AND IMPLEMENTATION SECURE CLOUD BY USING DIFFIE-HELLMAN KEY EXCHANGE AND TWOFISH CRYPTOGRAPHY ALGORITHM

Jainudin Anwar[1], Surya Michrandi Nasution[2], Tito Waluyo Purboyo[3]

Telkom University
Bandung, Indonesia jainudinanwar@students.telkomuniversity.ac.id[1],
michrandi@telkomuniversity.ac.id[2] titowaluyo@telkomuniversity.ac.id[3]

Abstrak

Perkembangan metode penyimpanan digital sekarang ini semakin beragam, salah satunya adalah metode penyimpanan berbasis *cloud* yang memberikan akses kepada penggunannya untuk menyimpan data di dalam Internet dengan kapasitas penyimpanan yang dapat disesuaikan dengan keinginan penggunanya. Namun, metode *cloud* ini memiliki kekurangan yaitu berkaitan dengan masalah keamanan data yang dapat dicuri. Pada penelitian ini akan akan dibahas mengenai keamanan data pada *cloud* dengan menggunakan kombinasi algoritma kriptografi *Twofish* dan pertukaran kunci *Diffie-Hellman*. Sistem yang dibangun adalah aplikasi berbasis *desktop* yang menyediakan konten untuk mengunggah dan mengunduh *file* dokumen dari *user*, yang didalam nya sudah terdapat proses enkripsi dan dekripsi dengan algoritma kriptografi *Twofish* serta pertukaran kunci user dengan *Diffie-Hellman*. Penelitian ini bertujuan untuk menganalisa performansi dari algoritma kriptografi *Twofish* pada keamanan *file* dokumen saat ada proses enkripsi dan dekripsi, *avalanche effect, resources*, waktu proses *Diffie-Hellman*

Kata kunci: *Cloud*, Kriptografi, Algoritma kriptografi *Twofish*, Pertukaran Kunci *Diffie-Hellman*

Abstract

The development of digital storage method is now more diverse, one of which is a cloud-based storage methods that provide access to the consumer to store the data on the Internet with a storage capacity that can be tailored to the desires of its users. However, this method has the disadvantage that cloud relating to data security issues that can be stolen.This research will be discussed regarding data security in cloud by using a combination of *Twofish* cryptographic algorithms and *Diffie-Hellman* key exchange. The system is built is a desktop-based application that provides content to upload and download files from the user document, which was already contained in the encryption and decryption process with *Twofish* cryptographic algorithms and key exchange with *Diffie-Hellman* user. This study aims to analyze the performance of *Twofish* cryptographic algorithms in the security document file when there is a process of encryption and decryption, avalanche effect, power consumption and delay.

Keyword: *Cloud, Cryptography, Twofish Cryptography Algorithm, Diffie-Hellman Key Exchange*

1. Pendahuluan

Sekarang ini kita pasti memiliki banyak data yang ada di komputer kita, baik itu berupa dokumen, gambar, film ataupun data lainnya, tentu hal ini membuat kita memerlukan kapasitas penyimpanan data yang besar. Dengan berkembang pesatnya teknologi pemrosesan dan penyimpanan serta sukses dari internet membuat biaya dalam *computing* semakin murah, kemampuan yang lebih tinggi, dan juga ketersediaan yang lebih terjamin [6].

Seiring dengan berkembangnya teknologi komputasi dan *cloud*, membuat semakin banyaknya informasi dari seorang individu maupun perusahaan, dengan semakin banyak pengguna dan informasi yang disimpan meningkat pula perhatian para pengguna *cloud* tentang keamanan dari sistem itu sendiri [4]. Gambar dibawah menunjukkan ancaman pada *Cloud Service* [4].

Untuk memastikan keamanan data di *cloud* dibutuhkan identifikasi dan analisis dari resiko dan langkah-langkah keamanan / teknik yang dapat diterapkan dalam setiap tahap dari siklus hidup data. Kelalaian dari salah satu tahapan, setidaknya dalam kasus data sensitif untuk organisasi, dapat menyebabkan kehilangan hal penting di organisasi [1].

Dari perihal diatas, maka pada tugas akhir ini membahas keamanan penyimpanan pada *cloud* untuk menjaga data agar tidak dicuri yaitu dengan menggunakan kombinasi dari enkripsi pada *file* yang akan disimpan di *cloud* [8]. menggunakan pertukaran kunci *Diffie-Hellman* dan algoritma *Twofish.*

2. Tinjauan Pustaka

2.1 Kriptografi

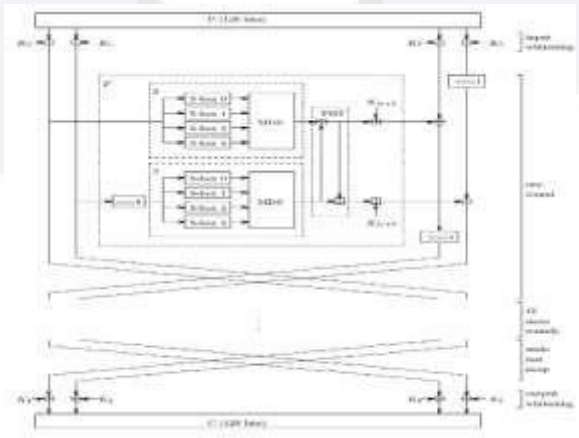
Kriptografi berasal dari Bahasa Yunani, yaitu ”*cryptos*” yang artinya rahasia, sedangkan ”*graphein*” yang berarti tulisan. Jadi kriptografi adalah ilmu dan seni untuk menjaga kerahasian pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya.

Dalam kriptografi pesan asli disebut *plaintext* serta terdapat istilah enkripsi dan dekripsi. Enkripsi adalah proses yang melakukan perubahan sebuah kode dari yang dapat dimengerti menjadi sebuah kode yang tidak dapat dimengerti (tidak terbaca). Enkripsi dapat diartikan sebagai kode atau *cipher*. Sedangkan, dekripsi adalah proses untuk mengembalikan informasi teracak menjadi bentuk aslinya dengan menggunakan algoritma yang sama pada saat mengenkripsi.

2.1.1 Algoritma Kriptografi Twofish

Twofish Cryptography Algorithm atau algoritma kriptografi *Twofish* adalah algoritma kriptografi yang di rancang oleh Bruce Scheiner dan rekan-rekannya untuk memenuhi kriteria dari NIST *(National Institute of Standards and Technology)* yang sedang mencari standar baru untuk algoritma kriptografi menggantikan standar yang lama yaitu algoritma kriptografi *DES*. Walaupun pada akhirnya algoritma kriptografi *AES Rijndael* lah yang menjadi standar baru algoritma kriptografi namun algoritma kriptografi *Twofish* tetap digunakan oleh sebagian besar kriptografer karena algoritma kriptografi ini menempati urutan kedua dalam kriteria NIST dibawah *AES Rijndael* [1].

Algoritma kriptografi *Twofish* merupakan jenis algoritma kriptografi *block chiper* 128 bit yaitu algoritma yang teknik enkripsi dan dekripsi nya pada blok-blok bit data, blok merupakan kumpulan bit dengan ukuran tetap yang terdapat pada data. Algoritma ini dapat menerima panjang variable kunci sebesar 256 bit. *Chiper* dari algoritma ini berasal dari 16 *round* jaringan *Feistel* dengan fungsi bijektif F yang dilanjutkan dengan empat *key-dependent* 8-*by-b-bit* S-*boxes*, satu fixed 4-by-4 *maximum distance separable matrix* dengan GF(2^8), satu *pseudo-hadamard transform*, satu rotasi *bitwise* dan satu desain *key schedule*. [7]



Gambar 2.1 Langkah-langkah di Twofish Cryptography Algorithm.[2,p-6]

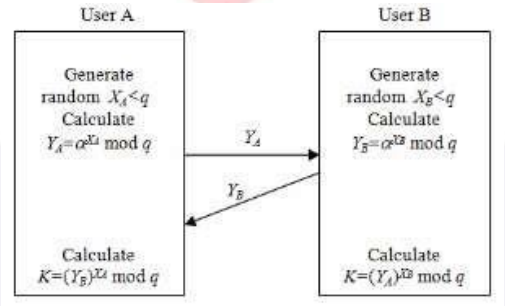
Algoritma kriptografi Twofish berstruktur *Feistel* 16-round dan terdapat penambahan *whitening* pada masukan dan keluaran. Untuk membuat suatu struktur *Feistel* murni perputaran dapat di pindah ke dalam fungsi F, tetapi

memerlukan suatu tambahan perputaran *word-word* yang tepay sebelum keluaran *whitening*. Plain teks dipecah menjadi empat *block word* seukuran 32 bit. Pada bagian langkah *whitening*, keempat *block word* di-XOR-kan dengan empat *word* dari kunci. Ini di ikuti dengan 16 kali perputaran dua *word* pada sisi kiri digunakan sebagai masukan kepada fungsi G (salah satunya diputarkan pada 8 bit pertama). Fungsi G terdiri dari empat byte-wide S-Box –key-*dependent*, yang diikuti oleh suatu langkah pencampuran linier berdasarkan suatu matrik MDS. Hasil kedua fungsi G dikombinasikan dengan menggunakan suatu *pseudo Hadamard Transform (PHT)*, dan ditambahkan dua *word* kunci. Kedua hasil ini kemudian di-XOR-kan ke dalam dua *word* pada sisi kanan *(*salah satunya diputarkan ke kanan 1 bit pertama, yang lainya di putarkan kekanan setelahnya). Bagian kiri dan kanan dibelah dua kemudian ditukar untuk perputaran berikutnya, dan empat empat *word* di-XOR-kan dengan lebih dari empat *word* kunci untuk menghasilkan chiper teks. [7]

Untuk langkah-langkah dekripsi dari algoritma kriptografi *Twofish* ini dapat dilakukan dengan cara sama seperti langkah-langkah enkripsi namun dengan membalik perintah dari *subkey*. *Subkey* digunakan sebagai kunci di dalam algoritma kriptografi Twofish. Kemudahan dalam dekripsi ini dikarenakan Algoritma kriptografi *Twofish* merupakan jenis algoritma berbasis jaringan *Feistel*. [9]

2.2 Pertukaran Kunci Diffie-Hellman

Pertukaran kunci *Diffie-Hellman* adalah metode pertukarang kunci rahasia pada komunikasi menggunakan kriptografi simetris. Kekuatan dari metode ini adalah pada sulitnya melakukan perhitungan logaritma diskrit antara pengirim dan penerima kunci [2].



Gambar 2.2 Langkah-langkah di pertukaran kunci *Diffie-Hellman*.

Tahap-tahap pertukaran kunci *Diffie-Hellman* adalah sebagai berikut :

1. Misalkan Alice dan Bob adalah pihak-pihak yang berkomunikasi. Mula-mula Alice dan Bob menyepakati 2 buah bilangan yang besar (sebaiknya prima) $\propto$ dan $\square$ sedemikian sehingga $\propto <$ $\square$ Nilai $\propto$ dan q tidak perlu rahasia,
2. Alice membangkitkan bilangan bulat acak X_A yang besar dan mengirim hasil perhitungan berikut kepada Bob,
$$Y\square = \propto^{XA} m\square\square q$$
3. Bob membangkitkan bilangan bulat acak X_B yang besar dan mengirim hasil perhitungan, berikut kepada Alice,
$$Y\square = \propto^{XB} m\square\square q$$
4. Alice lalu mendapatkan kunci dari,
$$K = Y\square^{X\square} m\square\square q$$
5. Bob mendapatkan kunci dari,
$$K = Y\square^{X\square} m\square\square q$$

2.3 Cloud Computing Cryptography

Cloud Computing atau komputasi awan adalah teknologi yang sangat bermanfaat bagi kehidupan di zaman sekarang, komputasi awan biasanya digunakan di dalam internet dan di dalam server pengatur pusat untuk melayani dan memelihara data dan aplikasi. Bahkan banyak aplikasi yang sudah bisa digunakan oleh pengguna melalui *cloud communications* tanpa harus instalasi terlebih dahulu. Selain itu, data dari pengguna dapat di akses dan dimanipulasi oleh komputer lain menggunakan jaringan internet. Meskipun akses dan pemakaian data dan aplikasi bersifat fleksibel yang sesuai den lingkungan komputasi awan. Masih banyak pertanyaan yang datang mengenai bagaimana caranya meningkatkan proteksi data dan aplikasi di dalam komputasi awan dari *hacker* dan penyusup.[8]

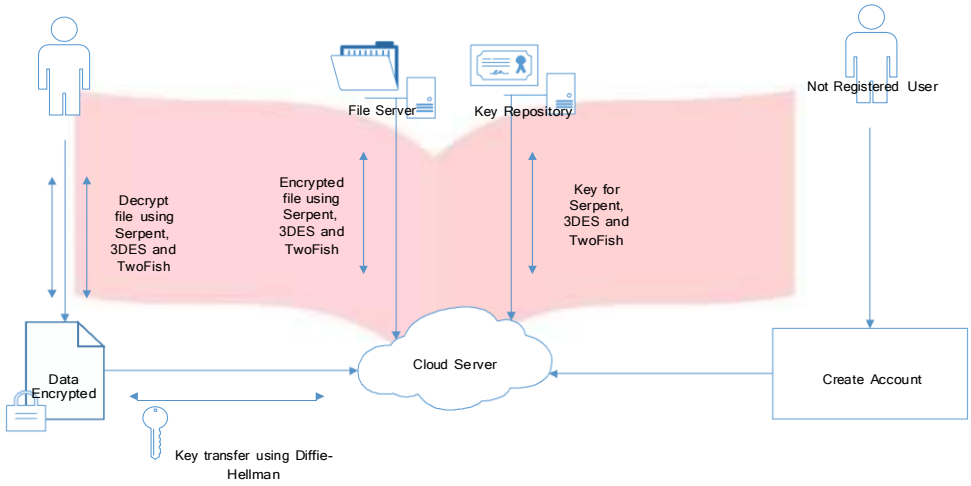
Untuk meningkatkan keamanan dari komputasi awan ini dapat dilakukan dengan menggunakan algoritma kriptografi yang memiliki tingkat keamanan yang tinggi untuk menjaga data dan aplikasi yang ada dengan cara

melakukan enkripsi dan dekripsi. Dapat pula menggunakan mekanisme "*key generation and management*" agar meningkatkan kemampuan pengamanan data dan aplikasi di lingkungan komputasi awan. [8].

3. Perancangan Sistem

3.1 Gambaran Umum Sistem

Gambaran umum sistem secara keseluruhan dapat dilihat pada gambar berikut :



Gambar 3.1 Gambaran Sistem Secara Umum

3.2 Perancangan Antarmuka

Perancangan antarmuka dari sistem yang dibangun terdiri dari :

Gambar 3.2 Rancangan Tampilan Antarmuka

4. Implementasi dan Pengujian Sistem

4.1 Implementasi Sistem

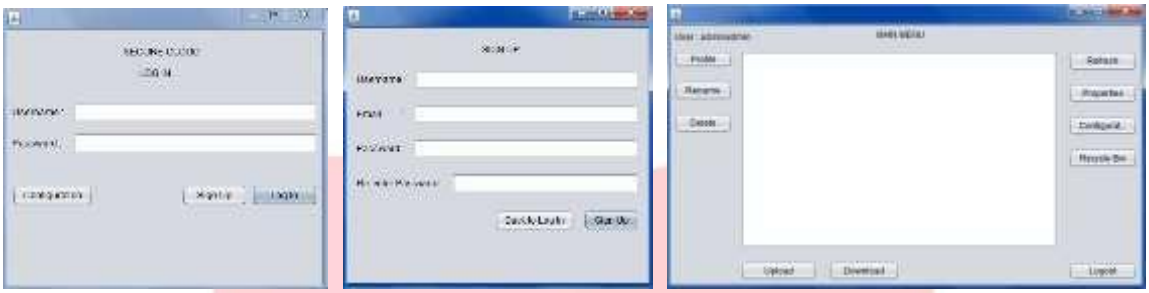
Dalam implementasi sistem enkripsi dan dekripsi file, langkah – langkah yang dilakukan ialah sebagai berikut:

1. Masuk kedalam aplikasi *Secure Cloud* dengan *log in* atau *sign up* terlebih dahulu.
2. Memilih *file* dokumen yang ingin di unggah ke dalam *Secure Cloud*
3. Saat memilih *file* dokumen *user* dapat menganti nama *file* tersebut
4. Saat mengunggah *file* dokumen maka secara langsung *file* tersebut terenkripsi
5. Saat mengunduh *file* dokumen maka secara langsung *file* tersebut terdekripsi
6. Menghitung waktu proses enkripsi *file* dokumen tersebut
7. Menghitung waktu proses dekripsi *file* dokumen tersebut
8. Menghitung nilai *avalanche effect* data biner dari *file* dokumen tersebut

9. Menganalisis pemakain daya yang terjadi saat proses enkripsi dan dekripsi dari *file* dokumen tersebut

4.2 Implementasi Antarmuka

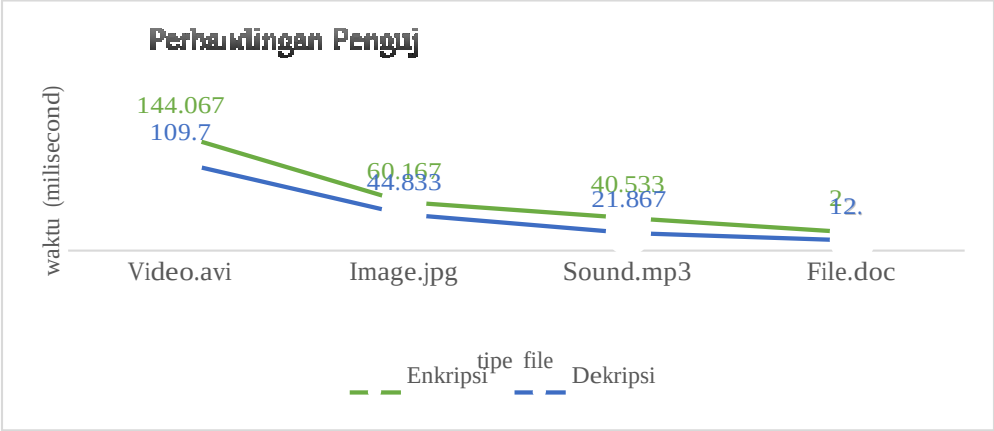
Implementasi antarmuka / *User Interface* merupakan tampilan dari aplikasi *file* yang akan menampilkan data *file* dokumen terenkripsi dan terdekripsi. Berikut merupakan tampilan antarmuka yang telah diimplementasikan :



Gambar 4.1 Implementasi Rancangan Antarmuka

4.3 Pengujian Sistem

4.3.1 Pengujian Waktu Enkripsi dan Dekripsi



Gambar 4.2 Diagram pengujian waktu enkripsi dan waktu dekripsi

Pada pengujian waktu proses enkripsi dan waktu proses dekripsi didapatkan hasil bahwa besar ukuran dari *file* yang diuji memiliki pengaruh dalam lama waktu proses enkripsi maupun lama waktu proses dekripsi. Untuk video.avi memiliki waktu proses enkripsi terlama yaitu 144.067 ms dan juga waktu proses dekripsi terlama yaitu 109.700 ms. Sedangkan file.doc memiliki waktu enkripsi tercepat yaitu 22.200 ms dan waktu dekripsi paling cepat yaitu 12.600 ms.

4.3.2 Pengujian Keamanan Sistem

Tabel 4.1 Tabel pengujian keamanan sistem

No	Nama File	Jumlah Bit Beda	Avalanche Effect %
1	File.docx	405665	50.0199%
2	Video.avi	4117510	50.0040%
3	Sound.mp3	741683	49.9819%
4	Image.jpg	1306524	49.9448%
Total Rata-Rata Avalanche Effect			**49.9877%**

Pada pengujian keamanan sistem atau *Avalanche Effect* didapatkan hasil bahwa nilai AE yang tertinggi didapat di file.docx dengan nilai 50.0199% dan nilai AE terendah didapat di image.jpg dengan nilai 49.9448%. total rata-rata AE yang didapatkan adalah 49.9877% ini sudah tergolong cukup bagus karena hampir mendekati setengah persen dari seratus persen dari total bit yang diujikan.

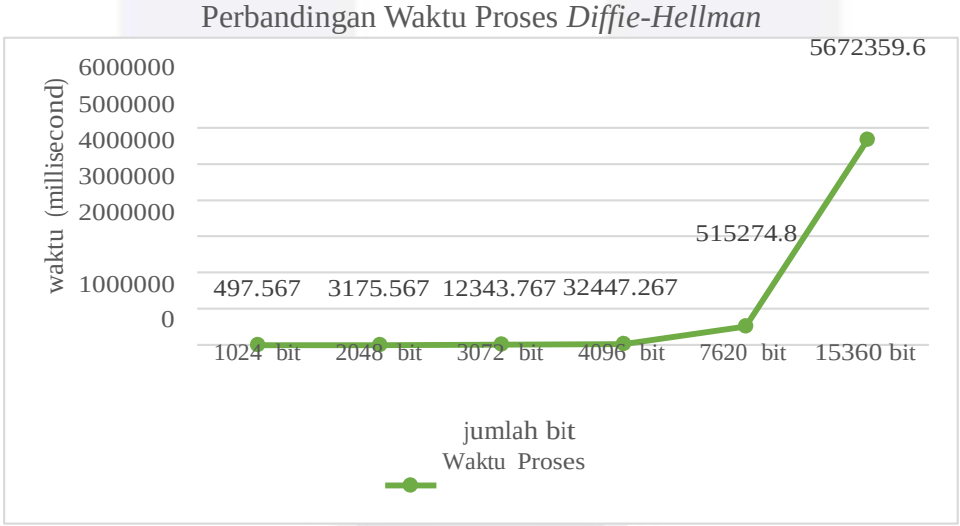
4.3.3 Pengujian Resources

Tabel 4.2 Tabel pengujian *resources*

No	Nama File	HS Enkripsi	HS Dekripsi	UH Enkripsi	UH Dekripsi
1	File.docx	162.500 MB	59.500 MB	25.167 MB	12.574MB
2	Video.avi	263.950 MB	182.417 MB	25.167 MB	115.740 MB
3	Sound.mp3	162.500 MB	74.483 MB	56.709 MB	23.422 MB
4	Image.jpg	232.033 MB	103.933 MB	65.133 MB	41.593 MB
Total Resources		**205.246 MB**	**105.083 MB**	**43.044 MB**	**48.332 MB**

Pada pengujian *resources* ini dilakukan dengan menguji daya pemakaian dengan menggunakan *Heap Size* dan *Used Heap*. *Heap Size* adalah pengaplikasian sejumlah memori untuk aplikasi *Java Virtual Machine*. *Used Heap* adalah jumlah memori yang sesungguhnya dipakai oleh proses aplikasi. Dari hasil pengujian didapatkan nilai HS enkripsi paling rendah adalah file.docx dan sound.mp3 dengan 162.500 MB dan nilai HS dekripsi paling rendah adalah file.docx dengan 59.500 MB. Untuk nilai UH enkripsi paling rendah adalah video.avi dengan 25.167 MB dan nilai UH dekripsi paling rendah adalah file.docx dengan 12.574MB. Perbandingan nilai HS dan nilai UH pada pengujian menunjukan tingkat pemakaian daya saat aplikasi di jalankan. Semakin nilai dari HS maupun UH rendah maka daya yang digunakan untuk pemrosesan semakin kecil.

4.3.4 Pengujian Diffie Hellmann



Gambar 4.3 Diagram Waktu Proses *Diffie-Hellman*

Pada pengujian waktu proses *Diffie-Hellmann* di ketahui total rata-rata waktu proses selama 497.567 ms untuk 1024 bit, 3175.567 ms untuk 2048 bit, 12343.767 ms untuk 3072 bit, dan 32447.267 ms untuk 4096 bit dan 515274.8 ms untuk 7620 bit. Hasil diatas menujukan bahwa dengan menggunakan *Diffie-Hellman* dengan kunci 1024 bit maka keamanannya setara dengan algoritma *3DES* 2 kunci, jika *Diffie-Hellman* dengan kunci 2048 bit maka keamanannya setara dengan *3DES* 3 kunci, jika *Diffie-Hellman* dengan kunci 3072 bit maka keamanannya setara dengan *AES*-128 bit, jika *Diffie-Hellman* dengan kunci 7680 bit maka keamanannya setara dengan *AES*-182 bit, dan jika *Diffie-Hellman* dengan kunci 15360 bit maka keamanannya setara dengan *AES*-256 bit.

4.3.5 Pengujian Big-O Notation

Pengujian *big-o notation* dilakukan untuk menguji kerumitan dari algoritma *Twofish* dan pertukaran kunci *Diffie-Hellman* berdasarkan *time complexity* dan *space complexity*. Algoritma *Twofish* memiliki *block size* dengan ukuran 128 bit. N direpresentasikan adalah *block*. Saat algoritma *Twofish* hanya memiliki satu N masukan, maka algoritma ini termasuk dalam jenis O(1). Namun, saat algoritma *Twofish* mendapatkan masukan lebih dari N *block*, maka algoritma *Twofish* termasuk dalam jenis O(N). Untuk *key scheduling* pada algoritma *Twofish* dikarenakan algoritma ini menggunakan salah satu *key length* diantara 128 bit, 192 bit, atau 256 bit maka termasuk dalam jenis O(1).

Untuk pertukaran kunci *Diffie-Hellman* termasuk dalam jenis O(N) dengan N adalah panjang kunci yang menjadi masukan di dalam *Diffie-Hellmann*. Sedangkan untuk proses pembuatan kuncinya, pertukaran kunci *Diffie-Hellman* memiliki beberaa tahapan. Tahap pertama adalah pembuatan bilangan prima yang memiliki tingkat kompleksitas $O((\log N)^4)$, N adalah ukuran dari bit *key*. Untuk pembuatan kunci privat memiliki tingkat kompleksitas O(log N), N adalah ukuran dari kunci privat. Dan untuk perhitungan dari kunci publik dan kunci privat termasuk dalam jenis O(1).

5. Kesimpulan

Kesimpulan yang dapat diambil dari penelitian Tugas Akhir ini adalah :

1. Hasil pengujian waktu enkripsi dan dekripsi didapatkan waktu proses enkripsi adalah 146.337 ms dan waktu proses dekripsi adalah 105.197 ms. Ini menunjukan bahwa algoritma *Twofish* memiliki waktu proses yang cepat dan tidak berbeda sangat jauh antara waktu enrkripsi dan dekripsi.
2. Hasil pengujian AE menggunakan key 256 bit adalah 49,9943 %, AE menggunakan key 192 bit adalah 49.9650%, dan AE menggunakan key 128 bit adalah 49.9686%. Ini menunjukan bahwa algoritma *Twofish* memiliki nilai AE yang tinggi karena mendekati 50%
3. Hasil pengujian *resources* didapatkan nilai HS enkripsi adalah 248.548 MB dan nilai HS dekripsinya adalah 148.974 MB, sedangkan nilai UH enkripsi 85.520 MB dan nilai UH adalah 71.723 MB. Ini menunjukan bahwa algoritma *Twofish* menggunakan resources lebih tinggi saat dekripsi daripada saat enkripsi.
4. Hasil pengujian waktu DH untuk 512 bit adalah 261.900 ms, 1024 bit adalah 497.567 ms, 2048 bit adalah 3175.567 ms, 3072 bit adalah 12343.767 ms, 4096 bit adalah 32447.267 ms, 7620 bit adalah 515274.8 ms, dan 15360 bit adalah 5672359.6 ms. untuk memenuhi kebutuhan keamanan data sekarang ini maka yang cocok untuk dijadikan pilihan adalah DH dengan panjag kunci 3072 bit yang kekuatannya setara AES-128.
5. Hasil pengujian *big-o notation* menujukan bahwa algoritma *Twofish* termasuk dalam tingkat kompleksitas O(N), N adalah jumlah blocksize yang dikerjakan algoritma *Twofish*. Sedangkan pertukaran kunci *Diffie-Hellman* termasuk dalam tingkat kompleksitas $O((\log N)^4)$, N adalah jumlah bit *key* untuk proses pembuatan kunci didalam *Diffie-Hellman*.

DAFTAR PUSTAKA

[1] B. Scheiner, J. Kesley, D. Withing, D. Wagner, C. Hall and N. Ferguson, "Twofish : A 128-Bit Block Chiper," First Advanced Encryption Standard (AES) Conference, 1998.

[2] D. W. and M. Hellman, ""New directions in cryptography"," IEEE Transactions on Information Theory, 1976.

[3] E. Barker, W. Barker, W. Burr, W. Pork and M. Smid, "Recommendation for Key Management - Part 1 : General (Revision 3)," NIST Special Publication 800-57, p. 64, July 2012.

[4] k. S. O., I. F. and A. O., "Cloud Computing Security Issues and Challenges," International Journal of Computer Network (IJCN), vol. 3, p. 248, 2011.

[5] Li. Na, "Research on Diffie-Hellman Key Exchange Protocol," Information Engineering Teaching and research section, 2010.

[6] M. Mircea, "Addressing Data Security in the Cloud," World Academy of Science, Engineering and Technology, vol. 6, 2012.

[7] M. Y. Soleh, "Studi Perbandingan Algoritma kunci simetris Serpent dan Twofish," Makalah IF3058 Kriptografi, 2011.

[8] O. K. Jasim, S. Abbas, E.-S. M. El-Horbaty and A.-B. M. Salem, "Cloud Computing Cryptography "State-of-theArt"," World Academy of Science, Engineering and Technology, vol. 7, 2013.

[9] S. L. Su, L. C. Wuu and J. W. Jhang, "A New 256-bits Block Chiper - Twofish256," International Conference on Computational & Experimental Engineering and Science, 2007.