Penyembunyian Data Untuk Seluruh Ekstensi File Menggunakan Kriptografi *Vernam Cipher* dan *Bit Shiffting*

Data Hiding for All Extention File using Cryptography Vernam Cipher and Bit Shiffting

Christy Atika Sari[1], Eko Hari Rachmawanto[2], Danang Wahyu Utomo[3], Ramadhan Rakhmat Sani[4]

[1,2,3,4]Jurusan Teknik Informatika, Universitas Dian Nuswantoro

Jl. Imam Bonjol No. 207, Telp. (+6224) 3517261, Semarang 50131, Jawa Tengah

e-mail: [1]atika.sari@dsn.dinus.ac.id, [2]eko.hari@dsn.dinus.ac.id, [3]danang.wu@dsn.dinus.ac.id, [4]ramadhan_rs@dsn.dinus.ac.id

Abstrak

Kriptografi sebagai salah satu cabang ilmu yang dapat digunakan untuk mengamankan data hingga saat ini terus dikembangkan melalui berbagai algoritma. Beberapa penelitian terkait mengenai kriptografi masih mengguankan media berupa teks saja, image saja, maupun file tertentu saja. Pada penelitian ini akan digunakan media berupa seluruh jenis file sebagai media inputan. Adapun algoritma yang dignuakan yaitu Vernam cipher dan Bit shiffting. Kedua algoritma ini dikenal cepat, mudah dan aman untuk digunakan. Percobaan yang dilakukan menggunakan 30 file berbeda ukuran maupun jenis file serta telah diuji melalui aplikasi yang dibangun dengan Visual Basic telah menghasilkan proses enkripsi dan dekripsi data yang berjalan dengan baik. File hasil enkripsi dapat dibuka dengan kunci yang telah ditetapkan dan tidak mengalami kerusakan, dan sebaliknya untuk proses dekripsi data juga demikian. Hasil percobaan menggunakan sampling file berukuran 1 kb hingga 24000 kb, dimana waktu terlama untuk mengenkripsi file yaitu 28,661 detik dan untuk proses dekripsi terlama membutuhkan waktu 27,222 detik.

Kata kunci*—Vernam cipher, Bit shiffting,* Kriptografi, File

Abstract

Cryptography as a branch of science that can be used to secure the data to date continue to be developed through various algorithms. Several studies have linked the media mengguankan cryptography still be text only, image only, or only certain files. This research will use an entire media file types as input media. The algorithm dignuakan namely Vernam cipher and Bit shiffting. Both of these algorithms are known for quick, easy and safe to use. Experiments were performed using 30 different file sizes and file types and has been tested through application built with Visual Basic has produced data encryption and decryption process is going well. File encryption result can be opened with a key that has been set and is not damaged, and vice versa for data decryption process, too. The results of the experiment using a sampling file size of 1 kb to 24 000 kb, where the longest time to encrypt a file that is 28.661 seconds and the longest for the decryption process takes 27.222 seconds.

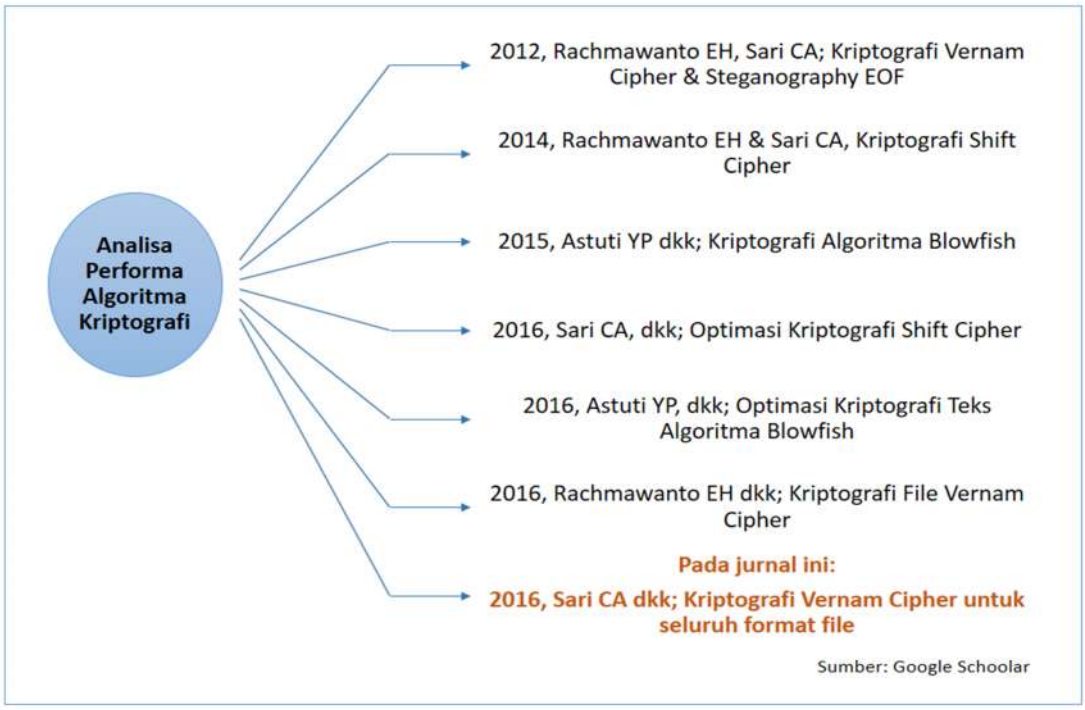
Keywords*—Vernam cipher, Bit shiffting*, *Cryptography*, File

1. PENDAHULUAN

Saat ini, penggunaan data oleh orang yang tidak berwenang semakin marak terjadi melalui media daring. Beberapa bentuk *cyber crime* yang dapat dilakukan antara lain *carding*, *cracking, hacking, defacing, spamming, physing, data leakage, data forgery* serta ilegal konten. Sebagai contoh yaitu pemalsuan dokumen *(data forgery)* pada media daring. Hal ini disebabkan pemilik data tidak mengetahui cara atau teknik yang dapat digunakan untuk melindungi data pribadi.

Terdapat sejumlah teknik yang dapat digunakan untuk mengamankan data, antara lain: *watermarking*, steganografi dan kriptografi. Ketiga teknik tersebut mempunyai kesamaan operasi yaitu operasi untuk menyembunyikan data dan operasi untuk mengekstraksi kembali data yang semula telah disembunyikan. Menurut Cheddad [1], perlu adanya file asli maupun kunci yang dapat digunakan untuk mengoperasikan ketiga teknik diatas. Salah satu yang mudah dana man untuk digunakan yaitu kriptografi. Hal ini dikarenkan algoritma kriptografi yang digunakan dapat memuat kunci yang berbeda sehingga lebih aman dan dapat mengecoh pihak yang tidak berkepentingan.

Adapun beberapa penelitian terkait dengan kriptografi telah berhasil dilakukan oleh beberapa peneliti pada Gambar 1.



Gambar 1 Penelitian Terkait dengan Kriptografi *Cipher*

Dapat dilihat pada Gambar 1 diatas bahwa penelitian mengenai kriptografi *cipher* sejak tahun 2012 telah dilakukan oleh Rachamwanto dkk untuk menanalisa algoritma kriptografi antara lain: Kriptografi *Vernam cipher* yang dikombinasikan dengan teknik Steganografi *End Of File* [2], penggunaan Kriptografi *Shift Cipher* melalui aplikasi untuk menguji kehandalan [3] dan *Shift Cipher* [4] untuk semua ekstensi file juga telah dioptimasi dengan hasil yang baik, sedangkan Astuti [5] menggunakan Algoritma *Blowfish* untuk mengamankan *password* dan Algoritma

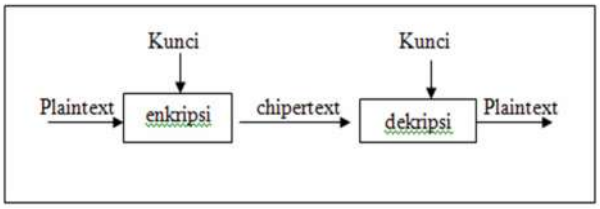
Blowfish juga telah dioptimasi [6] dengan melalukan perubahan kunci berupa heksadesimal maupun string untuk lebih mengamankan *password*, sehingga pada makalah ini diharapkan ada hasil yang maksimal dari penggunaan Algoritma Kriptografi *Vernam cipher* yang dikombinasikan dengan *Bit shiffting* untuk menyandikan seluruh ekstensi File.

2. METODE PENELITIAN

2.1 Kriptografi

2.1.1 Sejarah Kriptografi

Kriptografi berasal dari bahasa Yunani yaitu *kriptos* yang berarti tersembunyi dan *graphein* yang berarti tulisan [7], sehingga kriptografi diartikan sebagai ilmu yang digunakan untuk menyembunyikan pesan. Teknik ini sudah dikenal lebih dari 3000 tahun yang lalu dimana pertama kali digunakan oleh bangsa Sparta dari Yunani dalam bidang militer.



Gambar 2 Alur Kerja Kriptografi

Berdasarkan Gambar 1, terdapat beberapa istilah penting untuk melakukan proses kriptografi. *Plaintext* yaitu data asli yang akan disandikan menjadi bentuk lain yang tidak diletahui oleh orang lain. Bentuk *plaintext* beraneka ragam berdasarkan media yang digunakan antara lain: teks, gambar, suara, video maupun IP protokol. Untuk mengubah menjadi bentuk lain plainteks memerlukan sebuah kunci. Kunci tersebut dapat berbentuk angka maupun tulisan. Media kriptografi yang dapat digunakan hingga saat ini dapat dibagi menjadi 5 yaitu teks, gambar, audio, video dan protokol IP.

2.1.2 Tujuan Kriptografi

Secara umum terdapat 5 tujuan dilakukannya kriptografi, antara lain: (a) kerahasiaan yaitu informasi hanya dapat diakses oleh pihak yang berhak sehingga kerahasiaan harus dijaga; (b) otentikasi, dalam hal ini terdapat dua jenis otentikasi: otentikasi pesan berarti pesan yang diterima harus sama seperti pesan yang dikirimkam sedangkan otentikasi entitas *user* berarti pihak yang diajak berkomunikasi merupakan pihak yang benar-benar dikehendaki; (c) integritas merupakan keaslian pesan yang dikirim, (d) anti penolakan merupakan bukti bahwa seseorang telah mengirimkan pesan, (e) ketersediaan yaitu adanya sumber daya dari sistem komputer untuk mengakses oleh pihak yang berhak pada saat dibutuhkan.

2.1.3 Jenis-jenis Kriptografi

Kriptografi berdasarkan jenis kunci yang digunakan dapat digolongkan menjadi 3 yaitu kriptografi kunci simetris, kriptografi kunci asimetris dan kriptografi kunci *hybrid* (gabungan dari simetris dan asimetris). Pertama, kriptografi kunci simetris yaitu kriptografi yang dalam operasi enkripsi dan dekripsnya menggunakan kunci yang sama, dikenal sebagai *private key*. Contoh kriptografi kunci simetris yaitu DES *(Data Encyption Standard)*, 3DES, IDEA, *Blowfish, Twofish, Shift Cipher, Hill Cipher,*

Vernam cipher dan AES *(Advanced Encryption Standard)*. Kedua, kriptografi kunci asimetris yaitu kriptografi yang dalam operasi enkripsi dan dekripsinya menggunakan 2 buah kunci berbeda yang disebut dengan kunci privat dan kunci publik. Contoh dari kriptografi kunci asimetris yaitu RSA *(Riverst Shamir Adleman)*, DSA *(Digial Signature Algorithm)*, ECC *(Elliptic Curve Cryptography)*, DH *(Deffie Hellman)* dan *El Gamal*. Ketiga, kriptografi kunci gabungan simetris dan asimetris yaitu kriptografi yang menggunakan model persetujuan dari kedua belah pihak baik pengirim maupun penerima, dimana *session key* digunakan untuk mengenkripsi percakapan maupun mengenkripsi pertukaran data yang terjadi. Dalam hal ini setiap *session key* hanya dapat digunakan satu kali saja sehingga untuk sesi selanjutnya harus dibuat *session key* yang baru.

Berdasarkan kemunculannya [8], kriptografi digolongkan menjadi 2 yaitu kriptografi klasik dan kriptografi modern. Kriptografi klasik muncul pertama kali dan digunakan pada saat Perang Dunia II. Kriptografi klasik berbasis pada karakter dengan model operasi permutasi dan transposisi; dan biasanya merupakan kriptografi kunci simteris misalnya *Caesar Cipher*. Sedangkan kriptografi kunci modern yaitu kriptografi yang dibuat sedemikian rupa sehingga kompleks dan lebih sulit untuk dipecahkan disbanding dengan kriptografi kunci klasik umumnya. Jenis operasi pada kriptografi modern menggunakan mode bit, dimana semua data (plainteks, *cipher*teks, dan kunci) dinyatakan dalam string bit biner yaitu 0 dan 1. Rangkaian bit tersebut kemudian dipecah dalam blok-blok bit yang ditulis dalam berbagai cara bergantung pada panjang blok, misalnya menggunakan *padding bits* atau dengan mengubah ke bentuk heksadesimal.

2.2 Algoritma *Vernam cipher*

Vernam cipher adalah jenis algoritma enkripsi simetri. *Vernam cipher* dapat dibuat sangat cepat sekali, jauh lebih cepat dibandingkan dengan algoritma *block cipher* yang manapun. Algoritma *block cipher* secara umum digunakan untuk unit *plaintext* yang besar sedangkan *stream cipher* digunakan untuk blok data yang lebih kecil, biasanya ukuran bit. Proses enkripsi terhadap *plaintext* tertentu dengan algoritma *block cipher* akan menghasilkan *ciphertext* yang sama jika kunci yang sama digunakan [9]. Dengan *stream cipher*, transformasi dari unit *plaintext* yang lebih kecil ini berbeda antara satu dengan lainnya, tergantung pada kapan unit tersebut ditemukan selama proses enkripsi.

Satu *vernam cipher* menghasilkan apa yang disebut suatu *keystream* (suatu barisan bit yang digunakan sebagai kunci). Proses enkripsi dicapai dengan menggabungkan *keystream* dengan *plaintext* biasanya dengan operasi *bitwise XOR* [10]. Pembentukan *keystream* dapat dibuat independen terhadap *plaintext* dan *ciphertext*, menghasilkan *synchronous stream cipher*, atau dapat dibuat tergantung pada data dan enkripsinya, dalam hal mana *stream cipher* disebut sebagai *self-synchronizing*. Kebanyakan bentuk *stream cipher* adalah *synchronous stream cipher*.

Konsentrasi dalam *stream ciphers* pada umumnya berkaitan dengan sifat sifat teoritis yang menarik dari *one-time pad*. Suatu one-time pad, kadang-kadang disebut *Vernam cipher*, menggunakan sebuah string dari bit yang dihasilkan murni secara *random* (Kromodimoeljo, 2009). *Keystream* memiliki panjang sama dengan pesan *plaintext*; string random digabungkan dengan menggunakan *bitwise* XOR dengan *plaintext* untuk menghasilkan *ciphertext*. Karena *keystream* seluruhnya adalah random, walaupun dengan sumber daya komputasi tak terbatas seseorang hanya dapat menduga

plaintext jika melihat *ciphertext*. Metode *cipher* seperti ini disebut memberikan kerahasiaan yang sempurna (*perfect secrecy*). Metode *vernam cipher* yang umum digunakan adalah RC4. Satu hal yang menarik bahwa mode operasi tertentu dari suatu *block cipher* dapat mentransformasikan secara efektif hasil operasi tersebut ke dalam satu *keystream* generator dan dalam hal ini, *block cipher* apa saja dapat digunakan sebagai suatu *stream cipher*; seperti dalam DES, CFB atau OFB. Akan tetapi, *vernam ciphers* dengan desain khusus biasanya jauh lebih cepat.

*Cipher*teks diperoleh dengan melakukan penjumlahan *modulo* 2 satu bit plainteks dengan satu bit kunci:

$$C_i = (P_i + K_i)\ mod\ 26 \quad (1)$$

Dimana, P_i adalah bit plainteks, K_i adalah bit kunci, dan C_i adalah bit *cipher*teks. Plainteks diperoleh dengan melakukan penjumlahan *modulo* 2 satu bit *cipher*teks dengan satu bit kunci:

$$C_i = (P_i - K_i)\ mod\ 26 \quad (2)$$

Aliran-bit-kunci dibangkitkan dari sebuah pembangkit yang dinamakan pembangkit aliran-bit-kunci (*keystream generator*). Aliran-bit-kunci (sering dinamakan *running key*) di-XOR-kan dengan aliran bit-bit plainteks, p_1, p_2, …, p_i, untuk menghasilkan aliran bit-bit *cipher*teks:

$$C_i = P_i \oplus K_i \quad (3)$$

Di sisi penerima, bit-bit *cipher*teks di-XOR-kan dengan aliran-bit-kunci yang sama untuk menghasilkan bit-bit plainteks:

$$P_i = C_i \oplus K_i (4)$$

Merancang pembangkit bit-aliran-kunci yang bagus cukup sulit karena membutuhkan pengujian statistik untuk menjamin bahwa keluaran dari pembangkit tersebut sangat mendekati barisan acak yang sebenarnya.

2.3 Algoritma *Bit shiffting*

Shift cipher merupakan salah satu bentuk kriptografi klasik yang masih digunakan untuk mengamankan data. *Shift cipher* bekerja dengan menggeser plainteks sejauh yang diinginkan oleh pengguna, dengan maksimal pergeseran yaitu 26 [3]. Dalam penggunaannya, *shift cipher* menggunakan perhitungan modulo 26 dan kunci yang digunakan untuk proses enkripsi sama dengan proses dekripsi.

Shift *cipher* digunakan sejak jaman dahulu, tepaatnya saat pemerintahan Romawi Julius Caesar. Teknik ini merupkakan salah satu subtitusi *cipher*. Shift *cipher* yang merupakan generalisasi dari *Caesar cipher*, tidak membatasi pergeseran kunci sebanyak tiga huruf saja. *Shift cipher* menggunakan 26 kunci pergeseran sehingga lebih aman dibanding *Caesar Cipher*. Teknik ini menggunakan sisa bagi dari perhitungan yang dilakukan menggunakan proses penyandian menggunakan operasi modulo 26. Plainteks disimbolkan dengan "P" sedangkan *cipher*teks disimbolkan dengan "C" dimana kunci disimbolkan dengan "K", sehingga didapatkan rumus enkripsi:

$$\mathrm{C} = \mathrm{E(P)} = (\mathrm{P} + \mathrm{K}) \mathrm{mod}\ (26)\ (1)$$

Sedangkan rumus enkripsi adalah sebagai berikut:

$$\mathrm{P} = \mathrm{D(C)} = (\mathrm{C} - \mathrm{K}) \mathrm{mod}\ (26) (2)$$

Dalam proses penyandian, tambahkan huruf yang akan disandikan dengan kunci sehingga akan diperoleh huruf sesuai alphabet sandi, sedangkan untuk mendekripsi dapat digunakan cara sebaliknya. Berikut ini merupakan contoh penggunaan *shift cipher*. Misal, plainteks: "UDINUS", bentuk plainteks yaitu 21 4 9 14 21 19, apabila

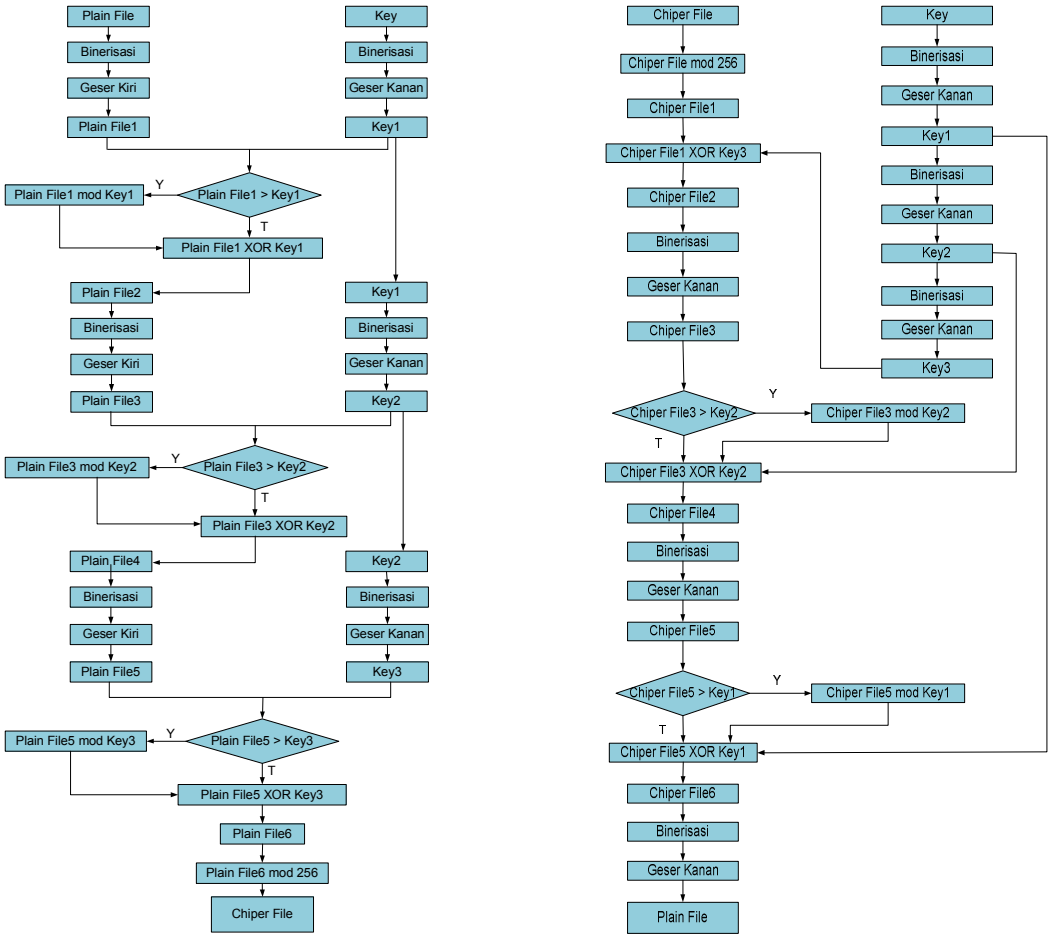
kunci yang digunakan yaitu 5 maka *cipher*teks menjadi 25 9 13 19 26 24 sehingga apabila ditransformasikan dalam huruf menjadi Z I N R Z W.

2.4 *Flowchart* Proses Enkripsi dan Dekripsi File

Berikut ini merupakan *flowchart* dari proses enkripsi dan dekripsi menggunakan algoritma *Vernam cipher* dan *Bit shiffting*:

Berdasarkan Gambar 3, proses enkripsi *file* dapat dijabarkan sebagai berikut:

1. Menyiapkan *file* dan kunci yang akan digunakan. *Plainfile* kemudian dibinerkan dan digeser ke kiri satu kali, misal C maka akan menjadi B, yang kemudian disebut *Plainfile*1. Sedangkan pada kunci juga dilakukan proses binerisasi tetapi kunci digesser ke kanan satu kali dan hasil pergeseran tersebut menghasilkan *Key*1.
2. Apabila Plainfile 1 lebih besar dari pada Key1, maka akan dilakukan proses mod pada Plain File 1 terhadap Key1 dan kemudian Plainfile1 XOR Key1.
3. Apabila Plain File1 lebih kecil atau sama dengan Key1 maka proses yang akan dilakukan adalah XOR pada Plainfile1 terhadap Key1.
4. Hasil dari XOR Plainfile1 disebut Plainfile2. Plainfile2 kemudian dibinerkan dan digeser ke kiri, dan hasil pergeseran ini disebut Plainfile3. Sedangkan Key1 dibinerkan dan digeser ke kanan, hasil proses pergeeran ini disebuut Key2.
5. Selanjutnya, Plainfile 3 dan Key2 digunakan untuk proses pemilihan, apakah Plainfile3 lebih besar dari Key2, apabila benar maka akan dilakukan proses Plainfile3 mod Key2 kemudian Plainfile3 XOR Key2. Apabila Plainfile3 lebih kecil atau sama dengan Key2 maka akan dilakukan proses Plain File3 XOR Key2.
6. Hasil dari XOR Plainfile3 disebut Plainfile4. Plainfile4 kemudian dibinerkan dan digeser ke kiri, dan hasil pergeseran ini disebut Plain File5. Sedangkan Key2 dibinerkan dan digeser ke kanan, hasil proses pergeeran ini disebuut *Key*3.



Gambar 3 *Flowchart* Proses Enkripsi dan Dekripsi File

Berdasarkan Gambar 3, proses enkripsi *file* dapat dijabarkan sebagai berikut:

1. Menyiapkan *file* dan kunci yang akan digunakan. *Plainfile* kemudian dibinerkan dan digeser ke kiri satu kali, misal C maka akan menjadi B, yang kemudian disebut *Plainfile*1. Sedangkan pada kunci juga dilakukan proses binerisasi tetapi kunci digesser ke kanan satu kali dan hasil pergeseran tersebut menghasilkan *Key*1.
2. Apabila *Plainfile* 1 lebih besar dari pada *Key*1, maka akan dilakukan proses mod pada Plain File 1 terhadap *Key*1 dan kemudian *Plainfile*1 XOR *Key*1.
3. Apabila Plain File1 lebih kecil atau sama dengan *Key*1 maka proses yang akan dilakukan adalah XOR pada *Plainfile*1 terhadap *Key*1.
4. Hasil dari XOR *Plainfile*1 disebut *Plainfile*2. *Plainfile*2 kemudian dibinerkan dan digeser ke kiri, dan hasil pergeseran ini disebut *Plainfile*3. Sedangkan *Key*1 dibinerkan dan digeser ke kanan, hasil proses pergeeran ini disebuut *Key*2.
5. Selanjutnya, *Plainfile* 3 dan *Key*2 digunakan untuk proses pemilihan, apakah *Plainfile*3 lebih besar dari *Key*2, apabila benar maka akan dilakukan proses *Plainfile*3 mod *Key*2 kemudian *Plainfile*3 XOR *Key*2. Apabila *Plainfile*3 lebih kecil atau sama dengan *Key*2 maka akan dilakukan proses Plain File3 XOR *Key*2.
6. Hasil dari XOR *Plainfile*3 disebut *Plainfile*4. *Plainfile*4 kemudian dibinerkan dan digeser ke kiri, dan hasil pergeseran ini disebut Plain File5. Sedangkan *Key*2 dibinerkan dan digeser ke kanan, hasil proses pergeeran ini disebuut *Key*3.

7. Selanjutnya, *Plainfile*5 dan *Key*3 digunakan untuk proses pemilihan, apakah *Plainfile*5 lebih besar dari *Key*3, apabila benar maka akan dilakukan proses *Plainfile*5 mod *Key*3 kemudian Plain File5 XOR *Key*3.
8. Apabila *Plainfile*5 lebih kecil atau sama dengan *Key*3 maka akan dilakukan proses *Plainfile*5 XOR *Key*3. Hasil dari XOR *Plainfile*5 disebut *Plainfile*6, kemudian *Plainfile*6 di mod 256 sehingga dihasilkan *Chiper File*.

Sedangkan proses dekripsi *file* dapat dijabarkan sebagai berikut:

1. Menyiapkan *Chiper File* dan *Key* hasil proses enkripsi.
2. *Chiper File* di mod 256, hasil proses ini disebut *Chiper File*1. Kemudian *Key* dibinerkan dan digeser ke kanan 1 sehingga menghasilkan *Key*1, *Key*1 dibinerkan dan digeser ke kanan 1 sehingga menghasilkan *Key*2, dan *Key*2 dibinerkan dan digeser ke kanan 1 sehingga menghasilkan *Key*3.
3. *Key* 3 kemudian digunakan untuk melakukan proses XOR pada *Chiper File*1, yaitu *Chiper File*1 XOR *Key*3, sehingga dihasilkan *Chiper File*2. Setelah itu, *Chiper File*2 kemudian dibinerkan dan digeser 1 kali ke kanan sehingga menghasilkan *Chiper File*3.
4. Apabila *Chiper File*3 lebih besar dari *Key*2 maka *Chiper File*3 mod *Key*2 dan kemudian di XOR kan yaitu *Chiper File*3 XOR *Key*2 sehingga dihasilkan *Chiper File*4.
5. Apabila *Chiper File*3 tidak lebih besar dari *Key*2 maka hanya akan dilakukan proses XOR saja. Selanjutnya *Chiper File*4 dibinerkan dan digeser ke kanan 1 kali sehingga dihasilakn *Chiper File*5.
6. Apabila *Chiper File*5 lebih besar dari *Key*1 maka *Chiper File*5 mod *Key*1 dan kemudian di XOR kan yaitu *Chiper File*5 XOR *Key*1 sehingga dihasilkan *Chiper File*6.
7. Apabila *Chiper File*5 tidak lebih besar dari *Key*1 maka hanya akan dilakukan proses XOR saja. Selanjutnya *Chiper File*6 dibinerkan dan digeser ke kanan 1 kali sehingga dihasilkan *Plain File*.

3. HASIL DAN PEMBAHASAN

Dalam penelitian ini digunakan 30 file berbagai format dan ukuran file untuk menguji algoritma *Vernam cipher* dan *Bit shiffting* melalui aplikasi keamanan data yang telah dibuat dengan Visual Basic 6.0 berikut.

Tabel 1 File Yang Digunakan Untuk Eksperimen

Nama File	Ukuran File	Format File
Main	24,393 kb	Pak
gSuperjigsaw	12,834 kb	Exce
Jinggle	3,689 kb	Mp4
Latihan	2,115 kb	Rtf
Coba	1,804 kb	Vsdx
Kripto	1,383 kb	Pdf
Analisis	1,291 kb	Rar
Cipher	1,216 kb	Pptx
Masih	1,000 kb	Docx

gChromeSetup	965 kb	Exe
Hasil	675 kb	Pdf
Contoh	591 kb	Zip
ScreenShot	490 kb	Png
TV4	469 kb	Png
Pengesahan	411 kb	Psd
Parameter	265 kb	Pptx
Boat	252 kb	Tif
Data	152 kb	Xlsx
Unconfirmed	131 kb	Crdownload
Paper	122 kb	Doc
Sendi	93 kb	Vxdx
Kriptografi	66 kb	Pptx
Gamextazy	29 kb	Ico
Logo	12 kb	Png
Logoudinus	10 kb	Jpg
Icon3	8 kb	Ico
Install_props	7 kb	Xml
Shortcuts	6 kb	Htm
LSB	2 kb	M
Scrabble	2 kb	Save
2013	1 kb	Xls
Readme First!	1 kb	Txt
drm.xml	1 kb	Sig
VERSION	1 kb	Cfg

Berikut ini merupakan tampilan aplikasi kriptografi file dengan algoritma *Vernam cipher* dan *Bit shiffting*.

Gambar 4 Tampilan Awal Aplikasi Kriptografi *Vernam cipher* dan *Bit shiffting*

Gambar 5 Proses Enkripsi File

Tabel 2 Lama Waktu Eksekusi Proses Enkripsi Dengan Algoritma *Vernam cipher* dan *Bit shiffting*

Nama File	Ukuran File	Format File	Lama Proses Enkripsi
2013	1 kb	Xls	0,212 detik
Readme First!	1 kb	Txt	0,120 detik
drm.xml	1 kb	Sig	0,310 detik
VERSION	1 kb	Cfg	0,114 detik

Gambar 6 Proses Dekripsi File

Tabel 3 Lama Waktu Eksekusi Proses Dekripsi Dengan Algoritma *Vernam cipher* dan *Bit shiffting*

Nama File	Ukuran File	Format File	Lama Proses Dekripsi
2013	1 kb	Xls	0,117 detik
Readme First!	1 kb	Txt	0,418 detik
drm.xml	1 kb	Sig	0,225 detik
VERSION	1 kb	Cfg	0,171 detik

Tabel 4 Lama Waktu Eksekusi Proses Enkripsi Dekripsi Dengan Algoritma *Vernam cipher* dan *Bit shiffting* Menggunakan File Berbeda Format dan Ukuran File

Nama File	Ukuran File	Format File	Lama Proses Enkripsi	Lama Proses Dekripsi
Main	24,393 kb	Pak	28,661 detik	27,222 detik
gSuperjigsaw	12,834 kb	Exce	19,610 detik	20,884 detik
Jinggle	3,689 kb	Mp4	8,912 detik	8,612 detik
Latihan	2,115 kb	Rtf	7.911 detik	8,001 detik
Coba	1,804 kb	Vsdx	7,319 detik	7,210 detik
Kripto	1,383 kb	Pdf	6,872 detik	6,819 detik
Analisis	1,291 kb	Rar	6,421 detik	6,791 detik
Cipher	1,216 kb	Pptx	6,237 detik	6,119 detik
Masih	1,000 kb	Docx	6,001 detik	5,905 detik
gChromeSetup	965 kb	Exe	5,801 detik	5,643 detik
Hasil	675 kb	Pdf	4,714 detik	4,409 detik
Contoh	591 kb	Zip	4,212 detik	4,405 detik

4. KESIMPULAN

Berdasarkan percobaan yang telah dilakukan menggunakan gabungan algoritma *Vernam cipher* dan *Bit shiffting* dapat disimpulkan bahwa:

- Gabungan algoritma ini berhasil melakukan proses enkripsi dan dekripsi dengan baik
- Proses enkripsi file dengan menggunakan sampling file berukuran 1 kb membutuhkan waktu eksekusi 0,114 detik dan paling lama 0,310 detik, sedangkan proses dekripsi file dengan ukuran file yang sama yaitu 1 kb tercepat yaitu 0,117 detik dan terlama 0,418 detik.
- Dengan menggunakan file berukuran besar yaitu berkisar antara 500 kb sampai 24000 kb, gabungan algoritma *Vernam cipher* dan *Bit shiffting* baik dalam proses enkripsi maupun dekripsi membutuhkan waktu antara 4 sampai dengan 28 detik.
- Gabungan *Vernam cipher* dan *Bit shiffting* terbukti dapat melakukan proses enkripsi dan dekripsi menggunakan seluruh ekstensi file dengan sampling data percobaan pada Tabel 1.

5. SARAN

Adapun saran untuk penelitian lanjutan yaitu mengkombinasikan kriptografi dengan teknik watermarking maupun steganografi dengan media berupa IP address untuk keamanan jaringan.

DAFTAR PUSTAKA

[1] A. Cheddad, J. Condell, K. Curran and P. Mc Kevitt , "Digital Image Steganography: Survey and Analysis of Current Methods," *International Journal of Signal Processing,* vol. 90, no. 3, pp. 727-752, 2010.

[2] C. A. Sari and E. H. Rachmawanto, "Gabungan Algoritma *Vernam cipher* dan End of File Untuk Keamanan Data," *Jurnal Techno.Com,* vol. 13, no. 3, pp. 150-157, 2014.

[3] E. H. Rachmawanto and C. A. Sari, "Keamanan File Menggunakan Teknik Kriptografi Shift *Cipher*," *Jurnal Techno. Com,* vol. 14, no. 4, pp. 329-335, 2015.

[4] C. A. Sari, E. H. Rachmawanto, Y. P. Astuti and L. Umaroh, "Optimasi Penyandian File Kriptografi Shift *Cipher*," in *Prosiding Sendi_U 2016*, Semarang, 2016.

[5] Y. P. Astuti, E. H. Rachamwanto and C. A. Sari, "Optimasi Enkripsi Password Menggunakan Algoritma Blowfish," *Jurnal Techno.Com,* vol. 15, no. 1, pp. 15-21, 2016.

[6] Y. P. Astuti, C. A. Sari and E. H. Rachmawanto, "Optimasi Metode Blowfish Untuk Mengamankan Password Pada Kriptografi," in *Seminar Nasional Multi Disiplin Ilmu dan Call Papers Unisbank ke 2 Tahun 2016*, Semarang, 2016.

[7] R. Sadikin, Kriptografi Untuk Keamanan Jaringan, Yogyakarta: Andi, 2012.

[8] D. Ariyus, Pengantar Ilmu Kriptografi: Teori, Analisi dan Implementasi, Yogyakarta: Andi, 2008.

[9] E. H. Rachmawanto, C. A. Sari, Y. P. Astuti and L. Umaroh, "Kriptografi Dengan Algoritma *Vernam cipher* Untuk Keamanan Data," in *Prosiding Sendi_U ke 2 Tahun 2016*, Semarang, 2016.

[10] S. Kromodimoeljo, Teori dan Aplikasi Kriptografi, Jakarta: SPK IT Consulting, 2009.