**AnoaTIK: Jurnal Teknologi Informasi dan Komputer** e-ISSN: **2987-7652**
Vol. 3, No. 1, Juni 2025 Hal. 42-49

# IMPLEMENTASI ALGORITMA KRIPTOGRAFI *BLOWFISH* UNTUK PENGAMANAN *FILE* BERBASIS DESKTOP

**Winda[*1], La Surimi[2], Ilham Julian Efendi[3]**

[1,2,3] Program Studi Ilmu Komputer, Fakultas Matematika & Ilmu Pengetahuan Alam
Universitas Halu Oleo
**Email**: [1]windatxt@gmail.com, [2]lasurimi@uho.ac.id, [3]ilham.julian.efendi@uho.ac.id@uho.ac.id
***** *Penulis Korespondensi*

**Abstrak**

Penelitian ini bertujuan untuk mengimplementasikan algoritma kriptografi *Blowfish* dalam aplikasi pengamanan *file* berbasis desktop, guna meningkatkan kerahasiaan dan keamanan data pengguna. *Blowfish*, sebagai cipher blok simetris, dipilih karena efektivitasnya dalam mengenkripsi data 64-bit melalui 16 putaran fungsi Feistel. Proses enkripsi melibatkan penggunaan tabel *P-array* dan *S-box*, serta operasi XOR dan penambahan modulo $2^{32}$. Aplikasi yang dikembangkan mendukung berbagai format *file*, termasuk PDF, TXT, JPG, PNG, MP3, dan MP4, dengan batasan ukuran maksimal 50 MB. Metodologi penelitian menggunakan *Rapid Application Development* (RAD) untuk mempercepat siklus pengembangan, dengan tahapan meliputi perencanaan kebutuhan pengguna, desain, pengembangan iteratif, dan pengujian. *White box testing* diterapkan untuk memverifikasi implementasi algoritma *Blowfish* dan fungsionalitas aplikasi. Analisis *avalanche effect* dilakukan untuk mengevaluasi sensitivitas algoritma terhadap perubahan kecil pada input, memastikan keamanan data yang kuat. Hasil pengujian menunjukkan bahwa algoritma *Blowfish* berhasil diimplementasikan, dengan setiap fungsi aplikasi berjalan sesuai harapan. *Avalanche effect* membuktikan bahwa perubahan kecil pada input menghasilkan perubahan signifikan pada output enkripsi, menandakan tingkat keamanan yang tinggi. Aplikasi ini dirancang untuk beroperasi secara *offline* dan memiliki antarmuka yang mudah digunakan, sehingga dapat diakses oleh pengguna dengan berbagai tingkat keahlian teknis.

**Kata kunci**: *Blowfish*, kriptografi, enkripsi, dekripsi, *avalanche effect*, *white box testing*

***Abstract***

*This research aims to implement the Blowfish cryptographic algorithm in a desktop-based file security application, to enhance user data confidentiality and security. Blowfish, as a symmetric block cipher, was chosen for its effectiveness in encrypting 64-bit data through 16 rounds of the Feistel function. The encryption process involves the use of P-array and S-box tables, as well as XOR and modulo 2^32 addition operations. The developed application supports various file formats, including PDF, TXT, JPG, PNG, MP3, and MP4, with a maximum size limit of 50 MB. The research methodology uses Rapid Application Development (RAD) to accelerate the development cycle, with stages including user requirements planning, design, iterative development, and testing. White box testing is applied to verify the implementation of the Blowfish algorithm and application functionality. Avalanche effect analysis is performed to evaluate the algorithm's sensitivity to small changes in input, ensuring robust data security. The test results show that the Blowfish algorithm was successfully implemented, with each application function running as expected. The avalanche effect proves that small changes in input produce significant changes in the encryption output, indicating a high level of security. This application is designed to operate offline and has a user-friendly interface, making it accessible to users with various levels of technical expertise.*

**Keywords**: *Blowfish, cryptography, encryption, decryption, avalanche effect, white box testing*

## 1. PENDAHULUAN

Perkembangan teknologi informasi yang pesat mendorong peningkatan kebutuhan akan keamanan data. Dalam era digital, banyak data sensitif yang diproses dan disimpan dalam perangkat komputer, baik di lingkungan personal maupun bisnis. Ketika data tidak diamankan dengan baik, risiko penyalahgunaan, pencurian, atau kebocoran informasi menjadi semakin tinggi. Oleh karena itu, dibutuhkan solusi keamanan yang efektif, salah satunya dengan menerapkan kriptografi [1].

Kriptografi adalah ilmu yang berkaitan dengan teknik untuk menjaga kerahasiaan data melalui proses enkripsi dan dekripsi. Enkripsi mengubah data asli

menjadi bentuk yang tidak dapat dibaca tanpa kunci tertentu, sedangkan dekripsi mengembalikan data ke bentuk aslinya. Salah satu algoritma yang digunakan untuk proses ini adalah Blowfish. Blowfish merupakan algoritma simetris berbasis block cipher yang dikembangkan oleh Bruce Schneier pada tahun 1993 sebagai alternatif untuk Data Encryption Standard (DES). Blowfish memiliki ukuran kunci yang bervariasi antara 32 hingga 448 bit dan dikenal dengan kecepatannya yang tinggi serta efisiensi dalam penggunaan memori [2]. Penelitian ini berfokus pada implementasi algoritma Blowfish dalam aplikasi desktop untuk pengamanan *file.*

## 2. TINJAUAN PUSTAKA

### 2.1 Penelitian Terdahulu

Penelitian pertama [3] dengan judul Pengamanan *File* Audio menggunakan Algoritma Kriptogafi *Blowfish* dan Pengujian UAT menunjukkan bahwa implementasi algoritma dalam aplikasi desktop berhasil meningkatkan keamanan data, kualitas suara setelah proses enkripsi dan dekripsi tetap terjaga, serta ukuran *file* suara tidak mengalami perubahan yang signifikan. *Delay* yang dihasilkan selama proses enkripsi dan dekripsi tergantung pada ukuran *file* yang diproses, namun tidak memakan waktu yang lama saat pengiriman. Penelitian yang dilakukan oleh [4] dengan judul Implemntasi Algoritma *Blowfish* untuk Pengamanan *File* PDF, menunjukkan bahwa aplikasi yang dikembangkan mampu melakukan enkripsi dan dekripsi *file* PDF dengan efektif menggunakan algoritma. *File* asli yang berisi informasi confidential berhasil dienkripsi menjadi *file cipher*, sehingga hanya dapat diakses oleh pihak yang memiliki *password* dan kunci dekripsi yang tepat.

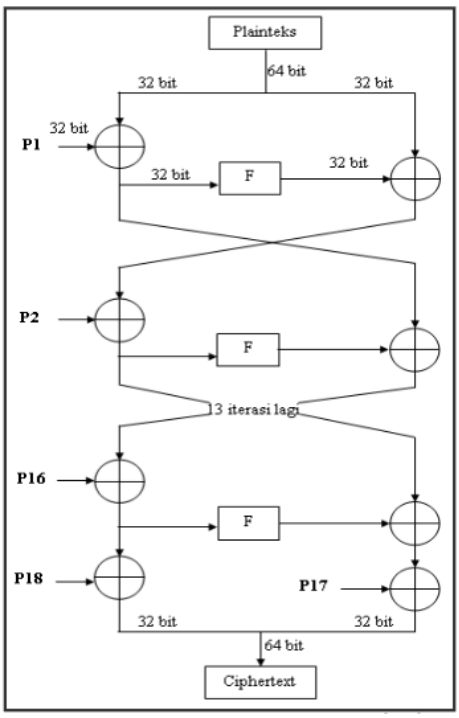
### 2.2 Kriptografi

Kriptografi menurut etimologinya berasal dari bahasa Yunani yaitu *crypto* dan *graphein*. *Crypto* berarti tersembunyi dan *graphein* berarti menulis. Kriptografi adalah ilmu yang mempelajarinya metode untuk mengirim pesan secara rahasia atau disamarkan sehingga hanya penerima yang dituju yang dapat menghapus penyamaran dan membaca pesan yang dikirim. Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika untuk menjaga keamanan informasi [5].

### 2.3 *Blowfish*

*Blowfish* adalah algoritma yang dikembangkan oleh seorang ahli kriptografi bernama Bruce Schneier, yang juga menjabat sebagai Presiden di *Counterpane Internet Security*, Inc. Algoritma ini diperkenalkan pada tahun 1994 dan dirancang khusus untuk digunakan pada komputer dengan mikroprosesor berukuran besar (32-bit ke atas) yang memiliki *cache* data yang besar *Blowfish* merupakan algoritma cipher blok dengan ukuran blok 64-bit dan menggunakan kunci yang panjangnya dapat disesuaikan [6]. Enkripsi data terdiri dari iterasi fungsi sederhana sebanyak 16 kali putaran, masukannya adalah 64bit elemen data X. Setiap putaran terdiri dari permutasi kunci-dependent dan substitusi kunci dan data dependent. Semua operasi adalah penambahan dan XOR pada variabel 32-bit. Operasi tambahan lainnya hanyalah empat penelusuran tabel array berindeks untuk setiap putaranLangkahnya adalah seperti berikut [7].

1. *Plaintext* yang akan dienkripsi diasumsikan sebagai masukan, *plaintext* tersebut diambil sebanyak 64-bit, dan apabila kurang dari 64-bit maka kita tambahkan bitnya, supaya dalam operasi nanti sesuai dengan datanya.
2. Hasilnya dibagi menjadi dua bagian, 32-bit pertama disebut XL, dan 32-bit yang kedua disebut XR..
3. Selanjutnya lakukan operasi :
   XL = XL xor Pi dan
   XR = F(XL) xor XR
4. Hasil dari operrasi diatas ditukar XL menjadi XR dan XR menjadi XL.
5. Lakukan perulangan sebanyak 16 kali, perulangan yang ke-16 lakukan lagi proses penukaran XL dan XR.
6. Proses ke-17 lakukan operasi untuk :
   XR = XR xor P17 dan XL = XL xor P18.
7. Terakhir satukan kembali XL dan XR sehingga menjadi 64-bit (kembali)




Gambar 2. 1 Pola Jaringan Feistel *Blowfish [8]*

Dekripsi *Blowfish* sama persis dengan enkripsi *Blowfish*, kecuali bahwa P1, P2,…, P18 digunakan pada urutan yang berbalik *(reverse).*

### 2.4 *Avalanche Effect*

*Avalanche effect* merupakan salah satu metode pengujian dalam kriptografi yang menentukan seberapa baik suatu algoritma dengan mencari besar persentase pengubahan pesan pada saat proses enkripsi dilakukan [9]. Menghitung *avalanche effect:*

$$Avalanche\ Effect\ (AE)\ =\ \text{X}\,100\%$$

### 2.5 *Whitebox Testing*

*White box testing* adalah metode yang digunakan untuk menguji struktur internal, desain, dan kode program dari suatu perangkat lunak. Metode ini dapat mengidentifikasi kesalahan yang terjadi dalam implementasi perangkat lunak [10].

## 3. METODE PENELITIAN

### 3.1 Prosedur Penelitian

Alur dari prosedur penelitian dilakukan dalam beberapa tahapan tahapan yang meliputi:

1. Studi literature
2. Analisis kebutuhan
3. Perancangan system
4. Implementasi
5. Pengujian

### 3.2 Alat atau Instrumen Penelitian

Alat atau instrument penelitian yang digunakan dapat dilihat pada Tabel 3.1.

Tabel 3.1 Alat atau Instrumen Penelitian

| No | Alat/Instrumen | Spesifikasi/Keterangan |
|---|---|---|
| 1 | Laptop | Intel(R) Celeron(R), RAM 4 GB, sistem operasi *Windows* 10. |
| 2 | Visual Studio 2022 | IDE yang mendukung bahasa pemrograman C#. |
| 3 | Bahasa Pemrograman C# | Bahasa pemrograman berbasis .NET *Framework* |
| 4 | Algoritma *Blowfish* | Implementasi algoritma enkripsi dan dekripsi *Blowfish.* |
| 5 | *File* Uji | *file* dokumen (PDF dan TXT), *file* gambar (JPEG/JPG dan PNG), *file* audio (MP3), dan *file* video (MP4). |

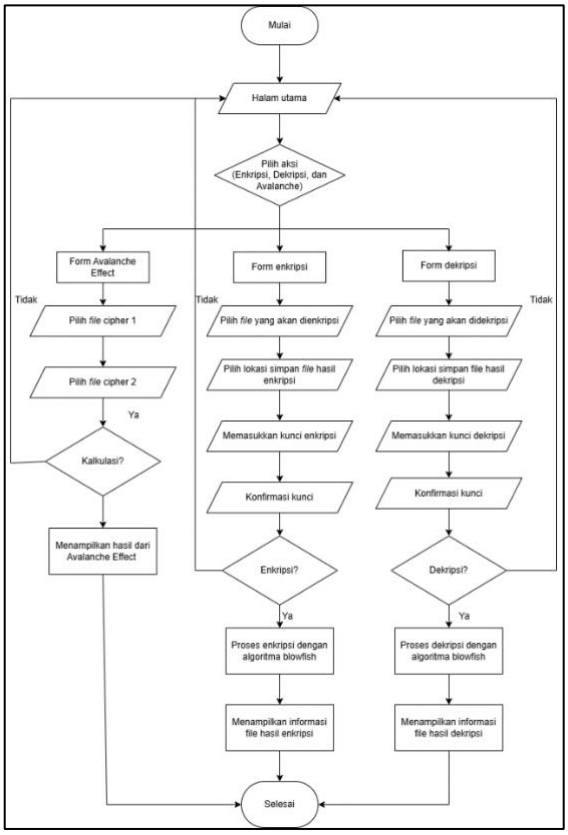
### 3.3 Metode Pengembangan Sistem

Dalam pengembangan sistem ini, digunakan metode *Rapid Application Development (RAD)*. Adapaun tahapan dari *Rapid Application Development (RAD).*

### 3.4 Perancangan Sistem

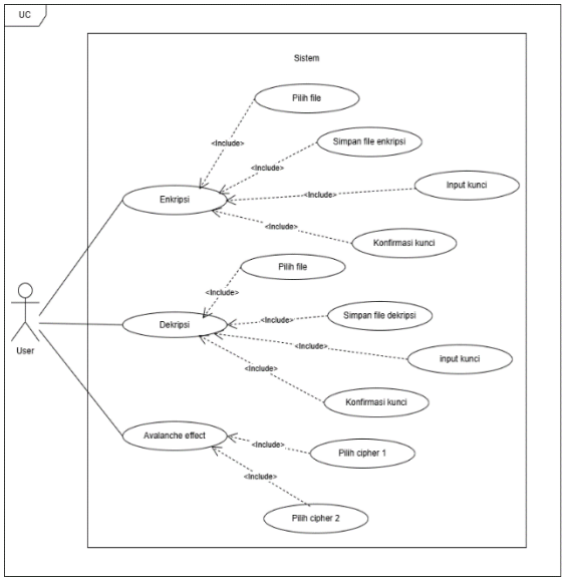
Dalam proses perancangan sistem ini, tahap pertama yang dilakukan adalah merancang struktur dan fungsionalitas dari sistem yang akan dibangun. Sebagai bagian dari perancangan ini, penulis memanfaatkan *Unified Modelling Language (UML)***,** sebuah standar yang umum digunakan dalam perancangan perangkat lunak untuk menggambarkan berbagai aspek dari sistem, seperti *use case*, *activity diagram,* dan *sequence diagram*.

*Flowchart* sistem membantu memvisualisasikan alur kerja aplikasi yang dikembangkan.




Gambar 3.1 *Flowchart* Alur Sistem

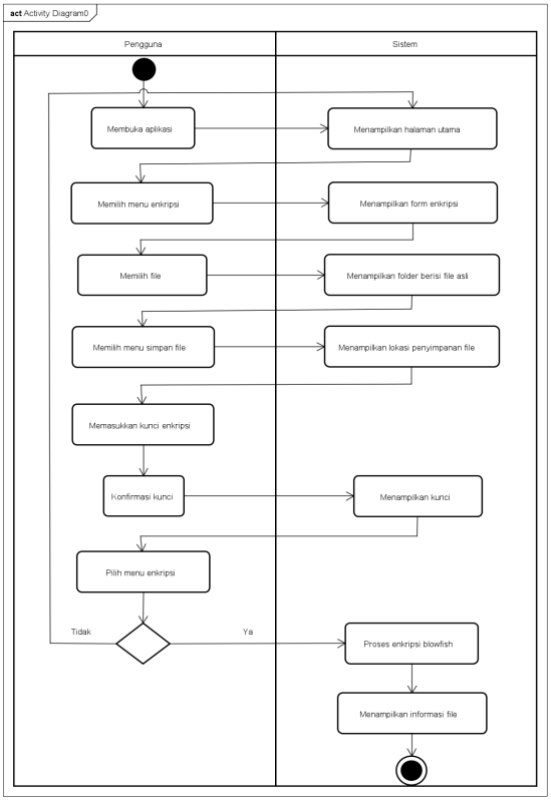
*Use Case Diagram* merupakan pemodelan dari fungsionalitas sistem. Pemodelan ini digunakan selama analisa kebutuhan untuk menentukan kebutuhan dan menentukan apa yang dapat dilakukan sistem, dapat dilihat pada Gambar 3.2.




Gambar 3.2 *Use Case Diagram*

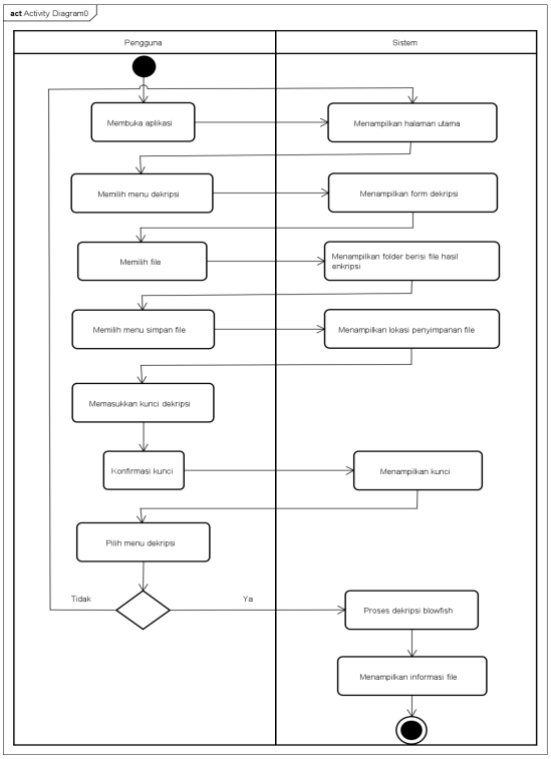
### 3.4.1 *Activity Diagram*

*Activity diagram* Gambar 3.3 menggambarkan proses enkripsi *file* menggunakan algoritma *Blowfish* hingga selesai.



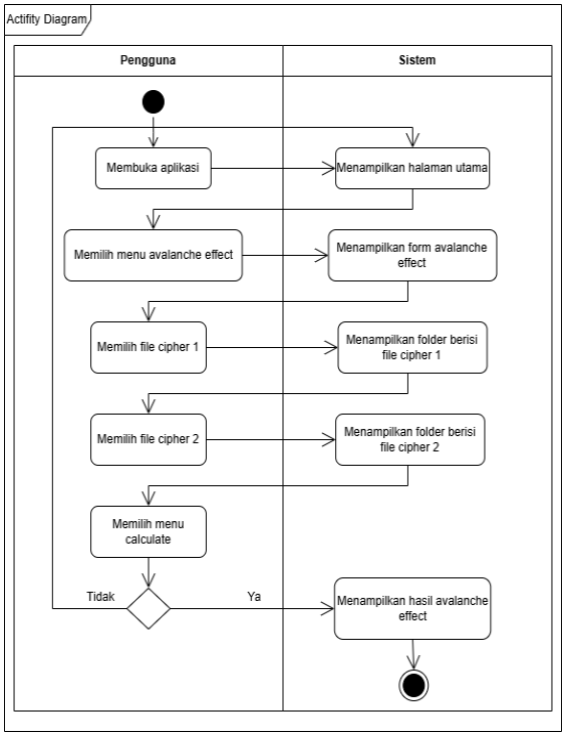

Gambar 3.3 *Activity Diagram* Enkripsi

*Activity diagram* Gambar 3.4 menggambarkan proses dekripsi *file* menggunakan algoritma *Blowfish* hingga selesai.




Gambar 3.4 *Activity Diagram* Dekripsi

Gambar 3.5 menggambarkan proses dekripsi *avalanche effect* hingga selesai.

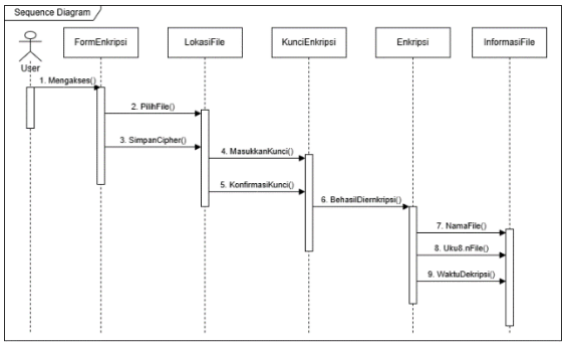



Gambar 3.5 *Activity Diagram Avalanche Effect*

### 3.4.2 *Sequence Diagram*

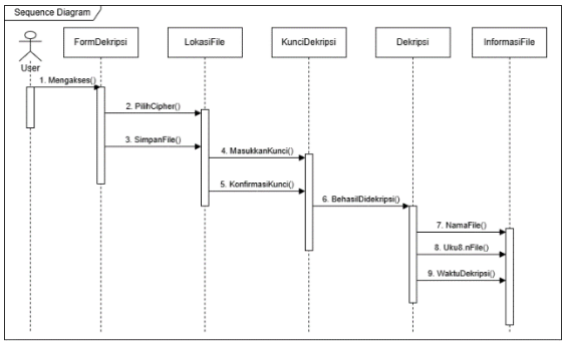
*Sequence diagram* adalah diagram yang menggambarkan kelakuan objek pada *use case* yang disusun dalam urutan kronologis.

. *Sequence diagram* enkripsi dapat dilihat pada Gambar 3.6.



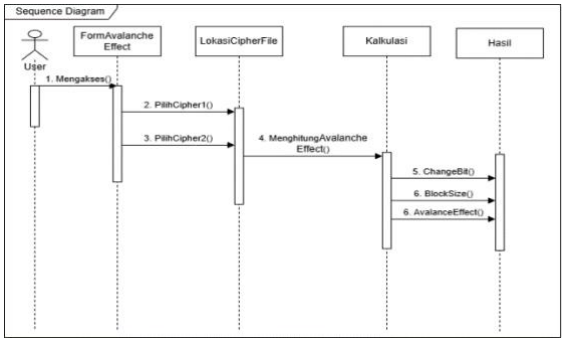

Gambar 3.6 *Sequence Diagram* Enkripsi

*Sequence diagram* dekripsi dapat dilihat pada Gambar 3.7.




Gambar 3.7 *Sequence Diagram* Dekripsi

*Sequence diagram avalanche effect* dapat dilihat pada Gambar 3.8.

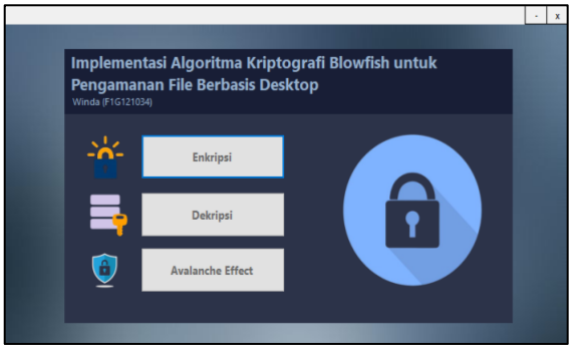



Gambar 3.8 *Sequence Diagram Avalanche Effect*

## 4. HASIL DAN PEMBAHASAN

### 4.1 Implementasi Form Utama

Tampilan form utama merupakan tampilan awal dari aplikasi pengamanan *file* yang dibuat. Pada form ini terdapat fitur yang memungkinkan pengguna memilih fitur yang diinginkan, seperti enkripsi *file,* dekripsi *file,* dan pengujian *avalanche effect*. Adapun tampilan fom utama dapat dapat dilihat pada Gambar 4.1.




Gambar 4.1 Form Utama

### 4.2 Implementasi Form Enkripsi

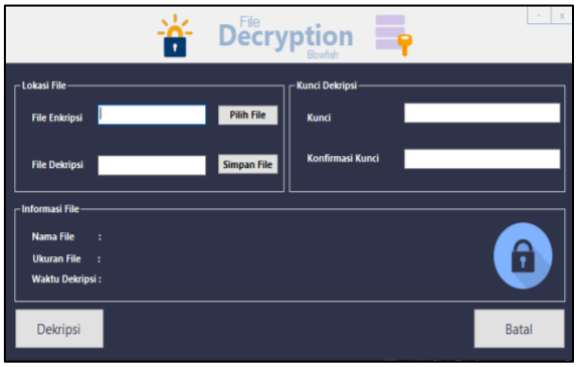
Implementasi form enkripsi merupakan form yang berfungsi untuk melakukan ekripsi *file.* Pada form ini pengguna dapat memilih *file* yang ingin dienkripsi, memilih lokasi untuk penyimpanan *file* hasil enkripsi dan memasukkan kunci enkripsi yang diperlukan. Pengguna dapat melakukan enkripsi dengan menekan tombol enkripsi. Form ini juga menampilkan informasi mengenai *file* yang telah dipilih, serta status atau hasil dari proses enkripsi yang telah dilakukan. Tampilan form enkripsi dapat dilihat pada Gambar 4.2.


Gambar 4.2 Form Enkripsi

### 4.3 Implementasi Form Dekripsi

Implementasi form dekripsi berfungsi untuk melakukan dekripsi *file.* Pada form ini pengguna dapat memilih *cipherfile*, memilih lokasi untuk penyimpanan *file* hasil dekripsi dan memasukkan kunci enkripsi yang diperlukan. Dekripsi dapat dilakukan dengan menekan tombol dekripsi. Form ini juga menampilkan informasi mengenai *file* yang telah

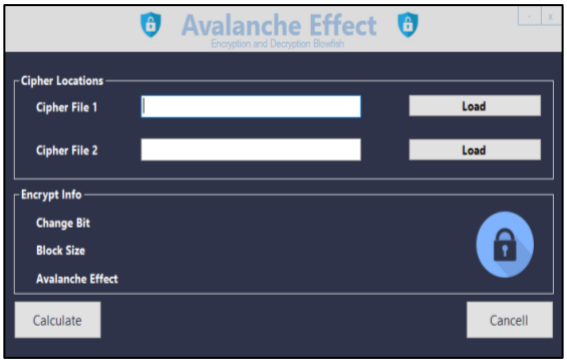
didekripsi, seperti nama *file*, ukuran *file,* dan waktu dekripsi. Adapun tampilan fom dekripsi dapat dapat dilihat pada Gambar 4.3.




Gambar 4.3 Form Dekripsi

### 4.4 Implementasi Form *Avalanche Effect*

Tampilan form *Avalanche Effect* merupakan form yang berfungsi untuk melakukan pengujian *avalanche effect* pada algoritma *Blowfish*. Form ini juga menampilkan informasi mengenai jumlah bit yang berubah *(change bit*), persentase *Avalanche Effect* yang dihasilkan, serta ukuran blok *(block size)* yang digunakan dalam pengujian. Hasil pengujian ini memberikan gambaran mengenai sensitivitas algoritma *Blowfish* terhadap perubahan kecil pada input. Tampilan form *avalanche effect* dapat dilihat pada Gambar 4.4.




Gambar 4.4 Form *Avalanche Effect*

### 4.5 Hasil Pengujian Data

Setelah dilakukan proses enkripsi dan dekripsi terhadap *file* uji, didapatkan waktu proses enkripsi dan dekripsi yang ditunjukkan pada Tabel 4.1.

Tabel 4.1 Waktu Enkripsi dan Dekripsi Berdasarkan Jenis *File*

| Jenis *File* | Format *File* | Ukuran *File* | Waktu Enkripsi | Waktu Dekripsi |
|---|---|---|---|---|
| *File* dokumen | PDF | 626 KB | 771.4201 Ms | 343.8371 Ms |
| | TXT | 22 KB | 21.4619 Ms | 42.4655 Ms |
| *File* gambar | JPG | 79.5 KB | 59.4297 Ms | 46.9159 Ms |
| | PNG | 4.68 MB | 2512.1052 Ms | 2545.3148 Ms |
| *File* audio | MP3 | 5.68 MB | 3064.5176 Ms | 3226.8546 Ms |
| *File* video | MP4 | 5.48 MB | 2957.9584 Ms | 2983.3066 Ms |

### 4.5 Pengujian Kualitas *Blowfish* Berdasarkan *Avalanche Effect*

Pengujian kualitas algoritma *Blowfish* berdasarkan *avalanche effect*, bertujuan untuk mengukur sejauh mana perubahan kecil pada input (seperti perubahan satu bit pada *plaintext* atau kunci) dapat mempengaruhi output *(ciphertext).* Pengujian dengan *avalanche effect* dilakukan dengan perubahan bit kunci, perubahan bit *plaintext,* dan pengujian kunci lagi. Pengujian *avalanche effect* dengan mengubah bit *plaintext.*

Diketahui:

1. *Plaintext* 1: SECURITY
2. *Plaintext* 2: SFCURITY (Perubahan 1 karakter: "E" menjadi "F")
3. Kunci 1: MYPASSWORD1234567
4. Kunci 2: MYPASSWORD1234567 (Tidak berubah)
5. Ciphertext 1 (Heksadesimal): D755AC55A208F024
6. Ciphertext 2 (Heksadesimal): 807068C65BB3EDBD
7. Biner 1: 0101011010001000001000111100000010010 0
8. Biner 2: 0001100101101110110011111010110111110 1

Perhitungan:

1. Memabandingkan setiap bit dari Biner 1 dan Biner 2 menggunakan operasi XOR.
2. Setelah dibandingkan, jumlah bit yang berbeda adalah 35.
3. Menghitung *avalanche effect:*
   $Avalanche\ Effect\ (AE) = \frac{\text{Jumlah yang berbeda}}{\text{Total bit}}$ X 100%

Jumlah bit yang berbeda = 35
Total Bit = 64 bit (8 byte * 8 bit/byte)
*Avalanche effect*: $\frac{35}{64}$ X 100% = 54.6875%

Hasil *avalanche effect* = 54.69% (Setelah pembulatan) Rata-rata *avalanche effect* yang dihasilkan pada perubahan bit kunci adalah 51,876%.

*Pengujian* avalanche effect *dengan mengubah bit* plaintext.

Diketahui:

1. Plaintext 1: SECURITY
2. Plaintext 2: SECURITY (tidak berubah)
3. Kunci 1: MYPASSWORD1234567
4. Kunci 2: NYPASSWORD1234567 (Perubahan 1 karakter: "M" menjadi "N")
5. Ciphertext 1 (Heksadesimal): D755AC55A208F024
6. Ciphertext 2 (Heksadesimal): DBF31692F5CF3ECB
7. Biner 1: 01010110100010000010001111000000100100
8. Biner 2: 01001011110101110011110011111011001011

Perhitungan:

1. Memabandingkan setiap bit dari Biner 1 dan Biner 2 menggunakan operasi XOR.
2. Setelah dibandingkan, jumlah bit yang berbeda adalah 38.
3. Menghitung *avalanche effect:*
   $Avalanche\ Effect\ (AE) = \frac{\text{Jumlah yang berbeda}}{\text{Total bit}}\ X\ 100\%$
   Jumlah bit yang berbeda = 38
   Total Bit: 64 bit (8 byte * 8 bit/byte)
   *Avalanche effect*: $\frac{38}{64}$ X 100% = 59.375%

Hasil a*valanche effect* = 59,38% (Setelah pembulatan). Rata-rata *avalanche effect* yang dihasilkan pada perubahan perubahan bit *plaintext* tercatat 51,25%, dan 51,564% pada pengujian kunci lagi. Hasil ini menunjukkan bahwa *Blowfish* mampu menghasilkan *ciphertext* yang sangat berbeda meskipun hanya ada perubahan kecil pada kunci atau *plaintext*.

### 4.6 Pengujian *White Box Testing*

*Cyclomatic Complexity* untuk *SetupKey():*

Jumlah Edge (E) = 45

Jumlah Node (N) = 38

V(G) = 45 - 38 + 2 = 9

Kompleksitas sedang (9). Ini menunjukkan bahwa fungsi ini memiliki beberapa jalur eksekusi yang berbeda.

*Cyclomatic Complexity* untuk *Cryp*t_ECB():

Jumlah Edge (E) = 13

Jumlah Node (N) = 13

V(G) = 13 - 13 + 2 = 2

Kompleksitas rendah (2). Fungsi ini relatif sederhana dengan satu percabangan.

*Cyclomatic Complexity* untuk *BlockEncrypt() dan BlockDecrypt():*

Jumlah Edge (E) = 4

Jumlah Node (N) = 4

V(G) = 4 - 4 + 2 = 2

Kompleksitas rendah (2). Fungsi-fungsi ini sangat sederhana.

*Cyclomatic Complexity* untuk *Encipher() dan Decipher():*

Jumlah Edge (E) = 20

Jumlah Node (N) = 18

V(G) = 20 - 18 + 2 = 4

Kompleksitas rendah (4). Fungsi-fungsi ini memiliki satu loop.

*Cyclomatic Complexity* untuk *FFunction():*

Jumlah Egde (E) = 3

Jumlah Node (N) = 4

V(G) = 3 - 4 + 2 = 1

Kompleksitas sangat rendah (1). Fungsi ini sangat sederhana tanpa percabangan atau loop.

## 5. KESIMPULAN

Berdasarkan hasil penelitian implementasi algoritma kriptografi *Blowfish* untuk pengamanan *file* berbasis desktop, dapat disimpulkan bahwa sistem berhasil mengenkripsi dan mendekripsi *file* dengan format (PDF, TXT, JPG, PNG, MP3, MP4) dengan batas ukuran *file* maksimum 50 MB. Sistem ini juga mendukung pengujian *avalanche effect* untuk mengukur sensitivitas perubahan kunci terhadap hasil enkripsi, yang menunjukkan bahwa tujuan aplikasi tercapai dengan baik. Hasil pengujian avalanche effect dengan rata-rata persentase 51,876%, 51,25%, dan 51,564% menunjukkan efektivitas algoritma *Blowfish* dalam menghasilkan perubahan signifikan pada *ciphertext* dan *cipherfile*, yang mengindikasikan bahwa algoritma ini memiliki tingkat keamanan yang baik dalam mengamankan *file*. Pengujian *white box testing* menunjukkan bahwa implementasi algoritma *Blowfish,* khususnya fungsi *SetupKey(), Crypt_ECB(), BlockEncrypt(), BlockDecrypt(), encipher(), decipher(), dan FFunction(),* berfungsi sesuai dengan spesifikasi yang diharapkan.

## 6. DAFTAR PUSTAKA

[1] Olivia, B., Irine, P., Tahir, M., Ayu, N., Cholili, D. Y., Mulaikah, D., Batsul, A., & Septian, M. (2023). Implementasi Kriptografi Pada Keamanan Data Menggunakan Algoritma Advance Encryption Standard ( Aes ) Cryptographic Implementation in Data Security Using Advanced Encryption Standard ( Aes ) Algorithm. *Jurnal Simantec*, *11*(2), 167–174.

[2] Oktaviani, S., Rizky, F., & Gunawan, I. (2023). Analisis Keamanan Data Dengan Menggunakan Kriptografi Modern Algoritma Advance Encryption Standar (AES). *Jurnal Media Informatika*, *4*(2), 97–101.

[3] Siswanto, Amsari, F. D., Prasetyo, B. H., Pramusinto, W., Utama, G. P., & Anif, M. (2021). Pengamanan *File* Audio Menggunakan Algoritma Kriptografi Blowfish. *SISFOTEK*, *5*, 262–269.

[4] Gamaliel, F., & Arliyanto, P. Y. D. (2024). Implementasi Algoritma Blowfish untuk Pengaman *File* PDF. *Jurnal Informatika & Rekayasa Elektronika*, *7*(1), 60–67.

[5] Simarmata, J., Sriadhi, & Rohim, R. (2022). *KRIPTOGRAFI Teknik Keamanan Data & Informasi* (Issue April 2022).

[6] Zuli, F., & Irawan, A. (2017). Implementasi Kriptografi Dengan Algoritma Blowfish Dan Riverst Shamir Adleman (Rsa) Untuk Proteksi *File*. *Jurnal Ilmiah FIFO*, *9*(1), 5–13.

[7] Fahriani, N., & Rosyid, H. (2019). Implementasi Teknik Enkripsi dan Dekripsi di File Video menggunakan Analogi Blowfish. 6, 697–702.

[8] Gamaliel, F., & Arliyanto, P. Y. D. (2024). Implementasi Algoritma Blowfish untuk Pengaman File PDF. *Jurnal Informatika & Rekayasa Elektronika*, *7*(1), 60–67.

[9] Karima, N. A., Aisyah, A. N., Silla, H. V., Handoko, L. B., & Sani, R. R. (2024). Kriptografi Teks Berbasis Algoritma Substitusi Vigenere Cipher 8 Bit. *Jurnal Masyarakat Informatika*, *15*(1), 1–13.

[10] Sie, J. B. L., Izmy Alwiah Musdar, & Syamsul Bahri. (2022). Pengujian White Box Testing Terhadap Website Room Menggunakan Teknik Basis Path. *KHARISMA Tech*, *17*(2), 45–57.