Analisis Penggunaan Algoritma *Advanced Encryption Standard* (AES) untuk Meningkatkan Keamanan pada Proses Kompresi File Berbasis Kata Sandi

Leony Angela - 18219032
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
leony.angela88@gmail.com

***Abstract*— Kebutuhan untuk penyimpanan data semakin besar seiring waktu karena semakin bertambahnya data dan informasi yang perlu dikomunikasikan. Akibatnya, penggunaan ruang penyimpanan data dan informasi juga ikut serta meningkat. Salah satu cara untuk mengatasi keterbatasan ruang penyimpanan dan pengiriman adalah dengan mengompresi file menjadi file ZIP (pengarsipan file), selain ukuran file menjadi kecil, waktu pengiriman juga menjadi semakin cepat. Untuk meningkatkan keamanan pengiriman dan keamanan file ZIP, diperlukan suatu mekanisme untuk menambahkan elemen keamanan pada file. Pada makalah ini akan dibahas penggunaan algoritma *Advanced Encryption Standard* (AES) sebagai salah satu teknik kriptografi yang dapat digunakan untuk meningkatkan keamanan file. Selain itu, juga akan dianalisis penggunaan algoritma AES pada aplikasi (*software*) yang menyediakan layanan pengamanan pengarsipan file.**

Keywords*— Password-based compression, Advanced Encryption Standard, ZIP compression

I. Pendahuluan

A. Latar Belakang

Perkembangan teknologi dan informasi saat ini yang begitu pesat membantu manusia dalam aktivitas sehari-hari, dan telah menjadi penopang utama kegiatan manusia. Salah satu kebutuhan manusia dari sejak dahulu hingga sekarang adalah terkait penyimpanan data. Data merupakan aset penting yang bukan saja dibutuhkan oleh perusahaan untuk menunjang bisnisnya, tetapi data juga menjadi aset yang berharga bagi masing-masing individu. Seiring dengan perkembangan zaman dan teknologi, media penyimpanan data juga berevolusi dari masa ke masa, mulai dari berbentuk *punch card* hingga sekarang data dapat disimpan pada *cloud storage*. Teknologi yang semakin canggih ini mampu menjadi fasilitas untuk menyimpan data, yang dimana sekarang sudah dapat berbentuk data digital seperti video, gambar, atau audio hanya dapat satu tempat.

Seiring waktu, penyimpanan data digital meningkat pesat, ditambah hampir seluruh perusahaan, organisasi, ataupun individu sudah jarang memanfaatkan arsip fisik seperti kertas untuk menyimpan data atau informasi. Perkembangan teknologi *Cloud Computing* telah masuk dan mempengaruhi individu untuk melakukan penyimpanan data secara *cloud* karena menghemat *disk space*, tidak akan hilang, dan terjamin keamanannya. Untuk dapat mengirim atau menyimpan data yang ukurannya besar, manusia mulai menggunakan pengarsipan file, atau yang biasa kita sebut sebagai file ZIP, dimana kita menggabungkan beberapa file dalam satu atau beberapa berkas yang ukurannya lebih kecil (*compression*) daripada ketika kita mengirim file tersebut satu persatu. File terkompresi, atau file ZIP merupakan cara yang paling nyaman untuk mengirim data karena pengiriman file hanya dilakukan dalam satu kali pengiriman, yaitu dalam ZIP.

Tentu, keamanan menjadi aspek utama dalam setiap aktivitas pertukaran informasi. Karena aktivitas pengiriman informasi sering dilakukan, terlebih melalui internet atau email, tidak menutup kemungkinan ada pihak-pihak yang menyalahgunakan teknologi yang ada untuk menyadap atau mengintervensi pengiriman informasi sehingga diperlukan suatu metode untuk menjaga keamanan isi dari file tersebut. Ada banyak metode yang dapat digunakan untuk meningkatkan keamanan data, salah satunya adalah kriptografi. Pada penelitian ini, akan dilakukan analisis terhadap salah satu algoritma kriptografi, yaitu *Advanced Encryption Standard* (AES), dalam kegunaannya sebagai metode untuk mengamankan file ZIP dari pihak yang tidak terotorisasi.

B. Tujuan Penulisan Makalah

Tujuan penulisan makalah kriptografi ini adalah sebagai berikut.

1. Memahami proses enkripsi dan dekripsi menggunakan *Advanced Encryption Standard* (AES).
2. Mengetahui penerapan Algoritma kriptografi AES pada pengamanan kompresi pesan.

C. Metode Penulisan Makalah

Berikut merupakan metodologi yang digunakan dalam penulisan makalah kriptografi ini.

1. Studi Literatur

   Studi literatur dilakukan dengan mengumpulkan informasi dan data yang mendukung penulisan

makalah, melalui pencarian sumber-sumber yang relevan dari jurnal penelitian, media elektronik, dan tulisan lainnya yang berkontribusi dalam penulisan makalah ini. Selanjutnya, sumber informasi dan data akan diolah menggunakan metode analisis deskriptif.

2. Analisis dan Pembahasan

   Analisis dilakukan terhadap bagaimana algoritma AES memberikan sistem keamanan file ZIP menggunakan kata sandi (*password*) sebagai parameter keamanan dari data. Pembahasan dilakukan dengan menganalisis cara kerja algoritma sistem pengenkripsian pesan berbasis kata sandi (*password*).

3. Penulisan Makalah

   Penulisan makalah dilakukan untuk mendokumentasikan hasil yang diperoleh dari studi literatur dan eksperimen yang dilakukan.

II. Studi Terkait

Kriptografi berbasis kata sandi banyak digunakan secara luas dan terdapat banyak pendekatan yang dikembangkan.

Pendekatan kriptografi berbasis kata sandi yang dikembangkan oleh Morris dan Thompson, menekankan pada penggunaan *salt* dan iterasi untuk membentuk enkripsi berbasis kata sandi. *Salt* dalam konteks ini secara tradisional adalah untuk menghasilkan satu set kunci yang sesuai dengan kata sandi yang masukan, di antaranya dipilih secara acak menurut *salt* [5]. Pendekatan lain dengan teknik derivasi kunci termasuk menghitung pengulangan yang dilakukan untuk mengulang fungsi dasar untuk menghasilkan *witch key*.

Otentikasi pesan berbasis kata sandi adalah metode kriptografi yang juga menerapkan *message authentication code* (MAC) [12]. MAC dapat mengkonfirmasi bahwa pesan yang dikirim oleh pengirim tidak berubah selama pengiriman pesan. Metode ini menyediakan satu kunci untuk *server* dan satu untuk klien dan keduanya hanya diketahui oleh pihak yang berkomunikasi.

Hash-based message authentication code (HMAC) menggunakan kunci masing-masing satu untuk klien dan server untuk melakukan komunikasi. HMAC unik dibuat pada sisi klien, kemudian pengguna membuat permintaan ke *server* untuk melakukan hashing data dengan kunci *server* dan mengirim kembali ke klien sebagai bagian dari permintaan. HMAC lebih aman daripada *Message Authentication Code* (MAC) karena kunci dan pesan di-*hash* dalam langkah-langkah independen [7].

III. Dasar Teori

A. File ZIP

Kompresi file adalah sebuah teknik yang digunakan untuk menggabungkan, memadatkan beberapa data menjadi satu file sehingga ruang penyimpanan yang digunakan untuk data tersebut dapat menjadi lebih kecil. Penyimpanan data yang lebih kecil dapat membuat waktu pemrosesan atau pengiriman data menjadi lebih singkat dan efisien.

ZIP merupakan format kompresi yang paling terkenal. Banyak aplikasi yang memberikan layanan untuk menangani proses kompresi file, seperti WinZip, 7-Zip, WinRAR, dan lainnya. ZIP merupakan format file arsip yang mendukung kompresi data *lossless* (data asli direkonstruksi sempurna dari data terkompresi).

1. Format File ZIP

   Setiap file asli yang di arsip ke dalam file ZIP, memiliki *Local File Header* yang berisi *metadata* file tersebut. *Central File Header* yang berada di dalam *Central Directory* berfungsi sebagai indeks pengarsipan untuk setiap file yang ingin diarsipkan. Didalam *Central File Header* juga terdapat salinan metadata dari *Local File Header*. Pada bagian akhir struktur file, terdapat End of *Central Directory* yang berisi ukuran dan posisi dari *Central Directory*.

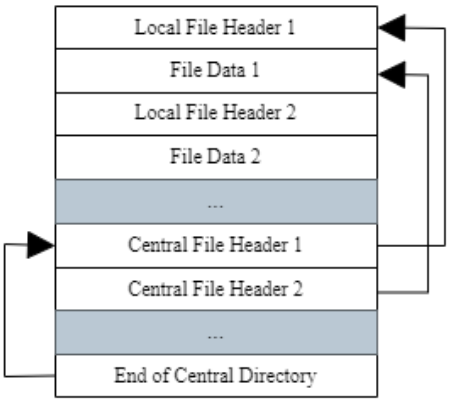

Figure 1. Format File ZIP (sumber: https://www.hanshq.net/zip.html)

   Setiap file didalam file arsip akan dikompresi dan disimpan sebagai satu kesatuan file, yang artinya jika terdapat kesamaan isi dari file-file tersebut, tidak akan dibuang untuk menghasilkan kompresi yang *lossless*.

2. Komponen Pembentukan File ZIP

   Kompresi Lempel–Ziv (LZW)

   Jacob Ziv dan Abraham Lempel, pencetus kompresi Lempel-Ziv, pada awalnya mengusulkan skema kompresi di mana data asli diuraikan menjadi urutan tiga tuple seperti

   (*pointer*, *length*, *next*)

   Algoritma LZW adalah algoritma berbasis *dictionary* yang terbentuk selama pembacaan *input* selama proses kompresi atau dekompresi. *Dictionary* dibentuk dengan tujuan untuk menyimpan karakter atau pola string tertentu [10]. Sebelum proses kompresi atau dekompresi dilakukan, *dictionary* akan diinisialisasi dengan simbol atau karakter, sehingga nilai *default* dari *dictionary* akan berisi 256 karakter

ASCII. Metode LZW menawarkan *lossless compression*.

Huffman Coding

Huffman Coding merupakan algoritma yang mengacu kepada frekuensi kemunculan sebuah simbol/karakter pada sebuah string sebagai sebuah *input*, dan menghasilkan *output* berupa kode *prefix*, yaitu pengkodean karakter/simbol menggunakan bit-bit yang ditentukan dari pembentukan *Huffman Tree*. *Huffman Coding* dicetuskan oleh David A. Huffman. *Huffman Coding* merupakan salah satu komponen algoritma dasar kompresi data, yaitu untuk dapat mengurangi jumlah bit yang digunakan untuk merepresentasikan pesan.

Huffman Coding menggunakan *Huffman Table* dan *Huffman Tree*. *Huffman Table* digunakan untuk menyimpan frekuensi kemunculan simbol dan *Huffman Tree* digunakan untuk menggambarkan sebuah rangkaian berdasarkan frekuensi pada *Huffman Table*. Hasil dari *Huffman Tree* digunakan untuk membuat *prefix code*.

B. Algoritma AES

Advanced Encryption Standard (AES) adalah algoritma penyandian berbasis *block cipher* yang dipublikasikan oleh NIST (National Institute of Standard and Technology) pada tahun 2001. AES bersifat *non-Feistel* karena ukuran blok yang dienkripsi selalu berukuran 128 bit. AES merupakan algoritma kriptografi lanjutan dari *Data Encryption Standard* (DES) karena telah ditemukan kunci yang digunakan dengan melakukan penyerangan secara *brute-force* [1]. Algoritma AES menggunakan 3 jenis panjang kunci kriptografi yang berbeda, yaitu 128, 192 dan 256 bit. Proses enkripsi menggunakan AES akan melewati putaran yang jumlahnya ditentukan berdasarkan panjang kunci yang digunakan.

AES memproses sebuah input berupa *state*. Sebuah blok file atau pesan (*plaintext*) dengan ukuran 128 bit harus diubah menjadi *state* terlebih dahulu. Pesan dipecah terlebih dahulu menjadi blok-blok dengan ukuran 128 bit (16 *byte*). Setiap 16 *byte* pesan dimasukan ke dalam matriks 4x4 *state* dalam notasi heksadesimal. Selanjutnya, proses enkripsi akan dilakukan terhadap *state* secara berulang dalam beberapa putaran. Sebelum masuk ke putaran, *state* di XOR dengan *round key*. Transformasi ini disebut *AddRoundKey*. Selanjutnya, untuk putaran ke-1 hingga n-1, transformasi yang dilakukan adalah sebagai berikut.

1. *SubBytes* melakukan substitusi *byte* pada matriks *state* dengan nilai yang ada pada S-box.
2. *ShiftRows* merupakan pergeseran baris *array state* secara *wrapping*, yaitu tidak ada *value* yang hilang, hanya saja posisinya yang bergeser.
3. *MixColumns* merupakan proses mengalikan matriks *state* dengan sebuah matriks untuk mengacak nilai matriks. Berikut adalah nilai matriks tersebut.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

Figure 2. Matriks Pengacak Nilai

4. *AddRoundKey* melakukan XOR antara *state* dengan *round key*.

Pada ronde terakhir, yaitu ronde ke-Nr dilakukan transformasi serupa dengan ronde lain namun tanpa transformasi *MixColumns*.

Untuk dekripsi AES, proses yang dilakukan merupakan kebalikan dari seluruh proses enkripsinya. Seluruh proses transformasi menggunakan proses transformasi *invers* dari transformasi dasar enkripsi AES [2].

1. *Invers AddRoundKey* melakukan operasi XOR antara matriks *state ciphertext* dengan *round key* yang. r*ound key* yang digunakan di setiap iterasinya berkebalikan dengan *round key* yang ada pada proses enkripsi.
2. *Invers SubBytes* dimana setiap nilai pada *state ciphertext* disubtitusi dengan nilai pada tabel *inverse* S-Box.
3. *Invers ShiftRow* yaitu melakukan transformasi pergeseran secara *wrapping* yang berlawanan arah dengan pada proses enkripsi (*wrapping* ke kanan) yang artinya bit bergeser ke arah kiri.
4. *Inverse MixColumns* dengan mengalikan setiap kolom hasil *Inverse AddRoundKey* dengan matriks berikut.

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix}$$

Figure 3. Matriks Pengacak Nilai

Setiap putaran dalam proses AES menggunakan kunci putaran (*round key*). Kunci ini biasa disebut *key schedule* yang digunakan untuk mendapatkan *round key* dari kunci masukan pengguna.

Pertama, kunci masukan pengguna diletakan pada sebuah array k[n] sepanjang empat elemen kunci. Selanjutnya, pembangkitan kunci putaran mengikuti operasi sebagai berikut.

1. RotWord merupakan operasi perputaran sebanyak satu *byte* ke arah kiri
2. SubBytes mensubstitusikan nilai hasil perputaran dengan tabel S-Box
3. Operasi Rcon melakukan operasi XOR antara hasil operasi SubBytes dengan *round constant* (Rcon)

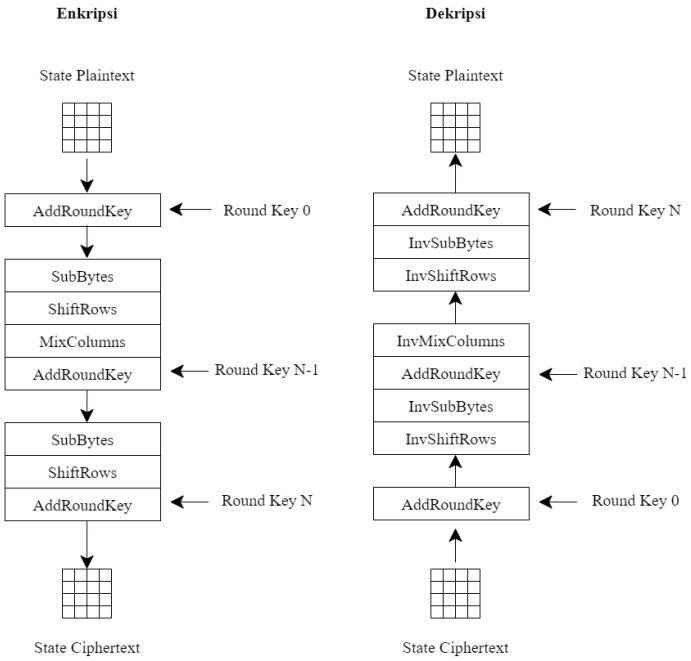

Figure 4. Enkripsi dan Dekripsi AES (sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/9%20-%20Review-beberapa-block-cipher-AES.pdf)

IV. Analisis dan Pembahasan

A. *Protokol Enkripsi Compressed File berbasis Kata Sandi*

Solusi paling umum untuk menyimpan data dalam bentuk terenkripsi sehingga tidak ada yang mengetahui sandi selain pemiliknya, adalah menggunakan enkripsi kunci simetris untuk mengenkripsi data. Protokol enkripsi berbasis kata sandi dirancang agar tetap dalam keadaan aman misalkan sebuah file terekspos kepihak yang tidak bertanggungjawab. Dalam pengiriman pesan melalui saluran komunikasi publik, seperti email, potensi terjadinya serangan seperti intervensi atau *passive eavesdropping*.

Dalam pembahasan ini, file yang diidentifikasi adalah file ZIP. File ZIP yang terkompresi dapat dilindungi menggunakan algoritma enkripsi dengan kata sandi sebagai kunci untuk algoritma tersebut. Saat ini, *cipher* yang sering digunakan untuk melakukan keamanan pada data adalah *Advanced Encryption Standard* (AES).

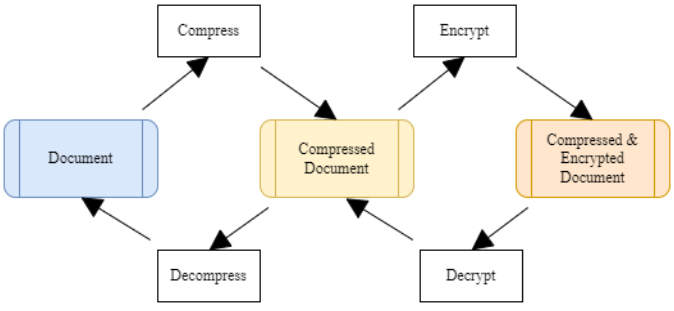

Figure 5. Metode Zipping dan Ekstraksi File berbasis Kata Sandi [3]

Sebuah fungsi bernama *Key Derivation Function* (KDF) digunakan untuk memproses kunci masukan pengguna yang akan digunakan dalam proses enkripsi dekripsi. *Password-based key-derivation functions* (PBKDF) mengambil kata sandi yang dimasukan pengguna dan memprosesnya untuk membangkitkan kunci untuk melakukan enkripsi dan dekripsi pesan. Fungsi ini menyediakan tingkat keamanan dua kali lipat karena kunci untuk proses enkripsi dan dekripsi bukanlah kunci masukan pengguna langsung, tetapi sebuah kunci baru yang dibangkitkan berdasarkan kunci masukan pengguna.

B. *Analisis Penggunaan AES dalam Kompresi File Berbasis Kata Sandi pada Aplikasi WinZip*

WinZip adalah layanan kompresi yang populer digunakan oleh pengguna komputer. Arsitektur kompresi WinZip mengikuti spesifikasi dari Info-ZIP, yaitu sebuah *open-source software* yang digunakan untuk menangani arsip ZIP.

Sebuah file ZIP dapat berisi banyak file lain didalamnya. Proses pengarsipan (*zipping*) file pada WinZip adalah dengan membuat dua buah record untuk masing-masing file, yaitu *main file record* dan *central directory record*. ZIP yang dihasilkan nantinya akan berisi seluruh *main file records* dari seluruh file asli yang digabungkan dengan seluruh *central directory record* dari seluruh file asli juga.

Proses enkripsi dilakukan dengan mengkompresi konten dari file *plaintext* dengan menggunakan metode kompresi yang digunakan pada WinZip. Kemudian, dari input kata sandi pengguna, akan dibangkitkan kunci AES, HMAC-SHA1, dan nilai verifikasi kata sandi menggunakan metode *password-based key-derivation functions* (PBKDF) yang telah dibahas pada bagian sebelumnya. Kunci AES yang dihasilkan akan digunakan untuk membangkitkan *round key* dan digunakan untuk mengenkripsi file yang telah terkompresi tadi menggunakan metode AES. Setelah file terkompresi telah dienkripsi, akan dihasilkan sebuah pesan MAC menggunakan algoritma HMAC-SHA1 dan hasilnya adalah 80 bit yang akan digunakan untuk metode autentikasi [13].

C. *Analisis Penggunaan AES dalam Kompresi File Berbasis Kata Sandi pada Aplikasi 7-ZIP*

7-ZIP merupakan salah satu pengarsip file yang juga banyak digunakan oleh orang lain karena sifatnya yang *multiplatform* dan *open-source* [6]. Aplikasi 7-ZIP juga memiliki layanan pengarsipan yang memanfaatkan kriptografi untuk membuat sebuah file ZIP yang terenkripsi. 7-ZIP menggunakan ZipCrypto dan AES-256 dengan mode CBC atau mode CTR untuk menghasilkan sebuah file ZIP terenkripsi [6]. Selain menghasilkan pengarsipan dalam format .zip, 7-ZIP juga dapat mengekstrak sebuah file .rar yang terenkripsi.

AES diimplementasikan dengan beberapa bentuk pada aplikasi 7-ZIP, yaitu implementasi murni menggunakan bahasa C dan implementasi menggunakan x86/x64 *assembler* dengan sebuah instruksi AES (AES-NI). AES-NI adalah sebuah set instruksi AES yang terdiri dari beberapa instruksi khusus untuk melakukan enkripsi/dekripsi [11].

D. *Analisis Algoritma AES dalam Melakukan Kompresi File berbasis Kata Sandi*

Sistem pengamanan file ZIP memanfaatkan konsep *Encrypt-then-Authenticate* dalam melakukan kompresi file menggunakan sebuah kata sandi untuk melakukan enkripsi dan dekripsi file tersebut.

Algoritma AES digunakan untuk pada tahap pengompresian folder/file menjadi file ZIP dan ekstrak file ZIP kembali menjadi folder file awal, dan kata sandi yang dimasukan akan menjadi parameter kunci keamanan file. File tidak dapat dibuka atau dibaca secara normal jika belum dilakukan dekripsi atau pembukaan kembali file ZIP dengan kata sandi.

Proses kompresi file berbasis kata sandi yang memanfaatkan algoritma AES digambarkan dengan diagram alur sebagai berikut.

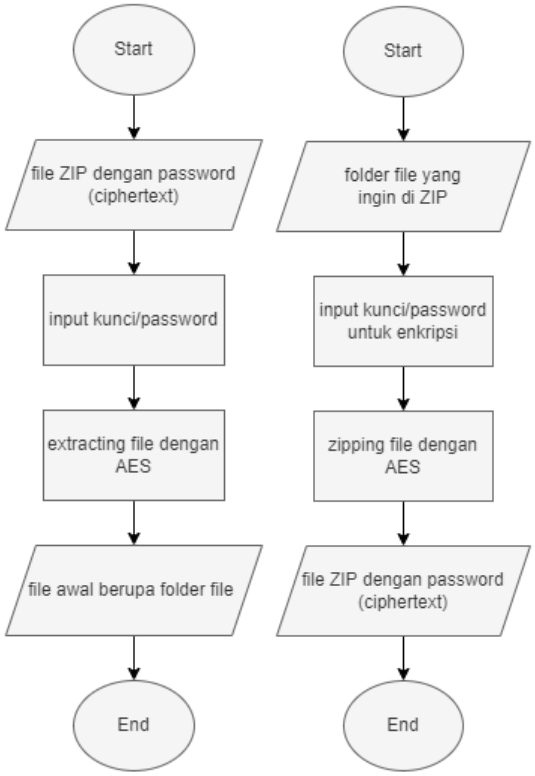

Figure 6. Diagram Alur Enkripsi File Terkompresi menjadi file ZIP berbasis Kata Sandi

Berikut ini merupakan *snippet code* yang dapat menggambarkan bagaimana proses pengarsipan file terkompresi yang menerapkan kata sandi dengan menggunakan algoritma AES.

```
import os
import pyzipper
from pyzipper import AESZipFile, ZIP_LZMA, WZ_AES
```

```
def archive (folder: str, filename: str, password: str) ->
None:
    if os.path.exists(filename):
        print('file already exists')
        return
    with AESZipFile(filename, 'w', compression =
ZIP_LZMA) as zip:
        zip.setpassword(bytes(password, 'utf-8'))
        zip.setencryption(WZ_AES)
        zip.writestr('password.txt', password)

        for root, dirs, files in os.walk(folder):
            for file in files:
                zip.write(os.path.join(root, file))

    with open(filename[:-4] + '.txt', 'w') as file:
        file.write(password)

def extract (filename: str, password: str) -> None:
    if not os.path.exists(filename):
        print('file does not exist')
        return

    with AESZipFile(filename, 'r') as zip:
        zip.setpassword(bytes(password, 'utf-8'))
        zip.setencryption(WZ_AES)
        zip.extractall()

def main() -> None:
    folder = input('input folder path: ')
    filename = input('input zip file name: ')
    password = input('input password: ')

    archive(folder, filename, password)
    extract(filename, password)

if __name__ == '__main__':
    main()
```

Fungsi ***archive*** akan menerima input berupa alamat direktori, nama file ZIP yang diinginkan, dan *password* yang digunakan untuk enkripsi file. Fungsi ini akan mengembalikan file zip terenkripsi dengan kata sandi.

Fungsi ***extract*** akan mengekstrak file ZIP dengan parameter *password* yang akan dimasukan oleh pengguna menjadi *folder* file seperti semula.

Main merupakan program utama yang dibuat untuk menerima masukan dari pengguna dan memanggil fungsi ***archive*** dan ***extract*** yang digunakan sebagai sistem pengamanan file.

Ketika dilakukan kompresi file menggunakan algoritma tersebut, didapatkan file berhasil dikompresi dengan menggunakan parameter kata sandi. Jika pengguna melakukan ekstraksi file secara manual, tanpa menggunakan algoritma ***extract***, maka pesan yang tersimpan didalam file ZIP tidak akan bisa terbaca. Berikut ini merupakan contoh file yang dibuka secara manual (diekstraksi langsung dari direktori lokal, tidak menggunakan algoritma ekstraksi file berbasis kata sandi).

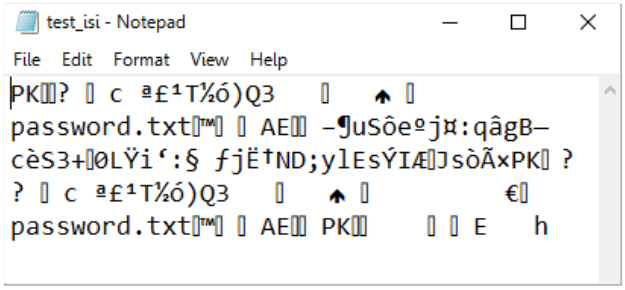

Figure 7. Hasil ekstraksi paksa tanpa melakukan dekripsi file terlebih dahulu

V. KESIMPULAN

Permasalahan keamanan tidak pernah habis untuk dibahas karena selalu bermunculan pihak-pihak yang tidak bertanggungjawab dan melakukan pencurian informasi. Salah satu hal yang terpengaruh adalah keamanan pada file, khususnya file ZIP. File ZIP banyak digunakan orang karena kegunaannya dalam mengkompresi file menjadi ukuran yang lebih kecil dan diarsipkan dalam satu file saja. Terlebih, file ZIP mendukung kompresi data yang *lossless*.

Pada makalah kriptografi ini, telah dianalisis bagaimana algoritma *Advanced Encryption Standard* (AES) digunakan untuk meningkatkan keamanan pada file ZIP, yaitu dengan melakukan proses enkripsi pada file yang telah terkompresi. Dari hasil analisis yang dilakukan, dapat disimpulkan bahwa penggunaan AES dapat mencegah serangan yang ingin mengakses dan membaca pesan yang disimpan atau sedang dikirim. Jika ada pihak yang membukanya secara paksa, maka isi file tidak akan bisa terbaca.

UCAPAN TERIMAKASIH

Dengan ini, penulis mengucapkan syukur dan terimakasih kepada Tuhan Yang Maha Esa atas kesempatan yang diberikan untuk boleh melaksanakan perkuliahan Kriptografi dan Koding dan menyelesaikan ini sehingga penulis mendapatkan pengalaman yang berharga selama pembelajaran selama 1 semester ini. Penulis juga ingin mengucapkan terimakasih kepada kedua orang tua yang turut membantu dan selalu mendukung penulis setiap harinya, kepada Pak Rinaldi Munir yang telah memberikan tidak hanya sekedar ilmu, tetapi juga pengalaman belajar dan cerita yang memotivasi penulis untuk selalu berusaha yang terbaik untuk menjalankan studi di ITB. Penulis juga ingin mengucapkan terimakasih kepada teman-teman kelas Kriptografi dan Koding yang suportif dan semangat dalam menjalani perkuliahan pada semester ini hingga akhir.

DAFTAR PUSTAKA

[1] Munir, Rinaldi. 2017. Slide Kuliah IF4020 Kriptografi: Advanced Encryption Standard (AES).

[2] Prameshwari, A., & Sastra, N. (2018). Implementasi Algoritma Advanced Encryption Standard (AES) 128 Untuk Enkripsi dan Dekripsi File Dokumen. Eksplora Informatika, 8(1), 52. doi: 10.30864/eksplora.v8i1.139

[3] Phong, P., Dung, P., Tan, D., Duc, N., & Thuy, N. (2010). Password recovery for encrypted ZIP archives using GPUs. Proceedings Of The 2010 Symposium On Information And Communication Technology - Soict '10. doi: 10.1145/1852611.1852617

[4] Prasetyo, Sigit. 2011. Implementasi Algoritma Advanced Encryption Standard (Aes) Rijndael Untuk Proteksi File Audio. Skripsi. Program Ekstensi S1 Ilmu Komputer Fakultas Ilmu Komputer Dan Teknologi Informasi. Universitas Sumatera Utara: Sumatera Utara.

[5] B. Kaliski, "Password-Based Cryptography Specification," 2000.

[6] PAVLOV, Igor. 7-Zip [online]. 2019. Available also from: http://www.7-zip.org/. Accessed on 04/24/2019.

[7] Techtarget Network. [Online]. http://searchsecurity.techtarget.com/definition/Hash-based-MessageAuthentication-Code-HMAC

[8] P. K. Ketan and V. Vijayarajan, "An Amalgam Approach using AES and RC4 Algorithms for Encryption and Decryption," Int. J. Comput. Appl., vol. 54, no. 12, pp. 29–36, Sep. 2012.

[9] Michel Abdalla, Pierre-Alain Fouque, David Pointcheval, "PasswordBased Authenticated Key Exchange," in International Workshop on Theory and Practice in Public Key Cryptography

[10] Wennborg, H. (2020). Zip Files: History, Explanation and Implementation. Retrieved 25 May 2022, from https://www.hanshq.net/zip.html

[11] Huˇsek, Josef. The use of cryptography in 7-zip. Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2019.

[12] Hoxha, Nertila. (2017). Password Based Cryptography. Budapest. Retrieved from https://kgk.uni-obuda.hu/sites/default/files/11_Hoxha.pdf

[13] Kohno, T.: 'Analysis of the WinZip encryption method',IACR Cryptol. ePrintArch., 2004, 2004

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, 25 Mei 2022

Leony Angela
18219032