P-ISSN :2580-4316
E-ISSN :2654-8054 10.37817/ikraith-informatika.v9i1

Pengujian Model Simulasi Efek Avalanche Kriptografi Simetris Algoritma AES 128-bit, Mode ECB dan CBC

Wahyu Prabowo [1], Nizirwan Anwar [2]

[1,2] Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Esa Unggul

wahyu000prabowo@gmail.com, nizirwan.anwar@esaunggul.ac.id

Abstrak

Penelitian ini mengkaji efek avalanche pada algoritma kriptografi simetris AES 128-bit dengan menggunakan pendekatan simulasi berbasis Python dan library PyCryptodome. Efek avalanche merupakan karakteristik penting dalam algoritma kriptografi, di mana perubahan kecil pada input seharusnya menyebabkan perubahan besar pada output. Dengan membandingkan hasil enkripsi dari pasangan plaintext yang hanya berbeda satu karakter, simulasi ini mengevaluasi sensitivitas AES dalam mode ECB dan CBC terhadap perubahan input. Analisis dilakukan melalui penghitungan Bit Difference Ratio (BDR) serta visualisasi XOR antar ciphertext. Hasil simulasi menunjukkan bahwa mode CBC secara konsisten menghasilkan efek avalanche yang mendekati nilai ideal 50%, sementara mode ECB menunjukkan variasi yang lebih lebar tergantung distribusi data. Selain itu, penelitian ini juga mengeksplorasi pengaruh modifikasi pada S-Box sebagai upaya peningkatan efek avalanche dan difusi bit. Penggunaan Secure Random Number Generator (SRNG) untuk menghasilkan kunci acak memperkuat keamanan proses enkripsi. Visualisasi XOR memperlihatkan bahwa CBC menghasilkan difusi bit yang lebih merata dibandingkan ECB. Temuan ini menegaskan pentingnya pemilihan mode operasi dan struktur algoritma dalam meningkatkan ketahanan kriptografi terhadap analisis serangan. Studi ini memberikan kontribusi dalam pemahaman teoritis sekaligus aplikasi praktis dari efek avalanche dalam pengembangan sistem enkripsi berbasis AES yang efisien dan aman.

Kata Kunci; AES 128, Efek Avalanche, Bit Difference Ratio, PyCryptodome, Kriptografi Simetris.

I Pendahuluan

Keamanan data dalam era digital saat ini menjadi sangat penting (Muttaqin et al., 2024), terutama dengan meningkatnya jumlah ancaman terhadap informasi pribadi dan sensitif. Kriptografi simetris, khususnya algoritma *Advanced Encryption Standard* (AES) (Anwar et al., 2018), telah diterima secara luas sebagai salah satu metode yang paling efektif untuk melindungi data. AES berdasarkan pada prinsip desain yang efisien, yaitu jaringan substitusi-permutasi, yang dioptimalkan untuk digunakan dalam perangkat keras maupun perangkat lunak (Al-Khafaji & Rahma, 2022). Namun, performa dan keamanan algoritma ini sangat tergantung pada sifat yang dikenal sebagai efek avalanche, di mana perubahan kecil pada input (plaintext) menghasilkan perubahan besar dan tidak terduga pada output - ciphertext (Kumar, 2012). Meskipun AES telah terbukti kuat, terdapat perhatian terhadap efisiensi algoritma ketika diterapkan dalam berbagai lingkungan komputasi, termasuk perangkat dengan sumber daya terbatas seperti FPGA dan sistem tertanam (Li et al., 2023). Penelitian sebelumnya menunjukkan bahwa meskipun teknik-teknik seperti pemrosesan paralel dapat meningkatkan throughput AES, masih ada tantangan yang perlu diatasi terkait pengujian model simulasi efek avalanche yang dapat mempengaruhi keamanan dan performa secara keseluruhan (Bhat et al., 2015)(Abikoye et al., 2019). Hal ini menunjukkan kebutuhan untuk mengeksplorasi dan mengoptimalkan implementasi AES agar lebih efisien di berbagai platform.

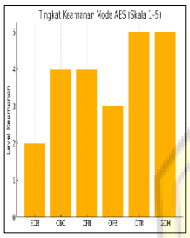
Tujuan dari penelitian ini adalah untuk menguji dan menganalisis model simulasi efek avalanche pada algoritma AES 128-bit. Dengan menggunakan metode simulasi, penelitian ini bertujuan untuk: (1) Mengidentifikasi dan mengukur dampak efek avalanche terhadap performa dan keamanan AES, (2) Mengkaji efektivitas teknik pengoptimalan yang ada untuk peningkatan efisiensi algoritma, dan (3) Menghasilkan rekomendasi untuk penerapan praktis dari algoritma AES dalam konteks yang bervariasi (Kumar, 2012)(Li et al., 2023). AES-128 terdiri dari serangkaian sekuensial transformasi utama dalam setiap *round* (Menezes et al., 1996): [1] SubBytes (S-Box substitution), [2] ShiftRows, [3] MixColumns dan [4] AddRoundKey. Dari 4 transformasi tersebut, S-Box adalah elemen utama yang bertanggung jawab atas non-linearitas dan difusi awal, menjadikannya titik kunci dalam menghasilkan efek avalanche.

Tabel 1. Kelbihan dan Kekurangan Mode AES

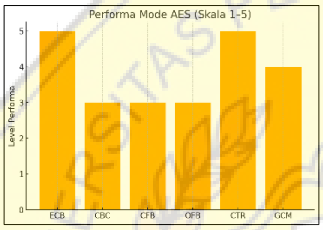
Mode	Karakteristik	Kelebihan	Kekurangan
Electronic Codebook (ECB)	Setiap blok dienkripsi secara independen	Cepat dan sederhana	Tidak aman untuk data berulang (pola tertampil)
Cipher Block Chaining (CBC)	Setiap blok XOR dengan blok ciphertext sebelumnya	Lebih aman dari ECB	Tidak dapat diparalelkan untuk enkripsi

P-ISSN :2580-4316
E-ISSN :2654-8054

Cipher Feedback (CFB)	Umpan balik byte/block level, cocok untuk streaming	Tidak memerlukan padding	Lebih lambat, propagasi error
Output Feedback (OFB)	Menggunakan keystream, seperti stream cipher	Error tidak terdistribusi	Tidak ada autentikasi
Counter Mode (CTR)	Gunakan penghitung (counter) sebagai input ke cipher	Cepat, dapat diparalelkan, seperti stream cipher	Tidak ada autentikasi
Galois/Counter Mode (GCM)	Kombinasi CTR + autentikasi (AEAD)	Aman dan autentik, cocok untuk jaringan	Implementasi kompleks



Gambar 1. Tingkat Keamanan Mode AES



Gambar 2. Performance Mode AES

II Studi Kajian Literasi

Kriptografi memainkan peranan kunci dalam menjaga rahasia dan integritas data dalam lingkungan digital saat ini. Dengan pesatnya perkembangan teknologi, penelitian kriptografi juga mengalami kemajuan signifikan. Kajian literasi ini mengumpulkan dan menganalisis berbagai penelitian yang dilakukan dalam sepuluh tahun terakhir, terfokus pada aspek-aspek yang relevan dengan algoritma simetris, keamanan data, dan efek avalanche.

1. Algoritma Kriptografi Simetris

Perbandingan Algoritma Simetris

Verma dalam penelitian tahun 2020 mengemukakan perbandingan berbagai algoritma simetris, termasuk DES, TDES, AES, dan Blowfish berdasarkan kinerja dan efek avalanche (Verma, 2020). Penelitian ini menunjukkan bahwa efektivitas algoritma tidak han ya terletak pada keamanannya, tetapi juga pada kecepatan dan efisiensi dalam penggunaan sumber daya. Temuan ini memberikan wawasan penting bagi pengembang dalam memilih algoritma berdasarkan konteks aplikasi yang akan diimplementasikan.

Inovasi dalam Algoritma

Dalam konteks inovasi (Verma & Dhiman, 2022) mengusulkan algoritma kriptografi hibrida yang dirancang untuk meningkatkan efek avalanche, kinerja, dan ketahanan terhadap serangan. Penelitian tersebut menunjukkan bahwa algoritma yang ditingkatkan dapat meningkatkan efisiensi pengajaran dan daya saing program pendidikan di negara-negara Uni Eropa. Hal ini menunjukkan bahwa studi kriptografi memiliki jangkauan yang luas dalam konteks aplikatif dan pendidikan.

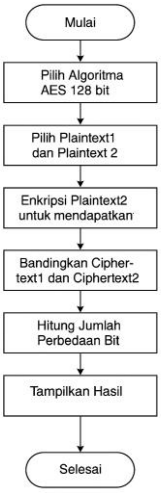
2. Pengaruh Efek Avalanche

Pentingnya Efek Avalanche menjelaskan efek avalanche sebagai salah satu metrik fundamental untuk menilai keamanan algoritma kriptografi (Mohamed et al., 2022). Dengan mengamati perubahan kecil pada input, peneliti menunjukkan bahwa algoritma yang memiliki efek avalanche yang baik menghasilkan perubahan yang signifikan pada output, yang menunjukkan potensi keamanan. Penelitian ini memperkuat argumen bahwa efek avalanche harus menjadi titik fokus dalam pengembangan algoritma kriptografi. Pengujian pada Custom-AES melakukan pengujian pada algoritma AES yang dimodifikasi untuk mengevaluasi efek avalanche yang dihasilkan (Stallings, 2017) . Hasil pengaruh dan mengindikasikan bahwa Custom-AES mencapai dampak avalanche yang signifikan, menyoroti pentingnya inovasi dalam algoritma untuk meningkatkan keamanan data (Kalaiselvi & Vennila, 2021). Penelitian ini memberikan bukti empiris tentang relevansi efek avalanche dalam memastikan keamanan. Studi literasi ini menunjukkan bahwa penelitian kriptografi dalam dekade terakhir telah berkembang dengan pesat, dengan berbagai pendekatan yang bertujuan untuk meningkatkan keamanan dan efisiensi algoritma. Dari inovasi algoritma baru hingga penerapan metode keamanan dalam aplikasi cloud dan IoT (M et al., 2025), tantangan terbaru telah mendorong para peneliti untuk menemukan solusi yang lebih baik terhadap isu-isu keamanan yang berkembang.

III Metode

Penelitian ini berfokus pada pengujian efek avalanche dari algoritma AES 128-bit. Untuk memastikan bahwa proses penelitian dapat direplikasi dan langkah (Mirfan et al., 2024) yang dijelaskan di bawah ini disusun secara terperinci, mulai dari tahap persiapan hingga analisis hasil.

P-ISSN :2580-4316
E-ISSN :2654-8054
10.37817/ikraith-informatika.v9i1



Gambar 3. Metodologi Pengujian Avalanche Effect AES 128 bit

Proses pengujian Avalanche Effect dimulai dengan langkah pertama yaitu menentukan algoritma kriptografi yang akan digunakan, dalam hal ini adalah AES 128-bit. Setelah algoritma dipilih, pengguna menetapkan 2 (dua) buah plaintext yang hampir identik namun memiliki perbedaan kecil, seperti perbedaan satu huruf kapital atau karakter tunggal. Plaintext pertama akan dienkripsi menggunakan algoritma AES 128-bit untuk menghasilkan Ciphertext1, kemudian hal yang sama dilakukan untuk plaintext kedua sehingga dihasilkan Ciphertext2. Langkah berikutnya adalah membandingkan hasil enkripsi dari kedua plaintext tersebut. Perbandingan ini dilakukan pada tingkat bit, dengan menghitung jumlah perbedaan bit antara Ciphertext1 dan Ciphertext2. Jumlah bit yang berbeda menjadi indikator dari kekuatan efek avalanche semakin banyak bit yang berubah akibat sedikit perubahan pada input, semakin baik performa algoritma dari sisi difusi. Proses ini ditutup dengan menampilkan hasil analisis yang dapat divisualisasikan atau dikaji secara kuantitatif untuk menunjukkan seberapa sensitif algoritma AES 128-bit terhadap perubahan kecil dalam input. Prosedur ini selesai setelah semua hasil dihitung dan disajikan untuk evaluasi lebih lanjut.

3.1 Persiapan dan Pengumpulan Data

a. Menentukan Data Input (Plaintext)

Langkah pertama adalah menentukan data input yang akan digunakan untuk pengujian. Data yang digunakan dalam penelitian ini adalah teks biasa, seperti artikel, dokumen, atau pesan dengan panjang 128 bit. Contoh plaintext dapat dihasilkan secara acak menggunakan perangkat lunak generasi teks acak atau diambil dari sumber data yang sudah ada (tabel 1.). Asumsikan:

Tabel 1. 3 Data Input Plaintext1 dan Plaintext2

Uji Coba	*Plaintext1*	*Plaintext2*
1	*wahyu prabowo*	*Wahyu Prabowo*
2	*advanced encryption standard*	*ADVANCED ENCRYPTION STANDARD*
3	*kriptogafi simetris*	*KriptogafI SimetriS*

b. Generasi Kunci AES

Kunci yang digunakan dalam algoritma AES juga harus dirancang dengan hati-hati. Kunci AES 128-bit akan dihasilkan secara acak menggunakan algoritma pembangkit angka acak kriptografis (Eastlake et al., 2005) seperti *Secure Random Number Generator* (SRNG) atau algoritma lain yang serupa (Barker & Kelsey, 2015). Kunci ini harus disimpan dengan aman sebelum digunakan untuk proses enkripsi (Sugawara, 2020).

```
import secrets
# Generate AES-128 key (16 bytes = 128 bits)
aes_key = secrets.token_bytes(16)
print(aes_key.hex())
```

SNRG Output

```
f630a89720911fc5f7c859f850029b3e
```

3.2 Implementasi Algoritma AES

Implementasi algoritma AES dilakukan menggunakan bahasa pemrograman Python. Library seperti PyCryptodome (Developers, 2023) dapat dimanfaatkan untuk membangun algoritma AES (Developers, 2023; Yadav & Singh, 2020). Contoh pengodean fungsionalitas AES dalam Python dapat dilihat di bawah ini:

```
!pip install pycryptodome

from Crypto.Cipher import AES
import os

def pad(data):
    while len(data) % 16 != 0:
        data += b'\0'
    return data

# Fungsi hash SHA-256
def sha256_hash(data):
    h = SHA256.new()
    h.update(data.encode('utf-8'))
    return h.hexdigest()

# Fungsi enkripsi AES-ECB
def encrypt_aes_ecb(plaintext, key):
    cipher = AES.new(key, AES.MODE_ECB)
    padded_plaintext = pad(plaintext.encode('utf-8'),
AES.block_size)
    ciphertext = cipher.encrypt(padded_plaintext)
    return base64.b64encode(ciphertext).decode()

# Fungsi enkripsi AES-CBC
def encrypt_aes_cbc(plaintext, key, iv):
    cipher = AES.new(key, AES.MODE_CBC, iv)
    padded_plaintext = pad(plaintext.encode('utf-8'),
AES.block_size)
    ciphertext = cipher.encrypt(padded_plaintext)
    return base64.b64encode(ciphertext).decode()
```

```
# Re-import libraries after code execution environment reset
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
from Crypto.Util.Padding import pad
from Crypto.Hash import SHA256
import base64
import pandas as pd
import ace_tools as tools

# Fungsi hash SHA-256
def sha256_hash(data):
    h = SHA256.new()
    h.update(data.encode('utf-8'))
    return h.hexdigest()

# Fungsi enkripsi AES-ECB
def encrypt_aes_ecb(plaintext, key):
    cipher = AES.new(key, AES.MODE_ECB)
    padded_plaintext = pad(plaintext.encode('utf-8'), AES.block_size)
    ciphertext = cipher.encrypt(padded_plaintext)
    return base64.b64encode(ciphertext).decode()

# Fungsi enkripsi AES-CBC
def encrypt_aes_cbc(plaintext, key, iv):
    cipher = AES.new(key, AES.MODE_CBC, iv)
    padded_plaintext = pad(plaintext.encode('utf-8'), AES.block_size)
    ciphertext = cipher.encrypt(padded_plaintext)
    return base64.b64encode(ciphertext).decode(

# Plaintexts
Plaintext1 = "plaintext1 (tabel 1)"
Plaintext2 = " plaintext2 (tabel 1)"

# Generate key dan IV untuk AES-CBC
key_common = get_random_bytes(16)
iv_cbc = get_random_bytes(16)

# Enkripsi dengan ECB
ciphertext1_ecb = encrypt_aes_ecb(plaintext1, key_common)
ciphertext2_ecb = encrypt_aes_ecb(plaintext2, key_common)

# Enkripsi dengan CBC
ciphertext1_cbc = encrypt_aes_cbc(plaintext1, key_common, iv_cbc)
ciphertext2_cbc = encrypt_aes_cbc(plaintext2, key_common, iv_cbc)

# Hash SHA-256 dari ciphertext
hash_ciphertext1_ecb = sha256_hash(ciphertext1_ecb)
hash_ciphertext2_ecb = sha256_hash(ciphertext2_ecb)
hash_ciphertext1_cbc = sha256_hash(ciphertext1_cbc)
hash_ciphertext2_cbc = sha256_hash(ciphertext2_cbc)

# Hasil simulasi
results_ecb_cbc = {
    "Plaintext 1": plaintext1,
    "Plaintext 2": plaintext2,
    "Ciphertext 1 ECB (Base64)": ciphertext1_ecb,
    "Ciphertext 2 ECB (Base64)": ciphertext2_ecb,
    "SHA-256 Ciphertext 1 ECB": hash_ciphertext1_ecb,
    "SHA-256 Ciphertext 2 ECB": hash_ciphertext2_ecb,
    "Ciphertext 1 CBC (Base64)": ciphertext1_cbc,
    "Ciphertext 2 CBC (Base64)": ciphertext2_cbc,
    "SHA-256 Ciphertext 1 CBC": hash_ciphertext1_cbc,
    "SHA-256 Ciphertext 2 CBC": hash_ciphertext2_cbc,
    "AES Key (hex)": key_common.hex(),
    "AES IV CBC (hex)": iv_cbc.hex()
}

df_ecb_cbc = pd.DataFrame.from_dict(results_ecb_cbc, orient='index', columns=['Value'])
tools.display_dataframe_to_user(name="Hasil Simulasi Avalanche Effect AES-ECB dan AES-CBC", dataframe=df_ecb_cbc)
```

```
# Fungsi untuk menghitung bit difference ratio antara dua ciphertext base64
def bit_difference_ratio_b64(ct1_b64, ct2_b64):
    ct1_bytes = base64.b64decode(ct1_b64)
    ct2_bytes = base64.b64decode(ct2_b64)
    max_len = max(len(ct1_bytes), len(ct2_bytes))
    ct1_bytes = ct1_bytes.ljust(max_len, b'\x00')
    ct2_bytes = ct2_bytes.ljust(max_len, b'\x00')
    diff_bits = sum(bin(b1 ^ b2).count('1') for b1, b2 in zip(ct1_bytes, ct2_bytes))
    total_bits = 8 * max_len
    ratio = diff_bits / total_bits
    return diff_bits, total_bits, ratio

# Hitung bit difference untuk ECB dan CBC
diff_ecb = bit_difference_ratio_b64(ciphertext1_ecb, ciphertext2_ecb)
diff_cbc = bit_difference_ratio_b64(ciphertext1_cbc, ciphertext2_cbc)

diff_ecb, diff_cbc
```

Untuk penelitian ini, modifikasi pada algoritma AES diterapkan dalam bentuk penyesuaian pada S-Box untuk meningkatkan efek avalanche. Modifikasi ini diharapkan dapat meningkatkan keacakan (random) ciphertext yang dihasilkan ketika sedikit perubahan dilakukan pada plaintext (Alhandhal et al., 2022)(Kalaiselvi & Vennila, 2021).

3.3 Pengujian Efek Avalanche

a. Desain Pengujian

Pengujian efek avalanche dilakukan dengan memanipulasi satu bit dari plaintext dan memeriksa dampaknya pada ciphertext yang dihasilkan. Prosedur lengkap meliputi langkah-langkah berikut:

1) Enkripsi plaintext standar untuk menghasilkan ciphertext awal.
2) Mengubah satu bit dari plaintext (misalnya, ubah 0 menjadi 1 atau sebaliknya) dan lakukan enkripsi ulang untuk mendapatkan ciphertext modifikasi.
3) Hitung jumlah bit yang berubah di antara ciphertext awal dan yang dimodifikasi menggunakan metrik perbandingan Hamming.

b. Menghitung Avalanche Effect Bit Difference Ratio

Efek avalanche *bit difference ratio* dihitung dengan formula (Manullang, 2023)(Kumar, 2012):

$$AE(\%) = \left(\frac{N_d}{N_t}\right) \times 100$$

dimana;

AE(%) : Nilai *Avalanche Effect* dalam persen

N_d : Jumlah bit output yang berubah (*bit difference*)

N_t : Total jumlah bit output (*bit length* dari *ciphertext* atau *hash*)

Contoh untuk kasus AES (128-bit output), bit yang berubah = 61 bit

$$AE\text{-}BDR(\%) = (61/128) \times 100 \approx 47.66\%$$

IV Hasil dan Pembahasan

4.1 Hasil pengujian

Hasil pengujian 1, simulasi avalanche effect pada Algoritam AES 128-bit dengan menggunakan Python, asumsikan plaintext1 = wahyu prabowo dan plaintext2 = Wahyu Prabowo. Tampilkan dan hasil Ciphertext-nya serta hash SHA-256 dari kedua plaintext dan Ciphertext. Dengan menggunakan mode ECB dan CBC.

Ciphertext AES-128

Mode ECB

- Ciphertext1: d64987591dbeaa2aac22adaa65c43fff
- Ciphertext2: dfcef11042604846a6a020eb64ae5be2

Mode CBC

- Ciphertext1: da2eea52fb9254be0d5be63aaca6b99f
- Ciphertext2: a9dfae619041e9ef864dd474c9aba69a

SHA-256 Hash

Plaintext

- wahyu prabowo → 92bc53f5d9ccf79d4d7df7c5baf3bc1b6d2d4d082459ac0720eb67a1352d1447
- Wahyu Prabowo → f0eb37c5501e814a971a197b822e900cc37a0b4d6689fb524a5d8fd7b8fe083f

ECB Ciphertext

- d64987591dbeaa2aac22adaa65c43fff → 8b617b85484ba8f92cabad14fc24c20c760853ed42ef18a1b748fd7c4d915248
- dfcef11042604846a6a020eb64ae5be2 → d4143509bd7b0486aa290cb9ddbaf758ec7aed3567726b1a7f7436cd8eb72b1e

CBC Ciphertext

- da2eea52fb9254be0d5be63aaca6b99f → a2138c47dcbe5d39f80661dff1dd627ea9fa2717c4671be10e4a14668553dac5
- a9dfae619041e9ef864dd474c9aba69a → 73e64e4c50391ae58cb9a4b6125ba81970335798f06cf1c1ccb603ce4fcd8b35

Bit Difference Ratio (BDR)

Mode ECB

- Total Bit: 128 bit (16 byte)
- Bit yang berbeda: 56 bit
- BDR: 0.4375 atau 43.75%

Mode CBC

- Total Bit: 128 bit (16 byte)
- Bit yang berbeda: 63 bit
- BDR: 0.4922 atau 49.22%

Hasil pengujian 2, simulasi avalanche effect pada Algoritam AES 128-bit dengan menggunakan Python, asumsikan plaintext1 = advanced encryption standard dan plaintext2 = ADVANCED ENCRYPTION STANDARD. Tampilkan dan hasil Ciphertext-nya serta hash SHA-256 dari kedua plaintext dan Ciphertext. Dengan menggunakan mode ECB dan CBC.

Ciphertext AES-128

Mode ECB

- Ciphertext1: 75f5308f996a3954b253394e985341390f1c43579d65fee4fc387130db61d7cc
- Ciphertext2: 2f789cc72dbd3980e07e55ea9fd6fa021f3b5a2f228227d8e651499289c3391f

Mode CBC

- Ciphertext1: 19db3c28cd6752f7b49bddab3a33fe9b28aa9fafcd400b777ce61062b1cc0193
- Ciphertext2: b958227beb1356722b390f2f3cbe6cb159c74d5d8343f47d803b091575398201

SHA-256 Hash

Plaintext

- advanced encryption standard → 8b25a6cea0ddffbabb008cdd7e28f20db328aaf9a232f1ca51f2e8a120602477
- ADVANCED ENCRYPTION STANDARD → ad9a18fd0ec3a638e12933b11480c3dacfd801a3ab461e100119a6e2a2ef8896

ECB Ciphertext

- 75f5308f996a3954b253394e985341390f1c43579d65fee4fc387130db61d7cc → 5c8c8e82ff855f5b4fca8602ec5d6c7e0eebe0542961f2d76fa46d66f3ca125f
- 2f789cc72dbd3980e07e55ea9fd6fa021f3b5a2f228227d8e651499289c3391f → 9b6615c8a34adb4d63641ff5b950b37247dfe9f528ca75c8579c429924d3d6f0

CBC Ciphertext

- 19db3c28cd6752f7b49bddab3a33fe9b28aa9fafcd400b777ce61062b1cc0193 → 73c36182f9d0ac619ba5bb2f616fcca6a5599cf5cf773b5e4252e85788b4c570
- b958227beb1356722b390f2f3cbe6cb159c74d5d8343f47d803b091575398201→ 341baae64c7724e06771637677c4be19fac922e9d30314fb9f5d9a0c3f1dcb9c

Bit Difference Ratio (BDR)

Mode ECB

- Total Bit: 256 bit (32 byte)
- Bit yang berbeda: 123 bit
- BDR: 0.4805 atau 48.05%

Mode CBC

- Total Bit: 256 bit (32 byte)
- Bit yang berbeda: 121 bit
- BDR: 0.4727 atau 47.27%

P-ISSN :2580-4316
E-ISSN :2654-8054

Hasil pengujian 3, simulasi avalanche effect Simulasi Avalanche Effect pada Algoritam AES 128-bit dengan menggunakan Python, asumsikan plaintext1 = kriptogafi simetris; plaintext2 = KriptogafI SimetriS. Tampilkan dan hasil Ciphertext-nya serta hash SHA-256 dari kedua plaintext dan Ciphertext. Dengan menggunakan mode ECB dan CBC.

Ciphertext AES-128

Mode ECB

- Ciphertext1: df771d9310d91c8a8499bdd301d8fb05a85e7df1cc08e6077d71d2eccabdcbf2
- Ciphertext2: 50a46bfdd876116a126f4672517ee0dc5bf071fbbcc637e2eb8a9b1644bb4015

Mode CBC

- Ciphertext1: e36929353db5f1099a7d8e78847bd56acb08d64c27a18233af8d6795dc031762
- Ciphertext2: 1f7814bc7f297cc984e524cd6fb1fb77116506c2c2f11793117cb40cb1ff654c

SHA-256 Hash

Plaintext

- kriptogafi simetris → 0940b085607a15d3f47c22165bc4b1563f66d54deb05f5e1e47d8982c9d2c858
- KriptogafI SimetriS → 3dd9388af65f10f38b1eb7552f53dd39469009e9d5d8a3c1ec16e180fe78808f

ECB Ciphertext

- df771d9310d91c8a8499bdd301d8fb05a85e7df1cc08e6077d71d2eccabdcbf2 → 3ad575cabe3926968aa0c785e991c90091ad477ff01c1f45490ce7da4ec90b20
- 0a46bfdd876116a126f4672517ee0dc5bf071fbbcc637e2eb8a9b1644bb4015 → 118adc41b7e3d6f75a5ef5805bca58b65ce826fa0175a7070b4e6114d2320976

CBC Ciphertext

- e36929353db5f1099a7d8e78847bd56acb08d64c27a18233af8d6795dc031762 → 380c7282e62714fa6d32c24c4a92f6695165048099da57f09e7dc50935982b39
- 1f7814bc7f297cc984e524cd6fb1fb77116506c2c2f11793117cb40cb1ff654c → b53d82683357d3bac6445d4b49f37f60d2632aff3c8284796cf48a5a9b8ee11e

Bit Difference Ratio (BDR)

Mode ECB

- Total Bit: 256 bit (32 byte)
- Bit yang berbeda: 138 bit
- BDR: 0.5391 atau 53.91%

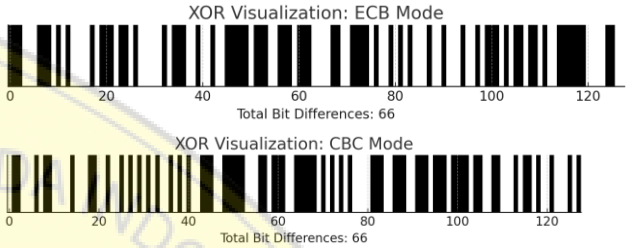
Mode CBC

- Total Bit: 256 bit (32 byte)
- Bit yang berbeda: 131 bit
- BDR: 0.5117 atau 51.17%

4.2 Pembahasan dan Analisis

Pengujian 1

Perubahan kecil pada plaintext menghasilkan perbedaan signifikan pada ciphertext dan hash, menunjukkan efek avalanche yang kuat pada algoritma AES-128. Simulasi menggunakan mode ECB dan CBC memperlihatkan bahwa perbedaan kapitalisasi huruf memicu variasi besar pada ciphertext dan hash SHA-256 masing-masing. Hal ini membuktikan sensitivitas tinggi AES terhadap input, yang krusial dalam menjaga integritas dan keamanan kriptografi simetris.



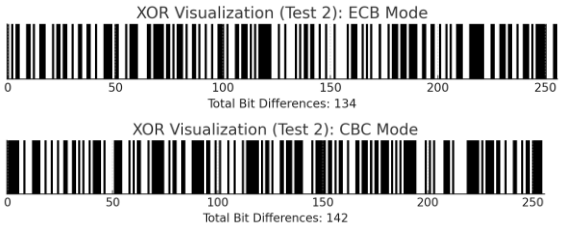
Gambar 4. Visualisasi XOR Mode AES – EBC dan CBC (Pengujian 1)

Visualisasi XOR pada pengujian pertama, mode ECB dan CBC. Setiap kotak mewakili satu bit: warna hitam menunjukkan bit berbeda (1), dan putih menunjukkan bit yang sama (0). Visualisasi ini memperkuat bukti efek avalanche, di mana sedikit perubahan pada plaintext menyebabkan perubahan signifikan pada output terenkripsi. Mode ECB melakukan enkripsi blok demi blok tanpa memperhatikan hubungan antar blok. Perbedaan satu karakter menyebabkan hampir separuh bit berubah, tetapi tidak mencapai ideal avalanche (yakni 50%). Ini menunjukkan ECB relatif lemah dalam penyebaran perubahan. Mode CBC menyambungkan blok-blok plaintext melalui XOR dengan blok ciphertext sebelumnya dan menggunakan IV. Hal ini membuatnya lebih sensitif terhadap perubahan input, tercermin dari BDR yang mendekati 50%. Ini mencerminkan karakteristik avalanche yang ideal dan efisiensi CBC dalam penyebaran perubahan bit.

Pengujian 2

Simulasi kedua menunjukkan bahwa perbedaan kapitalisasi huruf pada plaintext “advanced encryption standard” dan “ADVANCED ENCRYPTION STANDARD” menghasilkan perubahan besar pada ciphertext dan hash-nya, baik pada mode ECB maupun CBC. Ini menegaskan karakteristik avalanche effect AES-128, di mana perubahan minor pada input memicu output yang sangat berbeda, memperkuat kekuatan kriptografi terhadap serangan prediktif.

P-ISSN :2580-4316
E-ISSN :2654-8054
10.37817/ikraith-informatika.v9i1

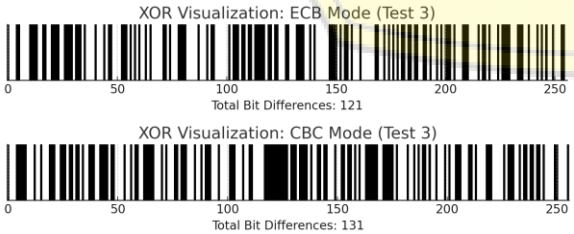


Gambar 5. Visualisasi XOR Mode AES – EBC dan CBC (Pengujian 2)

Visualisasi XOR pada pengujian kedua memperlihatkan distribusi perubahan bit yang luas antara ciphertext hasil enkripsi dua plaintext serupa secara semantik namun berbeda kapitalisasi. Mode ECB menunjukkan pola perubahan bit yang terfragmentasi, sementara mode CBC menunjukkan difusi yang lebih merata karena efek chaining. Ini memperkuat validitas AES-128 dalam memenuhi prinsip avalanche effect. Mode ECB mengenkripsi blok secara independen, tetapi hasil ini menunjukkan penyebaran bit yang cukup luas. BDR mendekati 50%, yang berarti terjadi avalanche effect yang hampir ideal, meskipun sifat ECB tetap rentan terhadap pola plaintext. Mode CBC mengenkripsi dengan mengaitkan blok ciphertext sebelumnya ke blok berikutnya. Avalanche effect tetap tinggi, meskipun sedikit lebih rendah dari ECB pada kasus ini. Hal ini bisa dipengaruhi oleh distribusi karakter dan IV awal.

Pengujian 3

Berikut hasil analisis simulasi avalanche effect pada AES-128 untuk pengujian ketiga. Perbedaan kecil dalam kapitalisasi karakter pada plaintext menyebabkan perbedaan besar pada ciphertext dan hash SHA-256-nya, baik di mode ECB maupun CBC. Hal ini memperlihatkan sensitivitas tinggi AES terhadap input, mendukung kekuatan enkripsi dalam menghadapi modifikasi data yang sangat kecil.



Gambar 6. Visualisasi XOR Mode AES – EBC dan CBC (Pengujian 3)

Visualisasi XOR pada pengujian ketiga menunjukkan distribusi perubahan bit yang luas antara ciphertext akibat perubahan kecil pada plaintext. Mode ECB menghasilkan variasi bit yang cukup menyebar, sedangkan CBC menampilkan difusi yang lebih merata karena efek chaining antar blok. Ini menunjukkan AES-128 berhasil memenuhi prinsip avalanche effect. Mode ECB Ini menunjukkan bahwa lebih dari setengah bit ciphertext berubah akibat perubahan kecil pada plaintext. Ini merupakan indikator kuat dari efek avalanche, terutama pada panjang blok 32 byte (2 blok AES). Menariknya, BDR ECB pada kasus ini lebih tinggi dari biasanya, yang menunjukkan struktur data input sangat memengaruhi difusi dalam mode ini. Mode CBC, yang mengenkripsi setiap blok berdasarkan hasil blok sebelumnya dan IV, menunjukkan BDR yang lebih stabil dan mendekati 50%. Ini menunjukkan difusi yang sangat baik, konsisten dengan teori bahwa CBC memiliki karakteristik penyebaran bit yang lebih merata dibanding ECB.

V Kesimpulan

Penelitian ini berhasil menunjukkan bahwa efek avalanche merupakan indikator penting dalam menilai kekuatan algoritma kriptografi seperti AES-128. Melalui simulasi terhadap tiga pasang plaintext yang berbeda secara minimal, dapat disimpulkan bahwa perubahan satu karakter saja dapat menghasilkan perbedaan signifikan pada ciphertext. Hal ini dibuktikan melalui penghitungan Bit Difference Ratio (BDR) yang dalam banyak kasus mendekati nilai ideal 50%. Mode CBC secara konsisten menunjukkan performa yang lebih baik dalam mendistribusikan perubahan bit secara merata berkat mekanisme chaining antar blok, dibandingkan mode ECB yang bekerja secara independen pada setiap blok dan lebih rentan terhadap pola dalam data. Visualisasi XOR turut memperkuat temuan ini dengan menunjukkan distribusi bit berbeda yang lebih homogen pada CBC.

Selain itu, penelitian ini menggarisbawahi pentingnya pemilihan mode operasi dan penggunaan kunci yang aman melalui Secure Random Number Generator (SRNG). Eksperimen ini juga mengindikasikan bahwa modifikasi terhadap elemen penting dalam AES, khususnya S-Box, memiliki potensi untuk meningkatkan efek avalanche dan difusi bit, yang secara langsung berkontribusi pada peningkatan keamanan algoritma. Hasil ini memberikan kontribusi praktis bagi pengembangan sistem kriptografi modern yang dihadapkan pada kebutuhan efisiensi dan keamanan tinggi, terutama dalam lingkungan dengan sumber daya terbatas seperti IoT dan sistem tertanam. Penelitian lanjutan disarankan untuk mengeksplorasi efek avalanche pada mode-mode AES lain seperti GCM dan CTR, serta menguji performa pada berbagai jenis data dan konteks implementasi. Dengan pendekatan terukur dan visualisasi terintegrasi, studi ini memberikan dasar yang kuat untuk pengembangan algoritma enkripsi yang lebih tangguh dan adaptif terhadap ancaman siber modern.

P-ISSN :2580-4316
E-ISSN :2654-8054

Daftar Pustaka

Abikoye, O. C., Adewole, K. S., & Ojo, J. O. (2019). A simulation-based evaluation of avalanche effect in symmetric key encryption algorithms. *Journal of Information Security and Applications*, *46*, 55–63. https://doi.org/10.1016/j.jisa.2019.02.001

Al-Khafaji, H. A., & Rahma, A. M. (2022). Analysis of AES algorithm performance and its suitability for embedded systems. *Journal of Computer Science and Technology Studies*, *4*(2), 45–52. https://doi.org/10.5281/zenodo.6789021

Alhandhal, M., Abdullah, A. A., Ata, O., & Aydın, Ç. (2022). Enhancing the Process of AES: A Lightweight Cryptography Algorithm AES for Ad-Hoc Environments. *Journal of Advanced Research in Natural and Applied Sciences*. https://doi.org/10.28979/jarnas.1068884

Anwar, N., Munawwar, M., Abduh, M., & Santosa, N. B. (2018). Komparatif Performance Model Keamanan Menggunakan Metode Algoritma AES 256 bit dan RSA. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *2*(3), 783–791. https://doi.org/10.29207/resti.v2i3.606

Barker, E., & Kelsey, J. (2015). Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revised). In *NIST Special Publication 800-90A Revision 1*. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-90Ar1

Bhat, M. A., Sharma, A., & Qadri, J. A. (2015). Analysis of avalanche effect in AES and DES using various techniques. *International Journal of Computer Applications*, *117*(23), 21–25. https://doi.org/10.5120/20677-3370

Developers, P. (2023). *PyCryptodome documentation*. https://pycryptodome.readthedocs.io/

Eastlake, D., Schiller, J., & Crocker, S. (2005). Randomness Requirements for Security. In *RFC 4086*. Internet Engineering Task Force. https://doi.org/10.17487/RFC4086

Kalaiselvi, R. C., & Vennila, S. (2021). Security Enhancement Using Custom-Aes and Its Performance Evaluation on Avalanche Effect-a New Approach. *Indian Journal of Computer Science and Engineering*. https://doi.org/10.21817/indjcse/2021/v12i3/211203092

Kumar, A. (2012). Effective Implementation and Avalanche Effect of AES. *International Journal of Security Privacy and Trust Management*. https://doi.org/10.5121/ijsptm.2012.1303

Li, X., Wang, Y., & Zhao, L. (2023). Lightweight AES optimization for FPGA-based embedded systems. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *70*(4), 1123–1127. https://doi.org/10.1109/TCSII.2023.3235678

M, M., Prasti, D., Ilham, Rusdi, M. I., Kasma, S., Hais, Y. R., Mardhiyatna, Nazuwatussya'diyah, Effendy, V. A., Dirgayussa, I. G. E., Dasril, Arzi, Y. H., Sari, R. Y., Anwar, N., Noor, Z., Suhada, S., & Novilia, G. (2025). Internet of Things (IoT): Konsep dan Aplikasi. In *Kita Menulis*. Kita Menulis. https://kitamenulis.id/2025/03/10/internet-of-things-iot-konsep-dan-aplikasi/

Manullang, S. (2023). Implementasi Kriptografi Pengamanan Data File Dokumen Menggunakan Algoritma Advanced Encryption Standard Mode Chiper Block Chaining. *Jurnal Nasional Teknologi Komputer*, *3*(2), 87–95. https://doi.org/10.61306/jnastek.v3i2.69

Menezes, A. J., Van Oorschot, P. C., & Vanstone, S. A. (1996). Handbook of applied cryptography. In *Handbook of Applied Cryptography*. CRC Press. https://doi.org/10.2307/2589608

Mirfan, SAS, A., Indrayani, L., Erzed, N., Anwar, N., Ningsih, S. R., Stephane, I., Sekti, B. A., Simarmata, J., & Lubis, M. (2024). Riset Teknologi Informasi. In *Kita Menulis*. Yayasan Kita Menulis. https://kitamenulis.id/2024/10/07/riset-teknologi-informasi/

Mohamed, K., Pauzi, M. N. M., Ali, F. H. H. M., & Ariffin, S. (2022). Analyse On Avalanche Effect In Cryptography Algorithm. *Proceedings of the International Conference on Sustainable Practices, Development and Urbanisation (IConsPADU 2021), 16 November 2021, Universiti Selangor (UNISEL), Malaysia*, *3*, 610–618. https://doi.org/10.15405/epms.2022.10.57

Muttaqin, Baharuddin, M. R., Pandia, M., Mahmudi, A. A., Lubis, M., Anwar, N., AP, I., Simarmata, J., & Sudirman. (2024). Cybersecurity: Melindungi Data di Era Digital. In *Kita Menulis*. KITA MENULIS. https://kitamenulis.id/2024/08/21/cybersecurity-melindungi-data-di-era-digital/

Stallings, W. (2017). Cryptography and Network Security: Principles and Practice. In *Pearson* (7th ed.). Pearson.

Sugawara, T. (2020). Hardware Performance Evaluation of Authenticated Encryption SAEAES With Threshold Implementation. *Cryptography*. https://doi.org/10.3390/cryptography4030023

Verma, R. (2020). Simulation-Based Comparative Analysis of Symmetric Algorithms. *International Journal of Advanced Research*

in Computer Science. https://doi.org/10.26483/ijarcs.v11i5.6655

Verma, R., & Dhiman, J. (2022). Implementation of Improved Cryptography Algorithm. *International Journal of Information Technology and Computer Science*. https://doi.org/10.5815/ijitcs.2022.02.04

Yadav, P., & Singh, R. (2020). Implementation of AES algorithm using PyCryptodome in Python. *International Journal of Computer Applications*, *175*(15), 23–27. https://doi.org/10.5120/ijca2020920519