JOCSIT, Vol. 1 No. 1, 2024 DOI: https://doi.org/10.69933/jocsit.v1i1.11
Journal of Collaborative Science and Informatics Technology
ISSN: 3048-1465 (Media Online)

IMPLEMENTATION OF AES AND SHA-3 CRYPTOGRAPHY ALGORITHMS IN SECURING USER SENSITIVE DATA ON THE ARTESIAN WATER BILL PAYMENT TRANSACTION WEBSITE

Muhamad Luthfi Assidiq[*1], Fathoni Mahardika[2], Deris Santika[3]

[1,2,3]Informatics, Information Technology Faculty, Universitas Sebelas April, Indonesia
Email: [1]luthfiassidiq12@gmail.com, [2]fathoni@unsap.ac.id, [3]deris@unsap.ac.id

(Article received: 13-05-2024; Revision: 02-06-2024; published: 02-06-2024)

Abstract

Engaging in online transactions has become familiar to almost all segments of society in the digital age. The advent of digital transactions brings convenience in conducting business activities in many fields. However, the use of the internet to facilitate transactions also comes with several challenges, one of which is hacking or cyber-attacks. In order to achieve secure transactions, the level of protection on websites providing digital transaction services must be made as strong as possible to prevent digital attacks. Therefore, a system of protection for user payment data on one of the transaction websites is created by implementing the AES and SHA-3 cryptographic algorithm methods. The AES algorithm method is chosen because of its uncomplicated implementation yet proven security, while SHA-3 is used to create unique keys from user passwords using the latest hash technology, making the encryption and decryption processes more protected by using a secret key for each user. By implementing cryptographic algorithms, additional protection will be provided against cyber-attacks on user transaction nominal data, ensuring that the data will never appear in plain form anywhere unless the user logs in with a valid account. This research will result in a flow of cryptographic program using the Python programming language and examples of the encryption and hash process outcomes during program execution.

Keywords: *AES, cryptographic, encryption, security system, SHA-3*

IMPLEMENTASI ALGORITMA KRIPTOGRAFI AES DAN SHA-3 DALAM MENGAMANKAN DATA SENSITIF PENGGUNA PADA WEBSITE TRANSAKSI PEMBAYARAN TAGIHAN AIR ARTESIS

Abstrak

Bertransaksi secara daring merupakan hal yang sudah tidak asing bagi hampir seluruh kalangan di era serba digital. Kehadiran transaksi digital membawa kemudahan dalam melaksanakan kegiatan berniaga dalam banyak bidang. Namun penggunaan internet dalam menciptakan transaksi yang serba mudah juga datang dengan beberapa tantangan, salah satunya adalah peretasan atau *hacking*. Demi mewujudkan transaksi yang aman, tingkat perlindungan pada situs penyedia layanan transaksi digital harus dibuat sekuat mungkin untuk mencegah serangan-serangan digital. Oleh karena itu dibuatlah suatu sistem perlindungan terhadap data pembayaran pengguna pada salah satu *website* transaksi dengan menerapkan metode algoritma kriptografi AES dan SHA-3. Metode algoritma AES dipilih karena implementasinya yang tidak rumit namun sudah terbukti keamanannya, sedangkan SHA-3 digunakan untuk membuat kunci yang unik dari *password* pengguna dengan teknologi *hash* terbaru, sehingga proses enkripsi dan dekripsi data menjadi lebih terproteksi karena menggunakan kunci rahasia setiap user namun tidak secara langsung. Dengan diterapkannya algoritma kriptografi akan memberikan perlindungan tambahan terhadap keamanan data nominal transaksi pengguna dari serangan siber sehingga data tersebut tidak akan pernah tampil dalam bentuk nyata di manapun kecuali pengguna melakukan *login* dengan akun yang sah. Penelitian ini akan menghasilkan suatu alur program kriptografi dengan bahasa pemrograman Python dan contoh hasil proses enkripsi dan *hash* saat eksekusi program.

Kata kunci: *AES, enkripsi, kriptografi, SHA-3, sistem keamanan*

1. Pendahuluan

Kemudahan dalam mengakses internet telah membawa perubahan secara signifikan terhadap berbagai peraturan hidup dan cara bekerja manusia[1]. Salah satu bidang yang terpengaruh besar dalam budaya masyarakat dengan teknologi internet yaitu bertransaksi. Cara bertransaksi yang memanfaatkan teknologi internet cenderung membawa kenyamanan tersendiri karena tidak adanya batas tempat dan waktu sehingga memberikan kebebasan pada kedua belah pihak dalam melakukan transaksi.

Pemanfaatan internet sebagai media transaksi merupakan suatu gebrakan yang sangat besar dan membawa perubahan pada pola berpikir masyarakat[2]. Perkembangan ini membuat banyak orang menikmati kenyamanan bertransaksi tanpa memikirkan kendala-kendala yang dapat muncul selama proses bertransaksi. Penggunaan media internet tidak selamanya menciptakan suatu proses transaksi yang aman, hal ini didasari pada banyaknya kasus kejahatan siber yang terjadi di internet khususnya di bidang yang berkaitan dengan keuangan.

Kejahatan siber bukan suatu hal yang asing bagi banyak pegiat maupun situs penyedia layanan transaksi, karena nyatanya telah banyak korban dari kejahatan ini yang dianggap sangat meresahkan kegiatan bertransaksi di internet[3]. Meskipun demikian, kegiatan transaksi di internet tidak pernah terhenti bahkan terus berkembang seiringan dengan penyedia layanan transaksi yang terus memperbaiki celah-celah keamanannya, hal tersebut dilakukan supaya penyedia layanan mendapatkan kembali kepercayaan pengguna dan tidak terulang lagi hal serupa di masa yang akan datang.

Untuk mendapatkan kepercayaan penuh dari pengguna, suatu *website* transaksi harus menyediakan pelayanan sebaik mungkin dengan tidak mengabaikan langkah-langkah keamanan yang dapat merusak nama baik produk atau *website*. Situs penyedia layanan transaksi harus siap dalam mencegah dan menangani segala bentuk kejahatan digital yang mungkin terjadi pada setiap *client* ataupun *database* perusahaan[4].

Pada penelitian ini, metode keamanan diimplementasikan pada suatu *website* pembayaran tagihan air artesis di salah satu perumahan di Sumedang. Website ini dibuat dengan tujuan utama mempermudah proses transaksi air artesis hanya menggunakan saldo digital. Namun *website* yang menyimpan nominal transaksi ini tidak mempunyai metode pengamanan apapun pada penyimpanan *database* di internet sehingga membuat data tersebut terlihat di *website* dalam keadaan rapuh. Maka atas alasan di atas, penulis menerapkan algoritma kriptografi dengan metode enkripsi yang dikombinasikan bersama metode *hash* untuk mengolah data tersebut supaya tersimpan di *database* dengan aman.

Metode pengamanan data yang digunakan pada penelitian ini yaitu algoritma enkripsi AES dan algoritma *hash* SHA-3. *Advanced Encryption Standard* atau AES merupakan algoritma kriptografi yang sudah umum digunakan untuk mengamankan data dengan cara melakukan enkripsi dan dekripsi data[5], [6]. Algoritma ini diterapkan pada data-data yang berhubungan dengan nominal pembayaran *user* sehingga data tersimpan dalam bentuk *ciphertext* dan tidak sembarang orang dapat melihat data tersebut[7]. Data yang telah tersimpan dalam bentuk teks enkripsi hanya dapat didekripsi oleh sistem pada saat sistem menerima kunci dekripsi. Penggunaan *password* sebagai kunci dekripsi merupakan cara yang paling aman karena setiap *password* akan menghasilkan kunci yang unik, hal ini memastikan bahwa *user* yang melakukan *login*

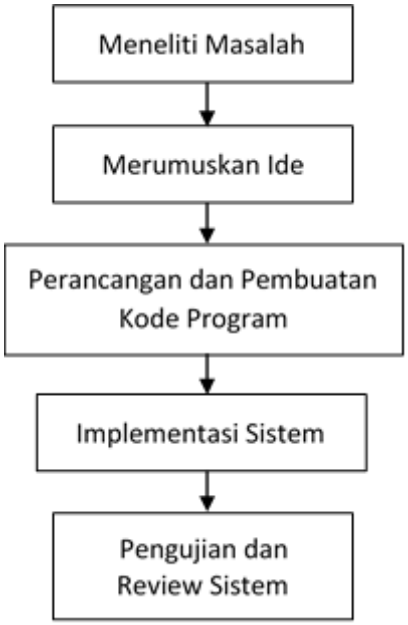
dengan valid yang hanya akan melihat datanya sendiri. Namun pertimbangan tersebut akan membuat *password user* menjadi tidak terjaga karena digunakan pada setiap proses dekripsi data yang terjadi di sistem, maka diimplementasikan pula algoritma *hash* yaitu *Secure Hash Algorithm 3* atau SHA-3 yang akan merubah *password* ke dalam bentuk *hash* sebelum digunakan di sisi *backend* sistem *website*.

Kunci dekripsi yang diambil dari *password user* akan diolah terlebih dahulu menggunakan algoritma SHA-3 untuk menjadikan kunci tersebut menjadi potongan baris *ciphertext* yang unik, hal ini dilakukan supaya kunci hanya bisa didapat dari pengolahan sistem dan hanya sistem yang bisa menggunakannya tanpa dapat dicampuri oleh tangan manusia karena mengingat bahwa *hash* merupakan algoritma yang cenderung tidak bisa diubah kembali ke bentuk aslinya[8], [9].

Penerapan kombinasi dari dua algoritma pada situs *web* transaksi air artesis bertujuan meminimalisir kebocoran data dan memperkecil celah-celah keamanan pada suatu situs *web* transaksi yang berada di internet. Metode enkripsi, dekripsi, dan *hash* yang diterapkan dapat mencegah beberapa metode peretasan yang terus berkembang hingga saat ini, seperti contohnya pemalsuan identitas (*spoofing*).

2. Metode

Penelitian tentang mengamankan data sensitif pengguna dengan metode kriptografi AES dan SHA-3 pada *website* transaksi berlangsung dengan beberapa tahapan yaitu meneliti masalah pada sistem, merumuskan ide penyelesaian masalah, membuat kode program, implementasi sistem baru, dan pengujian serta *review*.



Gambar 1. Alur Proses Tahapan Penelitian

2.1 Meneliti Masalah

Peneliti bekerjasama dengan *developer* dan *stackholder* dalam menganalisis kekurangan dan kelemahan *website* dari segi keamanan. Setelah peneliti beberapa kali melakukan *test run* dan *debugging website* air artesis, beberapa masalah keamanan

ditemukan dan dibandingkan untuk kemudian disimpulkan urutan prioritasnya. Penyimpanan data sensitif pengguna menjadi sorotan utama karena data tersebut berhubungan dengan data pribadi dan nominal pembayaran, bila data tersebut diubah oleh pihak yang tidak bertanggung jawab maka uang yang perlu dibayarkan warga akan berbeda dari tagihan aslinya.

2.2 Merumuskan Ide

Telah banyak metode pengamanan umum yang telah dipercaya dan tersebar di internet, salah satunya teknik enkripsi dan dekripsi dengan algoritma AES[10]. Algoritma AES dinilai cocok untuk mengamankan data dengan cara merubah format data yang tersimpan di *database* menjadi *ciphertext*, hal ini membuat data tidak akan mudah dilihat baik di sisi *database* maupun *website* bila diakses tanpa menggunakan proses *login* yang sah[11], [12]. Untuk melakukan proses dekripsi, selalu diperlukan suatu kunci unik pada setiap *user* dan kunci yang paling rahasia adalah *password* masing-masing *user*, namun untuk menghindari adanya kebocoran *password*, maka *password user* diubah terlebih dahulu ke dalam bentuk *hash* dengan metode SHA-3 sehingga *password* sebenarnya akan tetap aman dan tersembunyi.

2.3 Perancangan dan Pembuatan Kode Program

Website transaksi dirancang menggunakan bahasa Python dengan *framework* Django, ini membuat peneliti harus menyesuaikan dengan bahasa sistem yang telah ada supaya *website* tidak terbebani dengan menjalankan berbagai bahasa program. Kode program yang ditambahkan adalah *database* dengan kolom teks enkripsi, sistem enkripsi dan dekripsi dengan metode AES, dan menciptakan kunci dari *password user* dengan SHA-3.

2.4 Implementasi Sistem

Penerapan kode program keamanan ditambahkan dengan beberapa tahap penyesuaian terlebih dahulu sebelum logika enkripsi dan dekripsi dapat digunakan sepenuhnya pada alur kerja sistem. Pada tahap ini peneliti melakukan pembacaan kembali kode program dan membuat sinkronasi antara *backend website* dengan sistem enkripsi dan dekripsi.

2.5 Pengujian dan *Review* Sistem

Review dilakukan dengan cara menjalankan *website* secara utuh di sisi produksi, artinya *website* akan di*review*, *debug*, dan *testing* bersama pengembang *website* sebelum akhirnya akan dirilis untuk semua pengguna tanpa mengganggu proses transaksi *website* utama yang sedang berlangsung. Bila *website* baru telah diperiksa seutuhnya, maka *website* siap dilakukan *deploy* dan digunakan oleh semua pengguna.

3. Hasil dan Pembahasan

3.1. Pembuatan Kunci dengan SHA-3

Password yang diberikan *user* saat melakukan proses *login* akan diolah sintaks di bawah. Fungsi bernama *derive_key* akan memastikan kalau elemen *password* nyata

keberadaannya dan diinputkan secara benar oleh *user*. Setelah itu proses manipulasi *password* dimulai dengan mendeskripsikan metode yang akan dilakukan. Pada sintaks di bawah menggunakan metode *hash* SHA3_256 dengan 100.000 iterasi dan panjang 32 bit. Setelah dimanipulasi, kunci akan disimpan ke dalam variabel bernama *key*.

```
def derive_key(password):
    if password is None:
        raise ValueError("Password cannot be None for key derivation.")

    # Use SHA-3 directly for key derivation
    kdf = PBKDF2HMAC(
        algorithm=hashes.SHA3_256(),
        iterations=100000,
        salt=b'salt',
        length=32,
        backend=default_backend()
    )
    key = kdf.derive(password.encode())
    print(f'key: {key}')
    return key
```

Gambar 2. Sintaks Algoritma SHA-3

3.2 Enkripsi Data dengan AES

Sintaks pada gambar 3 digunakan untuk mengubah semua *input* teks menjadi *string* yang terenkripsi.

```
def encrypt_data(data, key):
    if data is None or key is None:
        raise ValueError("Data and key cannot be None for encryption.")

    # Pad the data before encryption
    padder = padding.PKCS7(algorithms.AES.block_size).padder()
    padded_data = padder.update(data.encode()) + padder.finalize()

    # Perform encryption
    cipher = Cipher(algorithms.AES(key), modes.ECB())
    encryptor = cipher.encryptor()
    ciphertext = encryptor.update(padded_data) + encryptor.finalize()

    return base64.urlsafe_b64encode(ciphertext).decode()
```

Gambar 3. Sintaks Enkripsi Data

Proses pemeriksaan dilakukan terlebih dahulu terhadap keutuhan data dan kunci, bila keduanya terpenuhi maka program enkripsi bisa dijalankan. Proses pra-enkripsi dilakukan untuk menentukan jenis algoritma yang digunakan dan panjang ukuran blok data, lalu data disimpan sementara untuk siap dilakukan manipulasi. Kemudian proses enkripsi dimulai dengan meminta kunci enkripsi dan sinkronasi kunci tersebut dengan *encryptor*, bila kunci valid maka data tersebut dimanipulasi ke dalam bentuk *ciphertext*.

3.3 Tempat Penyimpanan Hasil Enkripsi di *Database*

Ciphertext hasil manipulasi sistem enkripsi disimpan ke *database* dengan tipe *Char* sehingga data tersebut berbentuk *string*. Sintaks untuk membuat penyimpanan data adalah seperti gambar 4.

```
# encrypted here
encrypted_pemakaian_kubik_bulanan = models.CharField(max_length=500, blank=True, null=True)
encrypted_biaya_pemakaian_bulanan = models.CharField(max_length=500, blank=True, null=True)
encrypted_biaya_total_bulanan = models.CharField(max_length=500, blank=True, null=True)
```

Gambar 4. Sintaks Membuat Penyimpanan Enkripsi di *Database*

3.4 Eksekusi Sistem Enkripsi

Selanjutnya menjalankan proses *function* yang akan melakukan semua proses manipulasi data menjadi berbentuk enkripsi secara *realtime*. Semua proses kerja yang ada di atas dieksekusi dengan memastikan beberapa kondisi terlebih dahulu seperti ketersediaan data baru yang akan dienkripsi dan data hasil enkripsi sebelumnya, bila hasil enkripsi sebelumnya telah ada maka data tersebut akan digantikan dengan data yang lebih baru.

```
def encrypt_pemakaian_kubik_bulanan(self):
    # Check if the field is not already encrypted to avoid re-encryption
    if self.pemakaian_kubik_bulanan is not None and self.encrypted_pemakaian_kubik_bulanan is None:
        key = derive_key(self.password)
        encrypted_data = encrypt_data(str(self.pemakaian_kubik_bulanan), key)
        self.encrypted_pemakaian_kubik_bulanan = encrypted_data
    elif self.pemakaian_kubik_bulanan is not None and self.encrypted_pemakaian_kubik_bulanan is not None:
        key = derive_key(self.password)
        encrypted_data = encrypt_data(str(self.pemakaian_kubik_bulanan), key)
        self.encrypted_pemakaian_kubik_bulanan = encrypted_data
```

Gambar 5. Sintaks Menjalankan Enkripsi Data

Data hasil enkripsi tersebut akan disimpan ke *field* di *database* dengan sintaks seperti di bawah. Sintaks di bawah menyimpan tiga data enkripsi sekaligus ke dalam tiga kolom *database* secara bersamaan.

```
def save(self, *args, **kwargs):
    self.hitung_pemakaian_bulanan()
    self.hitung_total_bulanan()
    self.encrypt_pemakaian_kubik_bulanan()
    # print(f"Encrypted data: {self.encrypted_pemakaian_kubik_bulanan}")
    self.encrypt_biaya_pemakaian_bulanan()
    # print(f"Encrypted data: {self.encrypted_biaya_pemakaian_bulanan}")
    self.encrypt_biaya_total_bulanan()
    # print(f"Encrypted data: {self.encrypted_biaya_total_bulanan}")
    super().save(*args, **kwargs)
```

Gambar 6. Sintaks Menyimpan dan Memperbaharui *Database*

3.5 Dekripsi Data dengan AES

Untuk melihat data yang sebenarnya pada tampilan *admin* maupun tampilan

pengguna, maka teks enkripsi perlu dilakukan proses dekripsi terlebih dahulu.

```
def decrypt_data(data, key):
    if data is None or key is None:
        raise ValueError("Data and key cannot be None for decryption.")

    # Perform base64 decoding
    decoded_data = base64.urlsafe_b64decode(data)

    # Perform decryption
    cipher = Cipher(algorithms.AES(key), modes.ECB())
    decryptor = cipher.decryptor()
    decrypted_data = decryptor.update(decoded_data) + decryptor.finalize()

    # Unpad the decrypted data
    unpadder = padding.PKCS7(algorithms.AES.block_size).unpadder()
    unpadded_data = unpadder.update(decrypted_data) + unpadder.finalize()

    return unpadded_data.decode()
```

Gambar 7. Sintaks Dekripsi Data

Proses dekripsi data membutuhkan beberapa persyaratan sebelum bisa berjalan yaitu data yang ingin didekripsi dan kunci yang sama dengan kunci saat melakukan enkripsi, itu artinya *password* yang diinputkan oleh pengguna harus tepat dan valid saat ingin melihat data dalam versi sebenarnya. Kemudian data enkripsi di *database* akan dipanggil dan diolah kembali dengan AES untuk menghasilkan data yang asli.

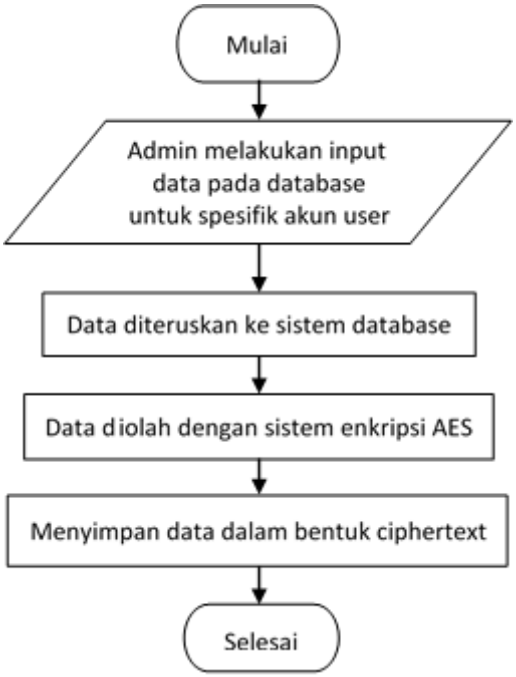
```
def get_decrypted_pemakaian_kubik_bulanan(self):
    # Decrypt the data only if it's encrypted
    if self.encrypted_pemakaian_kubik_bulanan is not None:
        key = derive_key(self.password)
        decrypted_data = decrypt_data(self.encrypted_pemakaian_kubik_bulanan, key)
        return int(decrypted_data)
```

Gambar 8. Sintaks Menjalankan Dekripsi Data

Pada gambar 8, fungsi dekripsi dijalankan dengan mengambil kunci dan data yang berbentuk enkripsi dari *database*, kemudian bila kunci tersebut berhasil melewati tahap pemeriksaan, maka fungsi ini akan menghasilkan data yang telah terdekripsi, tapi bila kunci enkripsi tidak sama seperti saat proses enkripsi maka akan mengalami *error*.

3.6. *Flowchart* Enkripsi Data

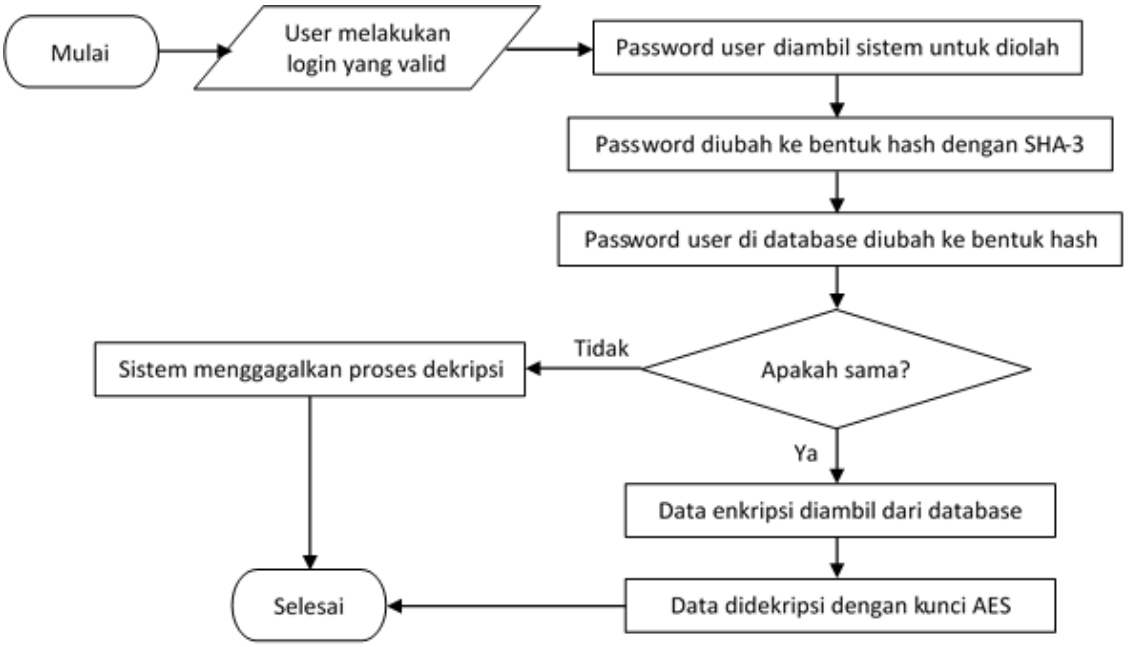
Mengubah data ke dalam bentuk enkripsi dengan AES dimulai dengan *admin* yang bertanggung jawab untuk menginputkan data setiap *user*, kemudian data diolah sistem algoritma AES yang ada di *backend*, dan akhirnya data yang berbentuk *ciphertext* disimpan ke *database*.



Gambar 9. Diagram Alir Enkripsi Data

3.7 *Flowchart* Dekripsi Data

Untuk mengubah kembali data yang ada di *database* ke dalam bentuk sebenarnya, maka dibutuhkan *password* pemilik akun. *Password* akun akan digunakan sebagai kunci dekripsi data AES, namun untuk mengamankan kunci akun dilakukan fungsi *hash* terlebih dahulu pada *password* tersebut dengan algoritma SHA-3.



Gambar 10. Diagram Alir Dekripsi Data

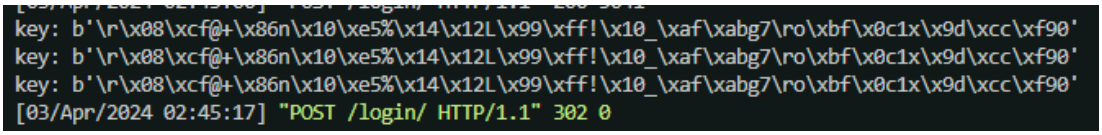
3.8 Data Hasil Enkripsi di *Database*

Data hasil enkripsi akan terlihat pada *database* dengan bentuk *ciphertext* yang telah berupa enkripsi AES seperti gambar berikut.

Gambar 11. Data Hasil Enkripsi di *Database*

3.9. Proses Pemanggilan Data di *Backend*

Saat *user* melakukan *login* maka terdapat beberapa baris proses seperti gambar di bawah pada *terminal backend website*.



Gambar 12. Proses Pemanggilan Fungsi Dekripsi

Baris-baris tersebut adalah proses pembuatan kunci pada fungsi dekripsi, kunci yang dibuat dari *password user* sengaja ditampilkan khusus pada penelitian ini untuk mengetahui bentuk sebenarnya kunci dekripsi yang berbentuk SHA-3. Terlihat kalau *string* hasil SHA-3 di atas berbentuk potongan *hash* yang acak dan akan berbeda berdasarkan dari *password user* yang ingin melakukan *login*. Bila proses dekripsi ini berjalan lancar, maka pengguna akan diperlihatkan data nominal pembayaran mereka.

3.10 Pengujian Sistem Keamanan

Data yang akan ditampilkan pada sisi pengguna adalah seperti gambar di bawah. Nominal harga yang tertulis adalah hasil dari dekripsi AES.

Gambar 13. Penyajian Data Nominal Pembayaran *User*

Pengujian keamanan ini diterapkan pada nominal pembayaran karena data yang berhubungan dengan uang merupakan data sensitif yang menjadi rahasia masing-masing pengguna. Dalam praktik sebenarnya, sudah sepantasnya kalau suatu *user* tidak bisa melihat halaman pembayaran dan nominal *user* lainnya, hal tersebut merupakan dasar keamanan yang harus dimiliki setiap *website* transaksi atau bahkan pengamanan dasar dari setiap *framework* pemrograman tak terkecuali Django yang digunakan sebagai *framework website* ini.

Namun untuk kepentingan penelitian, keamanan *website* diuji lebih lanjut dengan mencoba melihat data pengguna lain dengan tidak melakukan *login* yang valid. Cara ini kerap disebut *spoofing*, yaitu mengidentifikasi diri sebagai orang lain dengan memalsukan data untuk *login* dan melihat data *user* lain secara illegal. Dengan salah satu teknik *hacking* ini, memungkinkan suatu *user* menjadi *user* lainnya ataupun menjadi *admin* sehingga data dapat dengan mudah dilihat dan dimanipulasi oleh pihak yang tidak bertanggung jawab.

Gambar 14. Penyajian Data saat Dilakukan *Spoofing*

3.11 Hasil Penerapan Enkripsi dan Dekripsi AES-128

Dalam membuktikan keunikan kode enkripsi yang dibuat oleh AES, peneliti mengambil beberapa akun acak pada *server* dengan total biaya bulanan yang asli dan hasil enkripsinya. Terlihat pada tabel 1 kalau hasil enkripsi tidak memiliki kesamaan meskipun total pembayaran yang harus dibayarkan memiliki nominal yang sama.

Tabel 1. Data Hasil Manipulasi AES

USERID	TOTAL BULANAN	ENCRYPTED TOTAL BULANAN
D10	20000	pEvDM376ljNDe--QNO22PA==
C23	20000	Z6i8e_EZTWvYAyXmXbwBTA==
C12	86000	0P4Wra8IHPgWYjsZQyiN7w==
E10	36500	Re1oSMJtr74ppTjoMciMoQ==
E03	58500	G40qdolkl_hlp90KuFpePw==

3.12 Hasil *Generate Data* SHA3-256 dari *Password* Pengguna

Kunci yang berasal dari *password* pengguna merupakan syarat utama supaya proses dekripsi berjalan. Pada tabel 2 diambil *sample* beberapa *password* milik *admin* dan akun *dummy* yang dibuat pada saat penelitian ini dilakukan. Kode SHA3_256 yang dibuat oleh sistem *hash* tidak dapat dibaca karena susunan kode yang sangat acak, oleh karena itu akan sangat sulit untuk menggunakan kunci enkripsi dan dekripsi selain oleh sistem secara langsung.

Tabel 2. Data Hasil Manipulasi SHA-3

USER PASSWORD	SHA3_256
admin	\r\x08\xcf@+\x86n\x10\xe5%\x14\x12L\x99\xff!\x10_\xaf\xab g7\ro\xbf\x0c1x\x9d\xcc\xf90
adminadmin	\xb0\xb1\xae\xc6H\xef\x1e\x12\xc0K\x87\xe8/\n\xe3\xe6~#\x 038a^#\x825\x05\xad\x8fi\xf9@?
uQJDCvIUTx	\x16\x01\xacpw%9@\xbd3\xd3\xbc\xcf\xa69X\xfa{\xb8\xa9!\x 9d\x9bL\x1d#\x8a\xcfz\x0c.\x1b
uXvkAflGvI	\xf9K\xd7\xbe\x94M\xd4\xd1k)\x1fUf\x052\xc3\x0fB\xc0\r\x1 1\xe87\x95\x9b: \x86\xf4\xa8\x9c\xd5
hqhwhkXLRC	d\xf0\x95\xf5\x89\xe2\xd3\xd6\xa2\xa2z\xaa\xf8\xf8O\xb1@\ xb0\xef*\xb8\xfd\xda+;\xbbK\x1b\xb6\xa1\x80\x99

4. Diskusi

Penelitian yang dilakukan mencapai hasil yang diharapkan dari rencana yang telah disusun oleh peneliti, yaitu menerapkan suatu sistem keamanan kriptografi pada data sensitif pengguna. Adapun teknik kriptografi yang digunakan pada penelitian ini adalah AES dan SHA-3, AES merupakan teknik pengamanan yang sudah umum digunakan dan dapat dibilang sederhana sedangkan SHA-3 yang merupakan teknik *hash* terbaru yang digunakan peneliti karena akan menutup sisi kelemahan AES yang sudah terlalu banyak digunakan dan mungkin sudah dapat diretas dengan cepat[13], [14]. SHA-3 bertujuan untuk menyembunyikan *password user* yang digunakan sebagai kunci unik dekripsi ke dalam bentuk *ciphertext* supaya *password* asli tidak digunakan selama proses dekripsi.

Dalam penelitian ini, AES sengaja dipilih supaya tidak membebani sistem terlalu besar, maka untuk menambah keamanannya dilakukan proses kriptografi yang mutakhir untuk kunci enkripsi dan dekripsi, karena kunci hanya terdiri dari satu kata dari *password user*, maka beban yang diperlukan untuk melakukan enkripsi pada kunci tidak terlalu berat. Namun disamping itu sistem ini tetap menambah waktu kinerja sistem saat pengguna mencoba *login* sekitar 3 hingga 5 detik. Adapun sistem kriptografi yang diterapkan pada *website* telah melewati tahap pengujian dengan tes *spoofing* sederhana secara manual, yaitu menggunakan identitas palsu untuk melihat data yang ada di dalam suatu akun, dan cara ini hanya membuat tampilan *user* terbuka namun dengan data yang masih berbentuk *ciphertext* karena tidak mendapatkan kunci dekripsi yang sesuai.

Penerapan algoritma kriptografi AES dan SHA-3 secara bersamaan untuk mengamankan data juga diterapkan oleh S. Praptodiyono, M. A. Sidiq, and F. Muhammad dalam penelitiannya yang berjudul "Implementasi Algoritma SHA-3 Dan AES Sebagai Sistem Keamanan Pada Proses Pensinyalan *Mobile* IPv6". Dalam penelitian tersebut membahas tentang penggunaan algoritma AES dan SHA-3 yang diterapkan pada IPSec dalam mengamankan data berupa sinyal *Mobile* IPv6 (MIPv6) pada saluran komunikasi karena memiliki risiko terkena serangan. Penggunaan AES dan SHA-3 yang diterapkan pada IPSec dipilih karena tingkat keamanannya yang dinilai unggul, berdasarkan pengujian yang dilakukan pada penelitian tersebut, perlu waktu 2.7 x 10^25 tahun dan 6.2 x 10^21 tahun untuk menembus sistem AES dan SHA-3 dengan serangan *brute-force*[9]. Hal ini juga menjadi landasan kepercayaan peneliti untuk menggunakan metode kriptografi AES dan SHA-3 secara bersamaan dalam mengamankan data pengguna pada situs transaksi.

Selain itu, implementasi metode keamanan yang serupa juga diterapkan pada penelitian yang diteliti oleh E. Ramadhan, A. Rizkiana, P. Wiyardhana, dan D. Ogi yang berjudul "*Secure Self Payment and Monitoring Service Laundry using RFID with SHA-3 and AES 128*". Pada penelitian tersebut membahas tentang penggunaan algoritma AES-128 dan SHA-3 untuk mengamankan data-data yang berhubungan dengan pembayaran *user*, data *payment* dari *e-money* yang diamankan bertujuan untuk menghindari pengkloningan data sehingga data disimpan dalam bentuk *ciphertext*[15]. Perbedaan penelitian yang dilakukan dengan penelitian ini adalah kunci yang dienkripsi dengan SHA-3 berasal dari UID setiap *user*, sedangkan penelitian yang dilakukan pada *website* transaksi oleh peneliti yaitu menggunakan kunci yang didapat dari *password*.

5. Kesimpulan

Keamanan data sensitif pengguna pada *website* transaksi adalah hal yang sangat penting, karena *website* transaksi yang melayani perihal keuangan menjadi sasaran utama pelaku kejahatan siber, maka diterapkan proteksi awal algoritma kriptografi AES dan SHA-3 untuk menyimpan data dalam bentuk yang acak dan tidak bisa dibaca. Algoritma AES digunakan untuk enkripsi dan dekripsi data nominal pembayaran ke bentuk *ciphertext*, namun AES membutuhkan kunci saat melakukan prosesnya, maka dari itu digunakan algoritma *hash* SHA-3 untuk membuat kunci yang kuat dengan mengubah *password user* menjadi susunan *hash* acak. Algoritma AES yang digunakan adalah AES-128 yang sudah sangat sering digunakan namun masih dipercaya karena terbukti keamanannya. Sedangkan *hash* yang digunakan adalah SHA3-256 yang berasal dari keluarga SHA terbaru dengan nama lain Keccak.

Penelitian ini menghasilkan suatu sistem yang dapat memberikan proteksi awal dari beberapa metode kejahatan *hacking* pada dunia digital contohnya seperti pemalsuan identitas atau *spoofing*, untuk membuktikan keamanannya dilakukan simulasi *hacking spoofing* secara sederhana pada penelitian ini untuk menguji keamanan *website*, dan penelitian menghasilkan suatu bukti nyata bahwa data tetap berada dalam keadaan terenkripsi. Kemudian, peneliti melakukan pengumpulan data hasil uji coba kriptografi AES dan SHA-3 untuk melihat seberapa unik kode yang dibuat oleh sistem keamanan data, dan data hasil modifikasi tidak memiliki kesamaan.

Berasarkan beberapa penelitian dan pengalaman dari penulis tentang

pengamanan *website* transaksi dengan algoritma AES dan SHA, terdapat beberapa saran mengenai penelitian berikutnya. Pertama, Metode pengamanan AES-128 dianggap aman namun menggunakan bit yang kecil supaya tidak membebani sistem, maka ada baiknya bila meningkatkan bit yang digunakan saat metode enkripsi dan dekripsi data. Lalu, penelitian lebih lanjut dapat meneliti lebih banyak studi kasus kejahatan *hacking* untuk lebih memperluas cakupan metode pengamanan dan menerapkan pengamanan pada lebih banyak data sehingga dapat membuktikan efektifitas kedua algoritma ini. Selanjutnya, Keamanan AES dan SHA-3 dapat digunakan juga untuk penggunaan pada selain *website* dan platform lain dengan bahasa pemrograman serta data yang berbeda. *Syntax* pun dapat diubah pula sedemikian rupa untuk meringankan beban kerja *website*.

DAFTAR PUSTAKA

[1] D. Rifai, S. Fitri, I. N. Ramadhan, dan R. Ramadan, "Perkembangan Ekonomi Digital Mengenai Perilaku Pengguna Media Sosial Dalam Melakukan Transaksi", *ABDI*, vol. 3, no. 1, pp. 49–52, Jun. 2022, doi: 10.34306/abdi.v3i1.752.

[2] E. Purike, I. W. Kurniasih, F. W. Wulandari, and A. Nirwani, "TRANSAKSI DIGITAL DAN PERKEMBANGAN e-TOURISM DI INDONESIA," *NAWASENA*, vol. 1, no. 2, pp. 12–19, Aug. 2022, doi: 10.56910/nawasena.v1i2.157.

[3] S. Parulian, D. A. Pratiwi, and M. C. Yustina, "Ancaman dan Solusi Serangan Siber di Indonesia," *Telecommunications, Networks, Electronics, and Computer Technologies (TELNECT)*, vol. 1, no. 2, pp. 85–92, Dec. 2021, doi: 10.17509/telnect.v1i2.40866.

[4] A. Pambudi, A. Kusyanti, and M. Data, "Perancangan Sistem Pengamanan Data Transaksi Pada Database Terdistribusi Menggunakan Metode Hashing," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 3, no. 1, pp. 247–252, 2019, [Online]. Available: https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/4085.

[5] N. W. Hidayatulloh, M. Tahir, H. Amalia, N. A. Basyar, A. F. Prianggara, and M. Yasin, "Mengenal Advance Encrytion Standard (AES) Sebagai Algoritma Kriptografi Dalam Mengamankan Data," *Digital Transformation Technology (Digitech)*, vol. 3, no. 1, pp. 1–10, Mar. 2023, doi: 10.47709/digitech.v3i1.2293.

[6] M. B. Aryanto, M. Tahir, S. I. Devita, Z. N. Mustofa, Q. Ainiyah, and S. Sundoro, "Implementasi Enkrip Dan Dekrip File Menggunakan Metode Advance Encryption Standard (AES-128)," *JUISIK: JURNAL ILMIAH SISTEM INFORMASI DAN ILMU KOMPUTER*, vol. 3, no. 1, pp. 89–104, Mar. 2023, doi: 10.55606/juisik.v3i1.434.

[7] L. Mustika, "Implementasi Algoritma AES Untuk Pengamanan Login Dan Data Customer Pada E-Commerce Berbasis Web," *JURIKOM (Jurnal Riset Komputer)*, vol. 1, no. 7, pp. 148-155, 2020, doi: 10.30865/jurikom.v7i1.1943.

[8] M. P. Sari, "Analisis Algoritma SHA-3 Keamanan pada Data Pribadi," *TECHNOSCIENZA*, vol. 5, no. 2, pp. 231–242, Apr. 2021, doi: 10.51158/tecnoscienza.v5i2.429.

[9] S. Praptodiyono, M. A. Sidiq, and F. Muhammad, "Implementasi Algoritma SHA-3 Dan AES Sebagai Sistem Keamanan Pada Proses Pensinyalan Mobile IPv6," *Setrum : Sistem Kendali-Tenaga-elektronika-telekomunikasi-komputer*, vol. 10, no. 2, pp. 59–68, Nov. 2021, doi: 10.36055/setrum.v10i2.13075.

[10] N. Cristy and F. Riandari, "Implementasi Metode Advanced Encryption Standard (AES 128 Bit) Untuk Mengamankan Data Keuangan," *JIKOMSI [Jurnal Ilmu Komputer dan Sistem Informasi]*, vol. 4, no. 2, pp. 75–85, Sep. 2021, doi: 10.9767/jikomsi.v4i2.181.

[11] M. Azhari, F. J. Perwitosari, D. I. Mulyana, and F. Ali, "Implementasi Pengamanan Data pada Dokumen Menggunakan Algoritma Kriptografi Advanced Encryption Standard (AES)," *Jurnal Pendidikan Sains dan Komputer*, vol. 2, no. 1, pp. 163–171, Feb. 2022, doi: 10.47709/jpsk.v2i1.1390.

[12] J. S. Sianipar, N. B. Nuugroho, and I. Mariami, "Pengamanan Data Gaji Karyawan Dengan Menggunakan Metode Advanced Encryption Standard (AES)," *JURNAL SISTEM INFORMASI TGD*, vol. 3, no. 1, pp. 35–45, Jan. 2024, doi: 10.53513/jursi.v3i1.5653.

[13] A. Dharmawan and H. Munandar, "PENERAPAN ALGORITME KRIPTOGRAFI SHA-256 DAN AES-256 UNTUK PENGAMANAN FILE PADA PT PELANGI SENTRAL KREASI," *Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)*, vol. 2, no. 2, pp. 186–195, Sep. 2023, [Online]. Available: https://senafti.budiluhur.ac.id/index.php/senafti/article/view/857.

[14] Y. Putra, Y. Yuhandri, and Sumijan, "Meningkatkan Keamanan Web Menggunakan Algoritma Advanced Encryption Standard (AES) terhadap Seragan Cross Site Scripting," *Jurnal Sistim Informasi dan Teknologi*, vol. 3, no. 2, pp. 56–63, Jun. 2021, doi: 10.37034/jsisfotek.v3i2.44.

[15] E. Ramadhan, A. Rizkiana, P. Wiyardhana, and D. Ogi, "Secure Self Payment and Monitoring Service Laundry using RFID with SHA-3 and AES 128," *2023 IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs)*, pp. 41-46, Aug. 2023, doi: 10.1109/ICoCICs58778.2023.10277534.