Implementasi Algoritma Kriptografi AES dan SHA-3 Dalam Mengamankan Data Sensitif Pengguna Pada Website Transaksi

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

[1,2,3]Program Studi Informatika, Universitas Sebelas April
Jl. Angkrek Situ No.19, Situ, Sumedang Utara, Kabupaten Sumedang, 45323, Indonesia
e-mail: [1]luthfiassidiq12@gmail.com, [2]fathoni@unsap.ac.id, [3]deris@unsap.ac.id

Artikel Info : Diterima : 30-04-2024 | Direvisi : 06-05-2024 | Disetujui : 28-06-2024

Abstrak - Kehadiran transaksi digital membawa kemudahan dalam melaksanakan kegiatan berniaga dalam banyak bidang. Namun penggunaan internet dalam menciptakan transaksi digital datang dengan beberapa tantangan, salah satunya adalah peretasan atau hacking. Demi mewujudkan transaksi yang aman dan nyaman, tingkat perlindungan pada situs penyedia layanan transaksi digital harus dibuat sekuat mungkin untuk mencegah serangan digital. Maka dibuatlah suatu sistem perlindungan data sensitif pengguna dengan menerapkan metode algoritma AES dan SHA-3. Metode algoritma AES dipilih karena implementasinya yang mudah namun sudah terbukti keamanannya, sedangkan SHA-3 digunakan untuk membuat kunci yang unik dari password pengguna dengan teknologi hashing terbaru. Dengan diterapkannya kedua algoritma ini akan memberikan perlindungan tambahan terhadap keamanan data nominal transaksi dari serangan siber sehingga data tersebut tidak akan pernah tampil di manapun kecuali pengguna melakukan login dengan akun yang sah.

Kata Kunci : Keamanan Website, Advanced Encryption Standard (AES), Secure Hash Algorithm-3 (SHA-3)

Abstracts *- The presence of digital transactions brings convenience in conducting business activities in many fields. However, the use of the internet in creating digital transactions comes with several challenges, one of which is hacking. In order to realize safe and comfortable transactions, the level of protection on digital transaction service provider sites must be made as strong as possible to prevent digital attacks. Therefore, a system for protecting sensitive user data is created by implementing the AES and SHA-3 algorithm methods. The AES algorithm method is chosen because of its easy implementation yet proven security, while SHA-3 is used to create unique keys from user passwords using the latest hashing technology. With the implementation of both algorithms, it will provide additional protection against the security of nominal transaction data from cyber attacks so that the data will never appear anywhere unless the user logs in with a valid account.*

Keywords : Website Security, Advance Encryption Standard (AES), Secure Hash Algorithm-3 (SHA-3)

PENDAHULUAN

Kemudahan dalam mengakses internet telah membawa perubahan secara signifikan terhadap berbagai peraturan hidup dan cara bekerja manusia. Bahkan peranan internet sudah bisa dikatakan sebagai esensi utama dalam bidang teknologi, dan menciptakan dunia baru yang menghasilkan budaya tersendiri dalam masyarakat. Salah satu bidang yang terpengaruh besar dalam budaya masyarakat dengan teknologi internet yaitu bertransaksi. Cara bertransaksi yang memanfaatkan teknologi internet cenderung membawa kemudahan dan 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. Perkembangan ini seakan membuat banyak orang terlena kepada kenyamanan bertransaksi tanpa memikirkan beberapa 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. Dari hal tersebut sudah sepantasnya masyarakat harus selalu berhati-hati saat melalukan transaksi apapun pada dunia digital.

Kegiatan transaksi di internet tidak dapat menolak adanya kejahatan siber yang selalu mengikuti bagaimanapun bentuk kegiatan transaksinya. 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. 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. Saat ini bentuk kejahatan digital terus berkembang dan semakin beragam seiring dengan teknologi yang semakin canggih, maka sejalan dengan hal tersebut tingkat keamanan yang disediakan suatu pelayanan transaksi juga harus menggunakan metode mutakhir dengan teknologi terbaru.

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 sehingga membebaskan pembayaran dari halangan ruang dan waktu. Namun *website* yang menyimpan nominal transaksi ini tidak mempunyai metode pengamanan apapun pada penyimpanan *database* di internet sehingga membuat data tersebut hadir di *website* dalam keadaan rapuh. Maka atas alasan di atas, penulis menerapkan algoritma kriptografi dengan metode enkripsi yang dikombinasikan bersama metode *hashing* untuk mengolah data tersebut supaya tersimpan di *database* dengan aman dan rahasia.

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. Algoritma ini diterapkan pada data-data yang berhubungan dengan nominal pembayaran sehingga data tersimpan dalam bentuk cypertext dan tidak sembarang orang dapat melihat data tersebut. Data yang telah tersimpan dalam bentuk text 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 *hashing* 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 *cypertext* 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 *hashing* merupakan algoritma yang cenderung tidak bisa diubah kembali ke bentuk aslinya. Algoritma kriptografi yang digunakan pada tahap ini adalah SHA-3 atau sering juga disebut Keccak yang di mana merupakan suatu algoritma *hash* terbaru dari keluarga SHA dengan logika *sponge construction* yang membuatnya memiliki metode paling rumit dan sulit dipecahkan.

Terdapat implementasi keamanan yang mirip pada penelitian yang diteliti oleh Dharmawan A. dan Munandar H. yang berjudul "PENERAPAN ALGORITME KRIPTOGRAFI SHA-256 DAN AES-256 UNTUK PENGAMANAN FILE PADA PT PELANGI SENTRAL KREASI". Pada penelitian tersebut membahas tentang penggunaan algoritma AES-256 untuk melakukan enkripsi dan dekripsi file perusahaan, serta penggunaan SHA-256 untuk membuat kunci dekripsi. Aplikasi pada penelitian tersebut adalah aplikasi yang dibuat untuk melakukan enkripsi dan dekripsi file. Aplikasi bekerja dengan cara menginputkan file dan kunci enkripsi, kemudian untuk mendekripsi file tersebut untuk dapat dibuka kembali yaitu dengan cara menggunakan kunci dekripsi yang sama pada tahap awal enkripsi file. Pada penelitian tersebut penulis juga mencatat kecepatan aplikasi dalam melakukan proses enkripsi dan dekripsi file. Perbedaan penelitian tersebut dengan penelitian ini adalah dari penerapan metode dan alur yang digunakan dalam proses enkripsi dan dekripsi data, kemudian jenis algoritma SHA yang digunakan dalam melakukan dekripsi juga berbeda.

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 hashing yang diterapkan dapat mencegah beberapa metode peretasan yang terus berkembang hingga saat ini, seperti contohnya pemalsuan identitas (*spoofing*). Dengan adanya perlindungan awal ini diharapkan penyedia *website* bisa lebih menyadari betapa pentingnya proteksi pada dunia internet dan membuat pengguna dapat lebih memiliki kepercayaan pada pelayanan dari *website* transaksi tersebut.

METODE PENELITIAN

Penelitian tentang mengamankan data sensitif pengguna pada *website* transaksi berlangsung dengan beberapa tahapan yaitu meneliti masalah, merumuskan ide, membuat kode program, implementasi, dan *review*. Tahapan ini dilakukan demi memberikan suatu gambaran target yang ingin dicapai dan mencegah hal-hal tidak diinginkan yang mungkin dapat menghambat penelitian.

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. Kemudian didapatkan keputusan kalau penyimpanan data pada *database* yang akan menjadi prioritas utama dan dikembangkan cara pencegahannya. Penyimpanan data sensitif user 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. Merumuskan Ide

Setelah didapati prioritas data yang perlu diamankan, tahap selanjutnya adalah memilih metode pengamanan yang mudah, ringan, dan dilakukan sepenuhnya oleh sistem. Telah banyak metode pengamanan umum yang tersebar di internet, salah satunya teknik enkripsi dan dekripsi dengan algoritma AES. Algoritma AES dinilai cocok untuk mengamankan data dengan cara merubah *format* data yang tersimpan di *database* menjadi *cypertext*, hal ini membuat data tidak akan semudah itu terlihat baik di sisi *database* maupun *website* bila diakses tanpa menggunakan proses *login* yang sah. 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.

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 sistem tidak terbebani dengan menjalankan berbagai bahasa program. Kode program yang dimodifikasi adalah *database* dengan menambahkan kolom teks enkripsi, sistem enkripsi dan dekripsi dengan metode AES, dan menciptakan kunci dari *password user* dengan SHA-3.

a. 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, *password* akan disimpan ke dalam *variable* 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 1. Sintaks Membuat Kunci SHA dari *Password*

b. Enkripsi Data dengan AES

Sintaks pada gambar 2 digunakan untuk mengubah semua input teks atau *number* menjadi *string* yang terenkripsi. 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 *cypertext*.

```python
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 2. Sintaks Enkripsi Data

c. Penyimpanan Hasil Ekripsi ke *Database*

Cypertext hasil manipulasi sistem enkripsi disimpan ke database dengan tipe Char sehingga data tersebut berbentuk string. Sintaks untuk membuat penyimpanan data adalah seperti gambar 3.

```python
# 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 3. *Database* untuk Menyimpan Data Enkripsi

d. Eksekusi Sistem

Selanjutnya menjalankan proses *function* yang akan melakukan semua proses manipulasi data menjadi berbentuk enkripsi secara *real time*. Semua proses kerja 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.

```python
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 4. Sintaks Menjalankan Enkripsi Data

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

```python
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 5. Menyimpan dan Memperbaharui *Database User*

e. Dekripsi Data dengan AES

Untuk melihat data yang sebenarnya pada tampilan *admin* maupun tampilan pengguna, maka teks enkripsi perlu dilakukan proses dekripsi terlebih dahulu.

```python
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 6. 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 7. Sintaks Menjalankan Dekripsi Data

Pada gambar 7, fungsi dekripsi dijalankan dengan mengambil kunci dan data yang berbentuk enkripsi dari *database*, kemudian bila kunci tersebut berhasil melawati tahap pemeriksaan, maka fungsi ini akan dilanjutkan dan menghasilkan data yang telah terdekripsi, tapi bila kunci enkripsi tidak sama seperti saat proses enkripsi maka akan mengalami *error*.

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, setiap sisi diperhatikan karena proses ini berhubungan dengan data sensitif sehingga kesalahan keamanan sekecil apapun tidak dapat ditolerir. Pada tahap ini peneliti melakukan pembacaan kembali kode program dan membuat sinkronasi antara *backend website* dengan sistem enkripsi dan dekripsi.

5. Pengujian dan Review Sistem

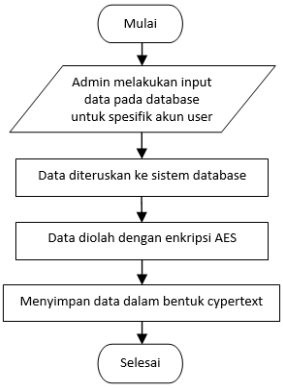
Proses tahapan terakhir yaitu memastikan semua siap digunakan pengguna layanan, dan tidak terdapat proses kesalahan baik dari sisi kinerja *website* maupun logika proses *website* tersebut. *Review* dilakukan dengan cara menjalankan *website* tersebut secara utuh di sisi produksi, artinya *website* akan di*review*, *debug*, dan *testing* oleh 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.

HASIL DAN PEMBAHASAN

1. Proses Implementasi Sistem Keamanan Data

a. *Flowchart* Enkripsi Data

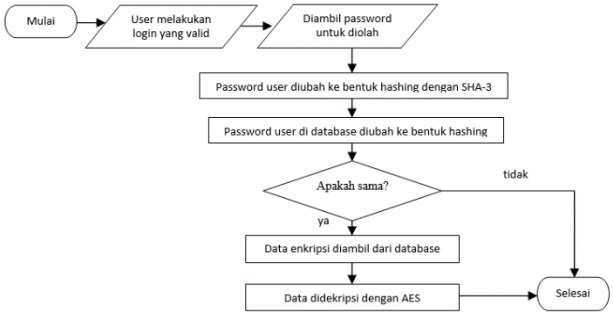
Mengubah data ke dalam bentuk enkripsi dengan AES dimulai dengan *admin* yang bertanggung jawab untuk menginputkan data setiap *user*, kemudian data diolah sistem algoritma AES yang ada di *backend*, dan akhirnya disimpan ke *database*.



Gambar 8. Diagram Alir Enkripsi Data

b. *Flowchart* Dekripsi Data

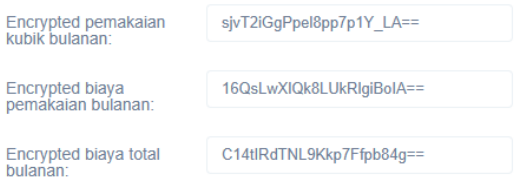
Untuk mengubah kembali data yang ada di *database* ke dalam bentuk sebenarnya, maka dibutuhkan *password* pemilik akun. *Password* akun akan digunakan sebagai kunci dekripsi data AES, namun untuk mengamankan kunci akun dilakukan fungsi *hash* terlebih dahulu pada *password* tersebut dengan algoritma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



Gambar 9. Diagram Alir Dekripsi Data oleh *User*

c. Penyimpanan Data di *Database*

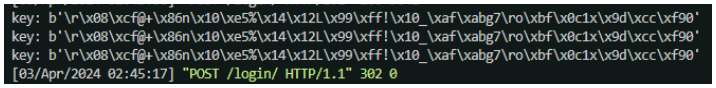
Data hasil enkripsi akan terlihat pada *database* dengan bentuk *cypertext* yang telah berupa enkripsi AES seperti gambar berikut.



Gambar 10. Penyimpanan Data Hasil Enkripsi di *Database*

d. Proses Pemanggilan Data

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



Gambar 11. Proses Pemanggilan Fungsi Dekripsi

Baris-baris data 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* 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.

2. 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 12. 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 13. Penyajian Data Nominal saat Dilakukan *Spoofing*

3. Data Hasil Manipulasi

a. 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==

b. *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.

Tabel 2. Data Hasil Manipulasi SHA-3

USER PASSWORD	SHA3_256
admin	\r\x08\xcf@+\x86n\x10\xe5%\x14\x12L\x99\xff!\x10_\xaf\xabg7\ro\xbf\x0c1x\x9d\xcc\xf90
adminadmin	\xb0\xb1\xae\xc6H\xef\x1e\x12\xc0K\x87\xe8/\n\xe3\xe6~#\x038a^#\x825\x05\xad\x8fi\xf9@?
uQJDCvIUTx	\x16\x01\xacpw%9@\xbd3\xd3\xbc\xcf\xa69X\xfa{\xb8\xa9!\x9d\x9bL\x1d#\x8a\xcfz\x0c.\x1b
uXvkAflGvI	\xf9K\xd7\xbe\x94M\xd4\xd1k)\x1fUf\x052\xc3\x0fB\xc0\r\x11\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

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 digital, maka diterapkan proteksi awal algoritma 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 *cypertext*, 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 memberikan proteksi awal dari beberapa metode kejahatan *hacking* pada dunia digital contohnya seperti pemalsuan identitas atau *spoofing*, maka dilakukan simulasi *hacking spoofing* secara singkat pada penelitian ini untuk menguji keamanan *website*, dan menghasilkan data yang 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 nyaris tidak memiliki kesamaan.

Berasarkan beberapa penelitian, temuan, dan pengalaman dari penulis tentang pengamanan *website* transaksi dengan algoritma AES dan SHA, terdapat beberapa saran mengenai penelitian berikutnya yang dimiliki penulis. 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* seperti *SQL Injection* dan *Cross Site Scripting* untuk lebih memperluas cakupan penggunaan 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. Sintaks pun dapat diubah pula sedemikian rupa untuk meringankan beban kerja *website*.

REFERENSI

Aryanto, M. B., Tahir, M., Devita, S. I., Mustofa, Z. N., Ainiyah, Q., & Sundoro, S. (2023). Implementasi Enkrip Dan Dekrip File Menggunakan Metode Advance Encryption Standard (AES-128). *JUISIK: JURNAL ILMIAH SISTEM INFORMASI DAN ILMU KOMPUTER*, *3*(1), 89–104. https://doi.org/10.55606/juisik.v3i1.434

Azhari, M., Perwitosari, F. J., Mulyana, D. I., & Ali, F. (2022). Implementasi Pengamanan Data pada Dokumen Menggunakan Algoritma Kriptografi Advanced Encryption Standard (AES). *Jurnal Pendidikan Sains Dan Komputer*, *2*(1), 163–171. https://doi.org/10.47709/jpsk.v2i1.1390

Cristy, N., & Riandari, F. (2021). Implementasi Metode Advanced Encryption Standard (AES 128 Bit) Untuk Mengamankan Data Keuangan. *JIKOMSI [Jurnal Ilmu Komputer Dan Sistem Informasi]*, *4*(2), 75–85. https://doi.org/10.9767/jikomsi.v4i2.181

Dharmawan, A., & Munandar, H. (2023). PENERAPAN ALGORITME KRIPTOGRAFI SHA-256 DAN AES-256 UNTUK PENGAMANAN FILE PADA PT PELANGI SENTRAL KREASI. *Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)*, *2*(2), 186–195. https://senafti.budiluhur.ac.id/index.php/senafti/article/view/857

Hidayatulloh, N. W., Tahir, M., Amalia, H., Basyar, N. A., Prianggara, A. F., & Yasin, M. (2023). Mengenal Advance Encrytion Standard (AES) Sebagai Algoritma Kriptografi Dalam Mengamankan Data. *Digital Transformation Technology (Digitech)*, *3*(1), 1–10. https://doi.org/10.47709/digitech.v3i1.2293

Mustika, L. (2020). Implementasi Algoritma AES Untuk Pengamanan Login Dan Data Customer Pada E-Commerce Berbasis Web. *JURIKOM (Jurnal Riset Komputer)*, *7*(1), 148–155. https://doi.org/10.30865/jurikom.v7i1.1943

Pambudi, A., Kusyanti, A., & Data, M. (2019). Perancangan Sistem Pengamanan Data Transaksi Pada Database Terdistribusi Menggunakan Metode Hashing. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *3*(1), 247–252. https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/4085

Parulian, S., Pratiwi, D. A., & Yustina, M. C. (2021). Ancaman dan Solusi Serangan Siber di Indonesia. *Telecommunications, Networks, Electronics, and Computer Technologies (TELNECT)*, *1*(2), 85–92. https://doi.org/10.17509/telnect.v1i2.40866

Praptodiyono, S., Sidiq, M. A., & Muhammad, F. (2021). Implementasi Algoritma SHA-3 Dan AES Sebagai Sistem Keamanan Pada Proses Pensinyalan Mobile IPv6. *Setrum : Sistem Kendali-Tenaga-Elektronika-Telekomunikasi-Komputer*, *10*(2), 59–68. https://doi.org/10.36055/setrum.v10i2.13075

Purike, E., Kurniasih, I. W., Wulandari, F. W., & Nirwani, A. (2022). TRANSAKSI DIGITAL DAN PERKEMBANGAN e-TOURISM DI INDONESIA. *NAWASENA*, *1*(2), 12–19. https://doi.org/10.56910/nawasena.v1i2.157

Putra, Y., Yuhandri, Y., & Sumijan. (2021). Meningkatkan Keamanan Web Menggunakan Algoritma Advanced Encryption Standard (AES) terhadap Seragan Cross Site Scripting. *Jurnal Sistim Informasi Dan Teknologi*, *3*(2), 56–63. https://doi.org/10.37034/jsisfotek.v3i2.44

Rifai, D., Fitri, S., Ramadhan, I. N., & Ramadan, R. (n.d.). *Perkembangan Ekonomi Digital Mengenai Perilaku Pengguna Media Sosial Dalam Melakukan Transaksi*.

Sari, M. P. (2021). Analisis Algoritma SHA-3 Keamanan pada Data Pribadi. *TECHNOSCIENZA*, *5*(2), 231–242. https://doi.org/10.51158/tecnoscienza.v5i2.429

Sianipar, J. S., Nuugroho, N. B., & Mariami, I. (2024). Pengamanan Data Gaji Karyawan Dengan Menggunakan Metode Advanced Encryption Standard (AES). *JURNAL SISTEM INFORMASI TGD*, *3*(1), 35–45. https://doi.org/10.53513/jursi.v3i1.5653

Widyawan, D., & Imelda. (2021). PENGAMANAN FILE MENGGUNAKAN KRIPTOGRAFI DENGAN METODE AES-128 BERBASIS WEB DI KOMITE NASIONAL KESELAMATAN TRANSPORTASI. *SKANIKA*, *4*(1), 15–22. https://doi.org/10.36080/skanika.v4i1.2216