Design and Development a Web Token Library for HTTP Data Encryption Using Exclusive-OR (XOR)
[Rancang Bangun *Library Web Token* Untuk Enkripsi HTTP Data Menggunakan Eksklusif-OR (XOR)]

Bagus Dwi Kurniawan[1)], Mochamad Alfan Rosid[*,2)], Irwan Alnarus Kautsar[*,3)], Nikko Enggaliano Pratama[*,4)]

[1,2,3)] *Program Studi Teknik Informatika, Universitas Muhammadiyah Sidoarjo, Indonesia*
[4)] *Seclab Indonesia*
*Email Penulis Korespondensi: [1)]191080200157@umsida.ac.id, [2)]alfanrosid@umsida.ac.id, [3)]irwan@umsida.ac.id, [4)]nikko@seclab.id

Abstract. *The rise of data hacking in Indonesia is a concerning problem due to the sensitivity of data. Encrypting data is essential to secure data exchange, and the XOR algorithm is a popular option due to its ease of implementation, simplicity, speed, and lightweight. Additionally, using the BLAKE2b hash algorithm provides better security and speed. This research focuses on securing data exchange on HTTP using XOR and BLAKE2b. However, data interception can occur without a digital signature token mechanism. This study proposes the design of a lightweight and easy-to-use web token library to address this issue. The system's suitability was tested using various methods, including encryption time tests for XOR and BLAKE2b algorithms to generate tokens. The results obtained were relatively fast compared to JWT with the HS256 algorithm. The study also tested the API-based authentication process.*

Keywords – *Web Token; XOR; Encryption; BLAKE2b; Library*

Abstrak. *Meningkatnya peretasan data di Indonesia menjadi masalah yang mengkhawatirkan karena data yang sensitif. Untuk mengamankan pertukaran data, enkripsi data sangat penting, dan salah satu pilihan yang populer adalah algoritma XOR karena mudah diimplementasikan, sederhana, cepat, dan ringan. Selain itu, penggunaan algoritma hash BLAKE2b memberikan keamanan dan kecepatan yang lebih baik. Penelitian ini berfokus pada pengamanan pertukaran data pada HTTP dengan menggunakan XOR dan BLAKE2b. Namun, intercept data dapat terjadi tanpa mekanisme token tanda tangan digital. Penelitian ini mengusulkan perancangan library web token yang ringan dan mudah digunakan untuk mengatasi masalah ini. Sistem diuji menggunakan berbagai metode, termasuk pengujian waktu enkripsi untuk algoritma XOR dan BLAKE2b untuk menghasilkan token. Hasil yang diperoleh relatif cepat dibandingkan dengan JWT dengan algoritma HS256. Penelitian ini juga menguji proses otentikasi berbasis API.*

Kata Kunci – *Web Token; XOR; Enkripsi; BLAKE2b; Library*

I. PENDAHULUAN

Peningkatan jumlah peretasan belakangan ini pada sejumlah data baik milik pemerintahan, instansi ataupun milik organisasi khusunya di Indonesia, mengakibatkan banyaknya kebocoran data yang membuat orang khawatir data diri mereka disalahgunakan. Hal ini menjadi situasi yang amat genting jika peretasan terjadi terus-menerus tanpa ada penanggulangan yang baik dan benar. Dengan adanya begitu perlindungan mengenai data menjadi hal yang wajib dilakukan[1].

Data menjadi suatu sasaran utama untuk dicuri bagi para peretas. Data mempunyai deskripsi tersendiri yang berarti sesuatu yang menunjuk pada profil organisasi, orang, kejadian atau peristiwa, kegiatan dan transaksi yang tercatat, terklasifikasi, dan tersimpan namun tidak terorganisasi untuk mendapatkan informasi spesifik[2]. Hal inilah yang menjadikan data menjadi sangat krusial untuk diamankan karena menyangkut informasi sensitif dari orang-orang yang ada pada instansi ataupun organisasi.

Oleh sebab itu, saat melakukan pertukaran data perlu dilakukan pengamanan dengan cara mengenkripsi data. Enkripsi yang berarti suatu proses ketika data atau informasi yang akan dikirim berupa teks biasa (*plaintext*) diubah menjadi teks acak yang tidak dikenali (*chipertext*) sebagai informasi awal dengan mekanisme algoritma tertentu[3]. Dengan cara dienkripsi tersebut sulit untuk ditebak apa informasi awal jika tanpa mengetahui kunci rahasia (*secret key*) terlebih dahulu.

Algoritma yang dapat digunakan dalam pengenkripsian adalah memanfaatkan aritmatika, salah satunya yaitu XOR. XOR atau disebut Eksklusif-OR adalah operasi logika bitwise dengan membandingkan dua bit biner. Apabila salah satu bit biner yang diinput mempunyai nilai 1, maka output tersebut akan menghasilkan angka 1. Namun, jika kedua bit biner yang diinputkan mempunyai nilai 0 atau keduanya mempunyai nilai 1 maka output akan bernilai 0.

XOR berkonsep pada setiap bit *plaintext* dikombinasikan dengan bit kunci secara periodik. Pada tiap 1 bit *plaintext* melakukan XOR dengan setiap bit kunci. XOR dinilai mudah ketika dalam proses pengimplementasian, relatif sederhana, cepat, andal dan murah secara komputasi sehingga dapat menjadi salah satu opsi yang baik jika digunakan dalam proses enkripsi[4]. Teknik XOR dapat ditingkatkan keamanannya dengan cara memperpanjang dan pengacakan kunci. Sehingga intruder tidak mampu menebak kunci asli meskipun dengan komputasional yang memadai.

Dengan pengimplementasian XOR dalam proses enkripsi pada pertukaran HTTP data, keamanan dapat ditingkatkan. HTTP (*Hypertext Transfer Protocol*) mempunyai definisi yang berarti protokol atau set aturan yang mengatur cara informasi disampaikan di antara komputer di internet. Data ini dapat berupa teks, gambar, video, audio, atau bahkan dokumen lainnya. Pertukaran HTTP data memungkinkan browser untuk mengirim permintaan ke server, seperti meminta konten halaman web tertentu. Server kemudian akan mengirimkan respons yang mencakup konten yang diminta dan informasi lainnya yang diperlukan untuk menampilkan halaman web. Konsep dari protokol HTTP adalah file yang berisi rujukan ke file yang lain yang mana memicu permintaan tambahan dalam mentransfer data [5].

Dalam proses pertukaran data, ada permasalahan yang dialami ketika tanpa adanya mekanisme tanda tangan digital untuk menghasilkan web token, yaitu server memerlukan penyimpanan yang lebih besar, tidak mendukung arsitektur aplikasi yang terdistribusi. Web token adalah sebuah objek keamanan digital yang berisi informasi terenkripsi yang digunakan untuk membuktikan identitas atau otoritas suatu entitas pada aplikasi web atau sistem yang terdapat di dalam jaringan. Web token biasanya digunakan sebagai mekanisme autentikasi dan otorisasi pada aplikasi web, yang memungkinkan pengguna untuk mengakses sumber daya atau layanan yang terbatas hanya dengan menggunakan token sebagai bukti identitas atau otoritas. Dengan tidak adanya mekanisme tersebut akan mudah di-*intercept* oleh klien, contohnya menggunakan aplikasi Burp Suite[6]. *Intercept* adalah proses menangkap, memblokir, dan mengubah lalu lintas HTTP. Intercept biasanya digunakan untuk menyaring, memblokir, dan mengontrol konten yang dapat diakses melalui jaringan[7].

Berdasarkan uraian diatas, penelitian ini mengusulkan bahwa perlu adanya sebuah token signature yang digunakan saat proses permintaan pada HTTP data dengan algoritma enkripsi XOR dan fungsi hash BLAKE2b. BLAKE2b sangat cepat dan aman, dan sering digunakan dalam aplikasi keamanan seperti protokol autentikasi dan enkripsi data[8]. BLAKE2b adalah varian dari BLAKE2 yang lebih aman dan efisien baik dari segi kecepatan dan keamanan[9] dan biasanya digunakan dalam aplikasi keamanan seperti penyimpanan password, pembuatan tanda tangan digital, dan validasi integritas data. BLAKE2b dirancang oleh Jean-Philippe Aumasson dan timnya, dan merupakan pengembangan dari algoritma hash awal BLAKE[10]. Hashing digunakan untuk menciptakan token, karena hashing dapat memproduksi nilai yang unik dan tidak dapat dipalsukan untuk setiap input yang berbeda, bahkan jika input tersebut mengalami sedikit perubahan[11]. Fungsi hash sangat penting dalam keamanan komputer karena dapat digunakan untuk memastikan integritas data dan mencegah perubahan data yang tidak sah[12].

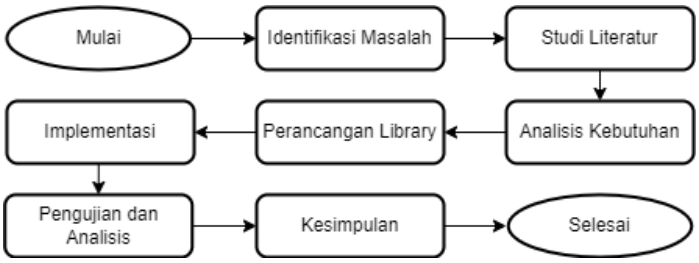
Dalam pembuatan token input yang dihash terdiri dari informasi tertentu, seperti data pengguna dan kunci sebagai pemberi nilai hash yang unik dan stabil untuk setiap data input yang berbeda. Dengan menghash informasi ini, kita dapat menghasilkan nilai hash yang unik dan dapat dipercaya sebagai token tanda tangan digital. Token tersebut digunakan sebagai validasi unik yang aman antara klien dan server. Token ini dapat digunakan untuk mengakses data sensitif atau mengatur akses ke layanan tertentu. Token juga dapat digunakan untuk menyimpan data unik di antara komunikasi klien dan server sehingga meningkatkan keamanan dan mengurangi risiko penyalahgunaan atau akses ilegal. Dalam pembuatan token ini memakai format komunikasi klien dan server dikenal sebagai JSON (*JavaScript Object Notation*). JSON merupakan format bahasa pertukaran data yang ideal karena mudah dibaca, ditulis, dan disusun oleh manusia serta mudah dibuat oleh komputer [13]. Dalam mempermudah pengiriman data JSON melalui protokol yang hanya mendukung data ASCII, seperti HTTP. Base64 juga membantu mencegah hilangnya informasi saat mentransmisikan data JSON yang terdiri dari karakter-karakter khusus, karena beberapa karakter khusus ini dapat mengakibatkan masalah jika dikirimkan dalam format teks biasa[14].

Pada penelitian yang dilakukan oleh Jun-Ya Lee, Wei-Cheng Lin pada tahun 2014[15], Penerapan protokol autentikasi yang diusulkan dalam penelitian tersebut menggunakan mekanisme kriptografi kunci simetris seperti XOR serta teknologi hash untuk melindungi kredensial pengguna dan informasi rahasia pada *Internet od Things*. Dengan penelitian tersebut, penerapan mekanisme XOR dan fungsi hash dapat digunakan sebagai acuan untuk diterapkan dalam pembuatan token tanda tangan digital.

Harapan dengan adanya penelitian ini supaya menjadi alternatif dalam membantu para pengembang dalam mengamankan pertukaran data, dan juga mempermudah pada saat pengembangan aplikasi. Penelitian ini menggunakan bahasa pemrograman PHP dan disajikan dalam bentuk *library lightweight*. *Library lightweight* adalah pustaka perangkat lunak yang dirancang untuk meminimalkan ukuran baik dari segi pemakaian memori dan kompleksitas, sambil tetap menyediakan fungsionalitas dasar yang dibutuhkan. lebih cepat, lebih mudah digunakan, dan lebih mudah diintegrasikan.

II. METODE

Pada bab metode ini memberikan pemaparan tentang alur kerja penelitan dengan penjelasan singkat di dalamnya. Alur metode penelitian diawali dengan identifikasi masalah yang terjadi kemudian dilanjutkan dengan studi literatur, Analsis Kebutuhan, Perancangan Library, Implementasi, Pengujian dan analisis, dan kesimpulan. Alur metode penelitian digambarkan pada gambar 1 dan disertai penjelasan singkat pada tiap segmen alur.



Gambar 1. Alur Metode Pebelitian

A. Identifikasi Masalah

Dengan mengidentifikasi masalah sebagai langkah pertama dalam menjalankan penelitian. Penelitian ini didasarkan pada permasalahan yang telah dijelaskan pada latar belakang yaitu pada saat proses pertukaran data, ada permasalahan yang dialami ketika melakukan tanpa adanya token signature. Hal ini dapat dengan mudah di-*intercept* oleh client, contohnya menggunakan aplikasi burp suite. Dengan permasalahan tersebut maka bagaimana cara menerapkan XOR pada algoritma enkripsi untuk pertukaran HTTP data. Kemudian apakah program ini dapat menjadi alternatif untuk membantu para pengembang (*developer*) dalam melakukan pengamanan pertukaran HTTP data. Setelah melakukan penentuan yang akan diteliti. Kemudian dilanjutkan dengan studi literatur.

B. Studi Literatur

Studi literatur digunakan sebagai upaya dalam memahami temuan dan penelitian yang sudah dilakukan sebagai sumber daya dalam dasar penelitian. Studi literatur diawali dengan pencarian pustaka yang relevan dengan subyek penelitian baik dari jurnal, buku ataupun sumber lainnya. Setelah melakukan pencarian, akan diketahui penelitian apa saja yang telah dilakukan sehingga dapat menghindari dari plagiasi atau duplikasi dengan penelitian terkait.

C. Analisis Kebutuhan

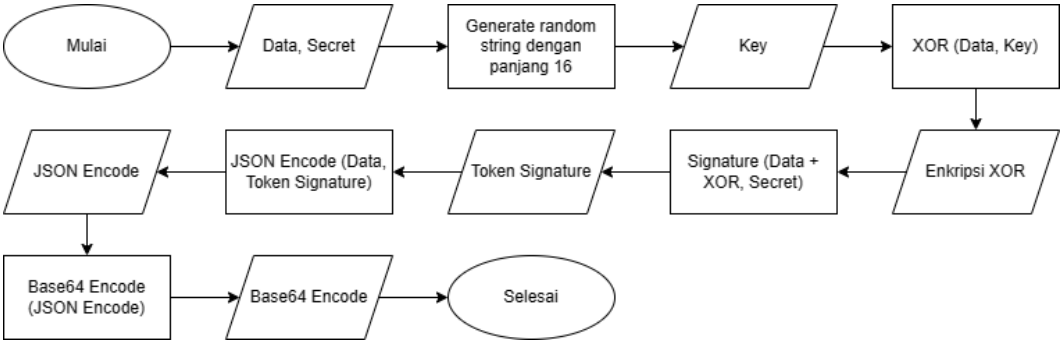
Analisis kebutuhan adalah proses yang digunakan untuk menentukan kebutuhan yang diperlukan untuk mencapai tujuan tertentu. Analisis kebutuhan membantu mengidentifikasi dan mengukur kebutuhan yang harus dipenuhi untuk mencapai tujuan yang telah ditentukan. Proses ini juga dapat membantu dalam mengidentifikasi dan strategi yang paling sesuai. Dalam penelitian ini menganalisis kebutuhan yang akan dibuat baik dari segi kebutuhan nilai variabel input dan jenis output yang dihasilkan. Analisis ini akan mengacu ketentuan tentang algoritma enkripsi. Analisis kebutuhan juga dapat digunakan untuk menentukan kebutuhan lain yang diperlukan untuk memastikan bahwa implementasi algoritma enkripsi XOR berjalan dengan baik.

Dalam analisis kebutuhan ini yaitu menjabarkan tentang kebutuhan fungsional dari sebuah implementasi dari metode pengamanan data dengan menghasilkan token tanda tangan digital menggunakan teknik XOR dan fungsi hash. Dalam analisis kebutuhan program ini, beberapa hal yang dapat diidentifikasi adalah:

1. Kebutuhan untuk membuat tanda tangan digital untuk data dan kunci yang diinputkan.
2. Kebutuhan untuk memastikan bahwa data yang diinputkan tidak dapat diubah tanpa terdeteksi.
3. Kebutuhan untuk memastikan bahwa informasi data yang diterima benar-benar berasal dari sumber yang sah.
4. Kebutuhan untuk mengenkripsi data sehingga orang yang yang tidak berwenang agar tidak dapat membacanya.
5. Kebutuhan untuk memverifikasi bahawa informasi data dan tanda tangan digital yang diterima untuk memastikan keaslian dan integritasnya.

D. Perancangan Library

Pada tahap perancangan library dilakukan setelah analisis kebutuhan sudah terpenuhi. Dalam perancangan library ini dimulai dengan pembuatan diagram alir untuk menjelaskan alur dan proses yang terjadi dalam library tersebut. Perancangan library ini meliputi diagram alir pembuatan token dan proses verifikasi token. proses tahapan pembuatan token tanda tangan digital digambarkan pada gambar 2 di bawah ini.

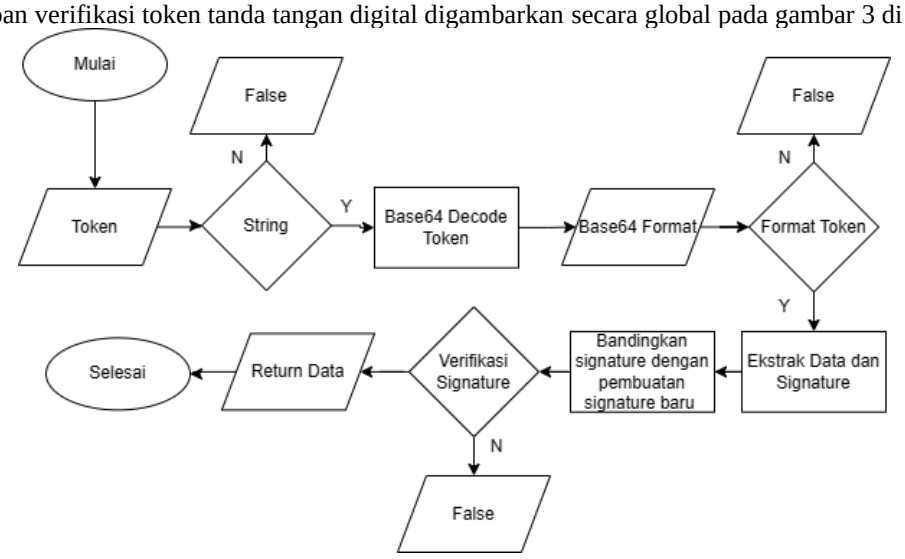


Gambar 2. Diagram Alir Proses Pembuatan Token

Diagram alir di atas menjelaskan proses pembuatan tanda tangan digital dari program di atas dengan tahapan-tahapan sebagai berikut:

1. Input data dan secret: Tahap pertama adalah memasukkan input data dan secret sebagai parameter untuk membuat tanda tangan digital.
2. Membuat key secara acak: Selanjutnya, program akan membuat key secara acak dengan panjang 16. Key ini berformat string hexadecimal. Key ini akan digunakan untuk melakukan operasi XOR pada data.
3. Melakukakn operasi XOR data dengan Key: Data yang dimasukkan akan di-XOR-kan dengan key yang dibuat pada tahap sebelumnya. Operasi XOR ini dilakukan pada setiap byte data dan key secara bergantian, sehingga hasilnya akan selalu sama panjang dengan data asli.
4. Menggabungkan data dan hasil XOR: Hasil XOR dari tahap sebelumnya akan digabungkan dengan data asli.
5. Signature Secret Key: Data yang sudah dihasilkan pada tahap sebelumnya akan di-hash dengan menggunakan fungsi yang disediakan oleh PHP yaitu sodium_crypto_generichash() yang menggukan algoritma BLAKE2b dengan menggunakan secret key yang dimasukkan pada tahap pertama. Hasil hash inilah yang akan menjadi signature.
6. Encode Data dan Signature dalam Base64 Format: Setelah signature berhasil dihasilkan, data dan signature akan di-encode dalam format Base64 agar dapat disimpan dengan aman.

Proses tahapan verifikasi token tanda tangan digital digambarkan secara global pada gambar 3 di bawah ini:



Gambar 3. Diagram Alir Proses Verifikasi Token

Berikut adalah penjelasan mengenai alur diagram alir dari proses verifikasi token pada program di atas:

1. Program menerima input berupa token yang akan diverifikasi.
2. Program melakukan validasi terhadap format token. Jika format tidak valid, program mengembalikan nilai false.
3. Program melakukan decode terhadap token menggunakan base64_decode dan json_decode. Jika gagal, program mengembalikan nilai false.
4. Program mengekstrak data dan signature dari token yang sudah didecode.

5. Program melakukan verifikasi terhadap signature dengan menggunakan algoritma yang sama dengan saat pembuatan signature. Jika signature tidak valid, program mengembalikan nilai false.
6. Program melakukan pembuatan signature baru dengan menggunakan data yang sudah diekstrak dari token, kunci rahasia yang sama dengan saat pembuatan signature, dan key yang dihasilkan pada saat pembuatan signature.
7. Program melakukan validasi terhadap data yang sudah diekstrak dari token dan data yang dihasilkan dari pembuatan signature baru. Jika kedua data tidak sama, program mengembalikan nilai false.
8. Jika semua validasi telah dilalui, program mengembalikan data yang sudah diekstrak dari token.

E. Implementasi

Pada tahap implementasi merupakan tahap lanjutan dari perancangan library yang sudah tersusun. Tahap implementasi adalah tahap mengeksekusi rencana untuk mencapai tujuan-tujuan yang telah ditetapkan. Proses implementasi dilakukan untuk menerapkan rancangan sistem yang telah dirancang agar dapat berubah menjadi sesuatu yang dapat menyelesaikan masalah dan memenuhi kebutuhan yang ada di latar belakang. Proses implementasi ini penulisan disajikan dengan *pseudocode* dengan tujuan memudahkan dalam memahami alur pembuatan program yang menerapkan XOR dan fungsi hashing.

F. Pengujian dan Analisis

Pada tahap ini library telah diimplementasikan. Pengujian dilakukan untuk mengkonfirmasi apakah library berfungsi sesuai dengan keinginan dan spesifikasi yang ditentukan sebelumnya. Strategi pengujian ini memperhatikan apakah output sesuai dengan harapan dan melakukan penanganan eror bila ada kesalahan saat menjalankan dengan cara melakukan pengujian menggunakan pengujian performa, notasi big O dan rest API.

G. Penarikan Kesimpulan dan Saran

Penarikan kesimpulan dan saran adalah tahap akhir pada penelian yang dilakukan. Hasil implementasi dan pengujian terhadap library yang telah dibangun akan membantu dalam menarik kesimpulan. Selain itu, saran juga digunakan untuk meningkatkan kualitas hasil penelitian dan untuk memperbaiki masalah yang terjadi, termasuk untuk kemajuan penelitian yang lebih lanjut.

III. HASIL DAN PEMBAHASAN

A. Hasil dan Pembahasan

Implementasi

Pada tahap implementasi meliputi implementasi algoritma utama yaitu algoritma XOR dan hashing. Penulisan algoritma pada implementasi disajikan dalam bentuk *pseudocode* pada tabel 1 di bawah ini.

Tabel 1. Algoritme pseudocode XOR

```
Algorithm 1: XOR
1     function xor():
2       dataLength = panjang(data)
3       keyLength = panjang(key)
4       xorResult = “”
5       for i = 0 to dataLength - 1:
6         dataChar = ambil karakter ke-i dari data
7         keyChar = ambil karakter ke-(i mod keyLength) dari key
8         xorChar = operasi XOR antara nilai dataChar dan keyChar
9         xorResult = gabungkan karakter xorChar ke xorResult
10      end
11      kembalikan nilai xorResult
12    end
```

Berikut penjelasan dari *pseudocode* untuk implementasi XOR di atas:

1. Baris 2 – 4: Mengambil panjang data dan kunci yang akan digunakan dari metode pengacakan di awal. Setelah itu, variabel xorResult diinisialisasi sebagai string kosong untuk menyimpan hasil operasi XOR nanti.
2. Baris 5 – 8: Program akan melakukan operasi XOR pada setiap bit pada data dan kunci dengan melakukan loop dari 0 hingga panjang data dikurangi satu. Pada setiap iterasi loop, program mengambil byte ke-i dari data dan kunci, kemudian melakukan operasi XOR pada byte tersebut.

3. Baris 9 – 11: Hasil dari operasi XOR kemudian dikonversi ke karakter dan ditambahkan ke variabel xorResult sebagai hasil akhir. Setelah loop selesai, program mengembalikan hasil operasi XOR sebagai string.

Setelah dilakukan pengimplementasian XOR, kemudian dilanjutkan dengan operasi hashing dengan yang disajikan dalam bentuk *pseudocode* tabel 2 di bawah ini.

Tabel 2. Algoritm Psudocode hashing

Algoritm 2: Hash	
1	Function sign():
2	signedData = this->data + this->xor()
3	signature = sodium_crypto_generichash(signedData, this->secret)
4	hexSignature = bin2hex(signature)
5	this->signature = hexSignature
6	End Function

Berikut penjelasan dari tabel *pseudocode* dari implementasi hash:

1. Baris 2 – 3: Program menggabungkan data asli dengan hasil operasi XOR yang dihasilkan oleh fungsi xor(). Setelah itu, program membuat tanda tangan digital menggunakan fungsi hash *sodium_crypto_generichash()* dengan memasukkan signedData dan secret sebagai parameter.
2. Baris 4 – 5: Setelah tanda tangan digital berhasil dibuat, program akan mengonversi tanda tangan digital tersebut ke dalam format heksadesimal menggunakan fungsi *bin2hex()*. Fungsi *bin2hex()* digunakan untuk mengubah format biner dari hasil hah ke bentuk string *hexadecimal*. Kemudian, program akan menyimpan tanda tangan digital yang telah diubah ke dalam format heksadesimal tersebut pada variabel signature pada objek ini.

B. Pengujian Library

Pengujian library dilakukan demi menghasilkan keluaran yang diharapkan. Pada tabel 3 dibawah ini adalah simulasi dari pengujiannya.

Tabel 3. Script run.php

Algoritme 3: script run.php	
1	<?php
2	require_once 'MySign.php';
3	$data = array(
4	"nama" => 'bagus',
5	"email" => 'bagus@gmail.com',
6	"role" => 'user');
7	$secret = 'Ini adalah kunci rahasia';
8	$Sign = new MySign();
9	$Sign->initiate($secret);
10	$encodedData = $Sign->safeData($data);
11	echo "Encoded data: " . $encodedData . PHP_EOL;
12	$isVerified = $Sign->verifyData($encodedData);
13	echo "Data: " . var_export($isVerified, true);

Berikut penjelasan dari script run.php di atas:

1. Baris 2: Memanggil file dengan nama 'MySign.php'.
2. Baris 3 – 6: Program mendefinisikan sebuah array dengan variabel $data yang berisi informasi seperti nama, email, dan role. Kemudian,
3. Baris 7: Program menentukan sebuah variabel $secret yang berisi kunci rahasia yang akan digunakan untuk mengamankan data.
4. Baris 8 – 9: Program membuat sebuah instance dari kelas MySign dengan memanggil fungsi initiate() dengan argumen $secret. Fungsi ini akan menginisialisasi objek MySign dengan kunci rahasia yang telah ditentukan.
5. Baris 10 – 11: Program menggunakan objek MySign untuk mengamankan data pada array $data dengan memanggil fungsi safeData(). Fungsi ini akan mengembalikan data yang telah dienkripsi menggunakan kunci rahasia yang telah ditentukan.
6. Baris 12 – 13: Setelah data dienkripsi, program mencetak hasil encoding dengan menggunakan perintah echo dan mengembalikan nilai boolean yang menunjukkan apakah data berhasil diverifikasi atau tidak dengan memanggil fungsi verifyData() pada objek MySign.

Gambar 4 di bawah ini adalah hasil keluaran dari pengujian library yang telah dilakukan.

```
$ php run.php
Encoded data: eyJkYXRhIjoielwibmFtYVwiOlwiYmFndXNcIixcImVtYWlsXCI6XCJiYWdlc0BnbWFpbC5jb21c
IixcInJvbGVcIjpcInVzZXJcIn0iLCJzaWduYXR1cmUiOiIwMzc3M2U1NDY1MDAwZjgwNjV1Y2I4ZTNiYWIxYTQxYm
I1OTdmMWExMGFjOGExNDRmMGNkNDN1YjgwNGNkN2U1In0=
Data: '{"nama":"bagus","email":"bagus@gmail.com","role":"user"}'
```

Gambar 4. Hasil Pengujian Sistem

C. Pengujian Performa Kecepatan

Performa suatu algoritma sangat penting untuk menjamin kualitas dan efisiensi dari suatu program. Dalam kasus pengujian kali ini, dilakukan perbandingan antara waktu eksekusi enkripsi menggunakan JWT (Json Web Token) dengan algoritma hashing HS256 dengan inputan array 1-15 field. Dua pengujian dilakukan dengan perbedaan panjang data pada masing-masing field. Pengujian pertama menggunakan panjang data 1000 pada setiap field, sementara pengujian kedua menggunakan panjang data antara 50-100 pada setiap field. Hasil dari pengujian ini akan memberikan informasi mengenai performa algoritma yang digunakan dan dapat digunakan sebagai dasar untuk memperbaiki kualitas program. Penting untuk memastikan algoritma yang digunakan mampu menangani berbagai ukuran data dengan cepat dan efisien.

Pada tabel 4 di bawah ini adalah hasil dari pengujian pertama yaitu memberikan inputan array 1 – 15 field dengan masing-masing panjang data 1000.

Tabel 4. Pengujian Kecepatan 1

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
XOR	0.0024	0.0049	0.00779	0.01151	0.01732	0.02440	0.03196	0.05537	0.06822	0.08102	0.11131	0.11802	0.14634	0.16840	0.19138
JWT	0.0043	0.0061	0.00903	0.01177	0.01483	0.01963	0.02476	0.04214	0.05329	0.07727	0.12204	0.11229	0.12382	0.15652	0.16656

Berdasarkan hasil pengujian pada tabel 4 di atas, terlihat bahwa waktu eksekusi untuk operasi XOR dan hashing BLAKE2b lebih cepat dibandingkan dengan JWT pada 1 – 4 field dengan panjang data masing-masing field 1000 dan pada 5 – 15 field dengan panjang data masing-masing field 1000 mengalami penurunan kecepatan yang dikarenakan algoritma XOR dan fungsi hash BLAKE2b algoritma yang lebih kuat cenderung memerlukan waktu yang lebih lama untuk dieksekusi, maka hal ini dapat mempengaruhi kecepatan. Hal tersebut menunjukkan bahwa algoritma XOR dan hashing BLAKE2b lebih efisien dalam mengeksekusi perintah enkripsi dengan field pendek dibandingkan dengan JWT dalam kasus pengujian ini. Namun, jika diperlukan performa yang lebih tinggi dan efisiensi dalam mengeksekusi enkripsi, algoritma XOR dan hashing BLAKE2b dapat menjadi alternatif yang lebih baik.

Pada tabel 5 di bawah ini adalah hasil dari pengujian kedua yaitu memberikan inputan 1 – 15 field dengan masing memiliki panjang data 50 – 100.

Tabel 5. Pengujian Kecepatan 2

	50	60	70	80	90	100
XOR	0.00253	0.00258	0.00284	0.00318	0.00309	0.00324
JWT	0.00420	0.00438	0.00448	0.00416	0.00488	0.00493

Dapat disimpulkan dari tabel 5 yang diberikan bahwa waktu eksekusi operasi XOR dan BLAKE2b lebih cepat daripada JWT dengan panjang data sampel yang berbeda. Untuk panjang data antara 50 hingga 100, waktu eksekusi XOR dan BLAKE2b berkisar antara 0,00253 hingga 0,00324 detik, sedangkan waktu eksekusi JWT berkisar antara 0,00420 hingga 0,00493 detik. Hal ini menunjukkan bahwa XOR dan BLAKE2b lebih efisien dibandingkan JWT dalam mengengkripsi data. Pengujian dengan panjang data yang lebih kecil bertujuan untuk menguji kecepatan operasi enkripsi dan dekripsi pada masing-masing metode dengan jumlah field yang berbeda pada setiap panjang data. Hal ini dilakukan untuk memperoleh informasi tentang kinerja masing-masing metode pada skala yang lebih kecil, sehingga dapat memberikan gambaran yang lebih jelas tentang kecepatan operasi pada setiap metode enkripsi data. Dengan demikian, pengujian pada panjang data yang lebih kecil dapat membantu dalam menentukan metode enkripsi data yang tepat dan efisien untuk digunakan pada skala yang lebih besar. Selain itu, pengujian pada panjang data yang lebih kecil juga dapat membantu dalam mengevaluasi dan membandingkan kinerja masing-masing metode dengan lebih mudah dan cepat.

Kesimpulan dari kedua pengujian yang dilakukan memperlihatkan bahwa operasi XOR dan BLAKE2b lebih efisien dalam kondisi field dan panjang data yang lebih kecil. Ini dikarenakan algoritma hash BLAKE2b menghasilkan hasil hash dengan ukuran yang lebih kecil.

D. Pengujian Berbasis API

Dalam pengujian ini, program diuji dengan melakukan permintaan API menggunakan metode POST dan GET pada data dengan panjang data yang berbeda. Pengujian ini bertujuan untuk memastikan bahwa data yang dikirim dan diterima melalui API dapat dilakukan dengan baik menggunakan library yang telah dirancang. Selain itu, tujuan dari pengujian ini adalah untuk mengevaluasi kemampuan program dalam menangani permintaan pada data dengan berbagai panjang yang berbeda. Hasil pengujian dapat dilihat pada tabel 6 di bawah ini.

Tabel 6. Hasil Pengujian Berbasis API

No.	Panjang Data	Jenis Request	Hasil Pengujian	Keterangan
1	10	POST	POST Data integrity is valid	Data berhasil dikirim dan integritasnya terjaga
		GET	GET Data integrity is valid	Data berhasil diterima dan integritasnya terjaga
2	100	POST	POST Data integrity is valid	Data berhasil dikirim dan integritasnya terjaga
		GET	GET Data integrity is valid	Data berhasil diterima dan integritasnya terjaga
3	1000	POST	POST Data integrity is valid	Data berhasil dikirim dan integritasnya terjaga
		GET	GET Data integrity is valid	Data berhasil diterima dan integritasnya terjaga
4	10000	POST	POST Data integrity is valid	Data berhasil dikirim dan integritasnya terjaga
		GET	GET Data integrity is valid	Data berhasil diterima dan integritasnya terjaga
5	100000	POST	POST Data integrity is valid	Data berhasil dikirim dan integritasnya terjaga
		GET	GET Data integrity is valid	Data berhasil diterima dan integritasnya terjaga

Pada tabel 6 dapat disimpulkan bahwa pengiriman dan penerimaan data melalui metode POST dan GET telah berhasil dilakukan dengan integritas data yang terjaga. Hasil pengujian menunjukkan bahwa dapat mengirim dan menerima data dengan benar pada berbagai ukuran data yang berbeda, mulai dari 10 hingga 100.000. Selain itu, hasil pengujian juga menunjukkan bahwa tidak ada kesalahan dalam pengiriman dan penerimaan data, sehingga integritas data terjaga dengan baik. Hal ini menunjukkan bahwa sistem yang digunakan untuk pengiriman dan penerimaan data telah berfungsi dengan baik.

IV. SIMPULAN

Pada hasil dan pembahasan dapat disimpulkan bahwa perancangan library dengan mengimplementasikan algoritma XOR dan hashing dapat mengengkripsi data yang menghasilkan token untuk proses autentikasi dan juga memverifikasi integritas data dari token yang telah dihasilkan. Kemudian pada sisi pengujian performa kecepatan dapat diperoleh waktu rata-rata lebih cepat daripada JWT dengan algoritma HS256 pada ukuran data yang relatif lebih kecil. Sehingga dapat dikatakan bahwa library ini mencukupi proses waktu yang efisen dan mengamankan pertukaran data dengan baik. Meskipun demikian, perlu diingat bahwa keamanan data adalah hal yang terus berubah seiring perkembangan teknologi. Oleh karena itu, untuk terus memperbarui dan meningkatkan keamanan dengan melakukan peninjauan keamanan secara berkala.

UCAPAN TERIMA KASIH

Puji syukur terhadap Allah Subhanahu Wataala yang telah memberikan rahmat dan ridho-Nya sehingga penelitian ini dapat diselesaikan. Penulis mengucapkan terima kasih banyak kepada Nikko Enggaliano Pratama S.Kom dari Secretlab Indonesia selaku ahli dalam bidang *secure coding* dan keamanan data informasi sebagai validator terhadap penelitian ini.

REFERENSI

[1] M. Betty Yel and M. K. M Nasution, "Keamanan Informasi Data Pribadi Pada Media Sosial," *J. Inform. Kaputama*, vol. 6, no. 1, pp. 92–101, 2022, [Online]. Available: http://jurnal.kaputama.ac.id/index.php/JIK/article/view/768.

[2] F. P. Nugroho, R. W. Abdullah, S. Wulandari, and Hanafi, "Keamanan Big Data di Era Digital di Indonesia," *J. Inf.*, vol. 5, no. 1, pp. 28–34, 2019.

[3] R. Rosdiana, "Sekuritas Sistem Dengan Kriptografi," *Al-Khwarizmi J. Pendidik. Mat. dan Ilmu Pengetah. Alam*, vol. 3, no. 1, 2018, doi: 10.24256/jpmipa.v3i1.216.

[4] Suparyanto dan Rosad, "IMPLEMENTASI ALGORITMA AES DAN ALGORITMA XOR PADA

APLIKASI ENKRIPSI DAN DEKRIPSI TEKS BERBASIS ANDROID," *Suparyanto dan Rosad*, vol. 5, no. 3, pp. 248–253, 2020.
[5] A. R. Pratama, M. H. H. Ichsan, and A. Kusyanti, "Implementasi Algoritme AES Pada Pengiriman Data Sensor DHT11 Menggunakan Protokol Komunikasi HTTP," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 3, no. 4, pp. 3781–3789, 2019.
[6] C. Mainka, V. Mladenov, T. Guenther, and J. Schwenk, "Automatic recognition, processing and attacking of single sign-on protocols with burp suite," *Lect. Notes Informatics (LNI), Proc. - Ser. Gesellschaft fur Inform.*, vol. 251, pp. 117–131, 2015.
[7] V. M. Deshpande, M. K. Nair, and D. Shah, "Major Web Application Threats for Data Privacy & Security-Detection, Analysis and Mitigation Strategies," *Accepted*, vol. 7, no. 10, pp. 182–198, 2017, [Online]. Available: www.ijsrst.com.
[8] N. F. Sitorus, A. Kusyanti, and A. Bhawiyuga, "Implementasi Autentikasi Berbasis Token Menggunakan Platform Agnostic Security Tokens (PASETO) Sebagai Mekanisme Autentikansi RESTful API," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 4, no. 11, pp. 3947–3955, 2020, [Online]. Available: http://j-ptiik.ub.ac.id.
[9] V. Ganesh and B. V. H. Sandilya, "Implementation of SIMD Instruction Set Extension for BLAKE2," *2019 10th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2019*, 2019, doi: 10.1109/ICCCNT45670.2019.8944835.
[10] J. Aumasson, S. Neves, Z. W. Hearn, and C. Winnerlein, "BLAKE2 : Simpler , Smaller , Fast as MD5," pp. 119–135, 2013.
[11] B. Maryanto, "Penggunaan Fungsi Hash Satu-Arah Untuk Enkripsi Data," *Media Inform.*, vol. 7, no. 3, pp. 138–146, 2008.
[12] M. R. Anwar, D. Apriani, and I. R. Adianita, "Hash Algorithm In Verification Of Certificate Data Integrity And Security," *Aptisi Trans. Technopreneursh.*, vol. 3, no. 2, pp. 65–72, 2021, doi: 10.34306/att.v3i2.212.
[13] J. Friesen, "Introducing JSON," in *Java XML and JSON*, 2019.
[14] F. W. C, A. P. Rahagiar, and F. Fretes, "Penerapan Algoritma Gabungan Rc4 Dan Base64 Pada Sistem Keamanan E-Commerce," *Semin. Nas. Apl. Teknol. Inf.*, vol. 2012, no. Snati, pp. 47–52, 2012.
[15] J. Y. Lee, W. C. Lin, and Y. H. Huang, "A lightweight authentication protocol for Internet of Things," *2014 Int. Symp. Next-Generation Electron. ISNE 2014*, pp. 1–2, 2014, doi: 10.1109/ISNE.2014.6839375.

Conflict of Interest Statement:
The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.