ISSN: 2746-4237

Sistem Keamanan Otentikasi Pengguna pada Modul Single Sign On Menggunakan OAuth 2.0 dan One Time Password

Ilham Gumeraruloh Arianto[1)], Wina Witanti[2)], Herdi Ashaury[3)]

[1)]Teknik Informatika, FMIPA, Universitas Jenderal Achmad Yani
[2,3)] Universitas Jenderal Achmad Yani
[1)] ilhamgariyanto6@gmail.com ✉

ABSTRACT

Information security is a top priority in protecting sensitive data within systems that handle data transfers. This study developed a Technical Support Assistance (TSA) system with enhanced security by combining a Single Sign-On (SSO) module based on Open Authentication (OAuth 2.0) and a time-based One-Time Password (OTP) method. This approach establishes an effective two-factor authentication (2FA) mechanism to counter risks such as sniffing, brute force attacks, and man-in-the-middle (MITM) attacks. Testing results showed that without OAuth 2,0 and OTP, the success rates of attacks were 63% for brute force, 50% for sniffing, and 65% for MITM. After implementing OAuth 2.0 and OTP, these rates significantly dropped to 25%, 5%, and 10%, respectively, demonstrating the effectiveness of combining OAuth 2.0 and OTP in enhancing system protection. Compared to previous authentication methods, TSA offers advantages such as dynamic token-based security, a significant reduction in attack risks, easier integration with various services, and greater authentication efficiency. This study provides an innovative solution for improving sensitive data security and is relevant for organizations requiring a high level of protection within their systems.

Keywords: Authentication, TSA, OAuth 2.0, Single Sign-On, OTP

ABSTRAK

Keamanan informasi menjadi prioritas utama dalam melindungi data sensitif pada sistem yang menangani transfer data. Penelitian ini mengembangkan sistem Technical Support Assistance (TSA) dengan keamanan yang ditingkatkan melalui kombinasi modul Single Sign-On (SSO) berbasis Open Authentication (OAuth 2.0) dan metode One-Time Password (OTP) berbasis waktu. Pendekatan ini menciptakan autentikasi dua faktor (2FA) yang efektif dalam menghadapi risiko serangan seperti *sniffing*, *brute force attacks*, dan *man-in-the-middle* (MITM). Hasil pengujian menunjukkan bahwa tanpa OTP, tingkat keberhasilan serangan adalah 63% untuk *brute force*, 50% untuk *sniffing*, dan 65% untuk MITM. Setelah penerapan Oauth 2.0 dan OTP, angka ini turun signifikan menjadi masing-masing 25%, 5%, dan 10%, membuktikan bahwa kombinasi OAuth 2.0 dan OTP meningkatkan perlindungan sistem secara signifikan. Dibandingkan metode autentikasi terdahulu, TSA menawarkan keunggulan berupa keamanan berbasis token dinamis, pengurangan risiko serangan secara drastis, integrasi yang lebih mudah dengan layanan lain, serta efisiensi autentikasi yang lebih tinggi. Penelitian ini memberikan solusi inovatif untuk meningkatkan keamanan data sensitif dan relevan bagi organisasi yang memerlukan perlindungan tingkat tinggi dalam sistem mereka.

Kata kunci: Otentikasi, TSA, OAuth 2.0, Single Sign-On, OTP.

I. PENDAHULUAN

Teknologi modern telah memberikan dampak signifikan pada keamanan informasi, tetapi juga menimbulkan tantangan baru, salah satunya adalah meningkatnya ancaman terhadap sistem autentikasi berbasis password. Password rentan terhadap serangan seperti *sniffing* dan *brute force*, yang memungkinkan peretas untuk mencuri data pengguna dengan mudah [1][2]. *Sniffing*, yang sering digunakan oleh peretas untuk mencuri *username* dan *password* adalah teknik menggunakan perangkat lunak untuk mengambil informasi otentikasi [3].

Dalam konteks keamanan sistem TSA yang mengelola data sensitif, penting untuk memiliki tingkat

Diterima Redaksi: 10-01-2025 | Selesai Revisi: 17-02-2025 | Diterbitkan Online: 27-02-2025

keamanan yang tinggi [4]. Salah satu solusi yang dapat diimplementasikan adalah *Single Sign-On* (SSO) [5]. Meskipun SSO meningkatkan kenyamanan pengguna dengan memungkinkan akses ke berbagai layanan menggunakan satu akun, kelemahan utama adalah resiko besar jika kredensial pengguna berhasil diretas yang dapat membuat seluruh sistem rentan [6].

OAuth 2.0 menawarkan solusi keamanan yang lebih baik dengan token akses dinamis [8], namun ancaman seperti *sniffing*, *brute force*, dan MITM tetap menjadi tantangan serius. Oleh karena itu, penggunaan *One Time Password* (OTP) berbasis waktu yang dikirimkan melalui email dapat memberikan lapisan keamanan tambahan untuk mencegah serangan seperti *sniffing* dan *brute force*.

Penelitian sebelumnya menunjukkan bahwa OTP yang dipadukan dengan algoritma SHA-512 menawarkan proses pengiriman kode yang cepat, namun penggunaan satu algoritma untuk menghasilkan OTP dapat menjadi titik lemah jika algoritma tersebut berhasil diretas [9][10]. Penelitian lain menunjukkan bahwa meskipun SSO memudahkan pengguna [11], risiko tinggi tetap ada jika kredensial diretas memungkinkan akses tidak sah ke berbagai layanan hanya dengan satu kredensial [12].

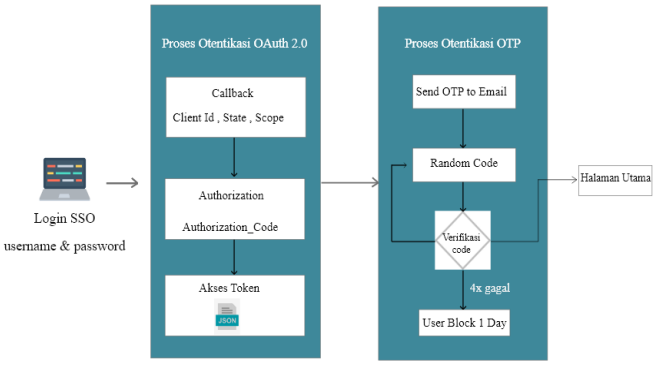
Meskipun OAuth 1.0 digunakan sebagai standar otorisasi delegasi untuk IoT [13], protokol ini memerlukan perangkat dengan spesifikasi tinggi dan tidak selalu dapat digunakan dalam jaringan yang lebih luas [15]. Sebaliknya, OAuth 2.0 menawarkan tiga lapisan protokol keamanan yang lebih baik [16] dan lebih sesuai untuk konfigurasi token akses dalam skenario besar seperti IoT [17]. Selain itu, Oauth 2.0 meningkatkan keamanan melalui enkripsi dalam distribusi akses token [18].

Penggunaan *Proof Key for Code Exchange* (PKCE) dalam OAuth 2.0 mengurangi risiko kebocoran data dengan memastikan bahwa entitas yang meminta token adalah pihak yang sama dengan yang melakukan autentikasi, sehingga meningkatkan keamanan secara signifikan [19]. Pengujian terhadap OAuth 2.0 menunjukkan bahwa sistem ini lebih unggul dalam aspek keamanan dibandingkan metode otorisasi standar seperti Basic Auth API atau API Key [20].

Meskipun OAuth memberikan autentikasi yang baik melalui SSO, sistem ini tetap rentan terhadap serangan jika tidak dilengkapi dengan mekanisme keamanan tambahan. Berdasarkan analisis, TSA memerlukan solusi keamanan yang lebih komprehensif. Kombinasi SSO berbasis OAuth 2.0 yang diperkuat dengan OTP berbasis waktu dapat memberikan dua faktor autentikasi (2FA) untuk melindungi data sensitif dan mitigasi risiko serangan siber, sekaligus meningkatkan kenyamanan pengguna [7].

II. METODE PENELITIAN

Penelitian ini terdapat beberapa tahapan untuk mencapai tujuan sistem untuk mengamankan sistem TSA, tahapan-tahapan tersebut di antaranya adalah perolehan data, implementasi Oauth (2.0), proses *one time password* (OTP), serta implementasi metode Oauth 2.0 dan *single sign on* (SSO).



Gambar 1. Metodologi Penelitian

Pada gambar 1 menunjukkan alur sistem login yang diterapkan pada modul Single Sign-On (SSO). Pengguna memasukkan username dan password, lalu sistem memproses login menggunakan OAuth. Sistem mengambil data seperti *client* ID, *state*, dan *scope* untuk dilakukan otorisasi. Setelah otorisasi, data dikonversi ke format JSON yang berisi *client* ID, *state*, *scope*, *access* token, *refresh* token, dan *authorization code*. Sistem kemudian memeriksa apakah *access* token masih aktif atau perlu diperbarui dengan *refresh* token. Jika valid, pengguna diarahkan ke tahap autentikasi OTP, di mana kode OTP dikirim secara acak dalam bentuk angka dan karakter.

A. Perolehan Data

Perolehan data melibatkan pengumpulan informasi tentang sistem TSA dan pengguna untuk implementasi modul Single Sign-On (SSO) dengan OAuth 2.0 dan *One Time Password* (OTP), bertujuan meningkatkan autentikasi pengguna pada layanan TSA.

1) Langkah Pertama

   Studi literatur mengkaji konsep SSO, OAuth 2.0, dan OTP. SSO memudahkan akses dengan satu login, OAuth 2.0 memungkinkan otorisasi tanpa berbagi kredensial, dan OTP menambah lapisan keamanan. Namun, ada tantangan seperti kebocoran token di OAuth 2.0, kerentanan sniffing dan *man in the middle* di SSO, serta kelemahan OTP jika salurannya tidak aman.

2) Langkah Kedua

   Analisis dokumentasi sistem TSA menunjukkan penggunaan HTTPS untuk enkripsi, namun belum ada implementasi OAuth 2.0 atau SSO, serta tidak ada pembatasan login berulang yang berisiko serangan *brute force*.

3) Langkah Ketiga

   Pengumpulan data lapangan melalui wawancara dan analisis jaringan menggunakan *Wireshark* dan *Burp Suite* mengungkapkan risiko *sniffing* dan kelemahan sistem terhadap serangan pada jaringan publik, dengan tingkat keberhasilan 30%. Simulasi *brute force* menunjukkan 15% keberhasilan login paksa dari 1000 percobaan.

4) Langkah Keempat

Analisis menunjukkan kelemahan pada autentikasi dan perlindungan *brute force*. Disarankan untuk mengintegrasikan OAuth 2.0 atau SSO, menambahkan OTP, serta menerapkan pembatasan login dan *rate-limiting* untuk memperkuat keamanan sistem TSA.

B. Proses Open Authorization (Oauth 2.0)

Pada tahapan ini pembuatan alur sistem yang akan dibuat sebagai pengamanan data pada E-TSA dimana ada 4 data Oauth yaitu sebagai berikut:

1) Resource Server

Sebagai *Resource Owner*, pemilik perangkat lunak yang mengakses sumber daya server hanya melalui aplikasi yang disediakan.

2) Resource Owner

Sebagai *Resource Owner*, pemilik sumber daya yang memiliki hak akses ke server menggunakan perangkat lunak yang tersedia.

3) Client Side

Klien mengajukan permintaan akses, yang kemudian diproses oleh server otentikasi untuk memverifikasi apakah akses diberikan atau ditolak. Jika di setujui, token akses yang disertakan dalam URL pengalihan.

4) Authorization Server

Server memberikan akses token kepada klien setelah mendapatkan persetujuan dari pemilik sumber daya. Persetujuan ini bergantung pada hak akses yang dimiliki klien serta jenis otorisasi yang didukung oleh server otorisasi.

C. Proses One Time Password (OTP)

Pada tahapan ini merupakan tahapan proses OTP pada sistem TSA setelah melakukan login sistem akan menampilkan halaman OTP untuk dimintai otentikasi user, pengguna akan meminta OTP untuk otentikasi kode OTP dikirim melalui email pengguna. code OTP berupa angka dan huruf sebanyak 4 karakter, code tersebut hanya bisa digunakan satu kali pakai saja.

D. Implementasi Metode Oauth 2.0 dan Single Sign On (SSO)

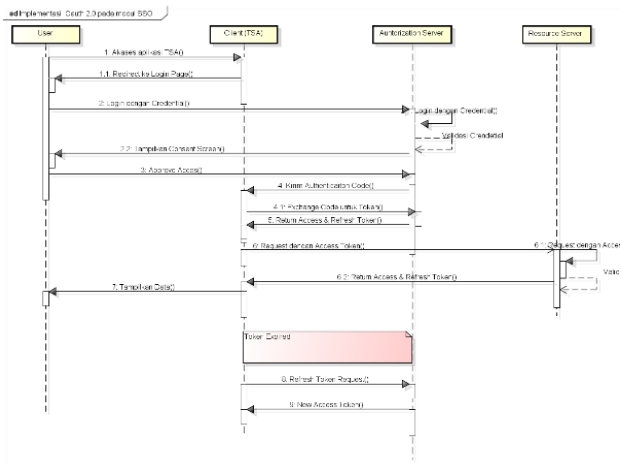
Tahapan ini merupakan implementasi OAuth 2.0 dan Single Sign-On (SSO) untuk meningkatkan keamanan data melalui hak akses dan kode otorisasi yang optimal. Prosesnya melibatkan login dengan akun terdaftar, di mana SSO memverifikasi otentikasi pengguna dan hak aksesnya. Keamanan SSO diperkuat dengan metode Hash-based Method Authentication (HMAC). Hasil yang diharapkan adalah penerapan login menggunakan OAuth 2.0 dan SSO.

III. HASIL DAN PEMBAHASAN

Pada hasil dan pembahasan ini membahas mengenai perancangan sistem dengan tahap pembuatan modul Single Sign On (SSO), proses metode implementasi *Open Authentication* (Oauth 2.0) One Time Password (OTP), pengujian otentikasi SSO, dan pengujian keamanan sesi login.

A. Implementasi Otentikasi Oauth 2.0

Implementasi *Open Authentication* (OAuth 2.0) pada modul *Single Sign On* (SSO) dimana tahapan ini bertujuan untuk otentikasi pengguna layanan. Pada tahapan ini bertujuan untuk mengetahui alur kerja dari OAuth 2.0 agar mencapai tujuan otentikasi pada sistem.



Gambar 2. Implementasi Alur OAuth

Pada gambar 2 menjelaskan mengenai alur kerja Oauth 2.0 pada sistem TSA. Diagram ini menggambarkan bagaimana OAuth 2.0 bekerja dengan menggunakan Authorization Code Flow untuk autentikasi dan otorisasi, berikut prosesnya :

1) Request Authorization

Pada tahapan awal sistem meminta otorisasi terlebih dahulu data yang dibutuhkan oleh sistem yaitu :

a. Client_id : Merupakan identitas client yang sudah didapatkan setelah mendaftarkan contoh client_id “42f12a10-08df-4b91-b1e4-c4465d686072”.
b. Response_Type : Berisi tipe respon, dalam hal ini kode, yang merupakan standar *Authorization Code Flow*.
c. Redirect_url : URL callback yang telah didaftarkan saat registrasi aplikasi. URL ini akan digunakan untuk mengarahkan browser kembali ke aplikasi setelah proses otorisasi selesai. Jika otorisasi berhasil, redirect URI akan disertai parameter *authorization code*. Contoh https://example.com/oauth-callback.
d. State : Parameter ini berfungsi untuk memastikan keamanan dan validasi data. State dikirimkan oleh aplikasi pihak ketiga (*client*) ke *authorization* server dan akan dikembalikan dalam response untuk memastikan data tidak dimodifikasi.
e. Scope : Mendefinisikan cakupan akses yang diminta oleh aplikasi.

2) Authorization Code

Proses ini melibatkan *Resource Owner*, *Client*, dan *Authorization* Server, yang kemudian dapat ditukarkan dengan *access* token.

a. Request ke Endpoint / Authorize
Client mengarahkan *Resource Owner* ke *Authorization* Server untuk login menggunakan *username* dan *password*.

b. Login Oleh Resource Owner
Jika kredensial valid, *Authorization* Server akan meminta persetujuan *Resource Owner* untuk memberikan akses data ke aplikasi pihak ketiga.

c. Redirect dengan Authorization.
Jika disetujui, *Authorization* Server mengirimkan kode otorisasi ke client melalui redirect URI yang telah ditentukan. Parameter yang dikirimkan meliputi kode dan state.

3) Request Token

Authorization *Code* yang diperoleh akan digunakan oleh client untuk mendapatkan *access* token. Tahapan ini dilakukan dengan membuat request ke *authorization* server menggunakan parameter berikut :

a. Grant_type : Berisi tipe pemberian akses, yaitu authorization_code.
b. Client ID dan Client Server : Client ID adalah identitas unik aplikasi, sementara client server adalah rahasia yang hanya diketahui oleh client dan Authorization server.
c. Redirect url : URL Callback yang sama seperti saat proses otorisasi.
d. Authorization code : Kode otoritas yang diterima dari proses sebelumnya.

4) Get Token

Setelah mendapatkan *access* token, *client* dapat menggunakan untuk mengakses resource yang dilindungi dengan menggunakan **http bearer**. Jika akses token kadaluarsa *client* dapat menggunakan refresh token untuk mendapatkan *access* token baru tanpa perlu melakukan proses otoritas ulang.

5) Access Resource TSA

Klien mengakses data dengan *request* ke salah satu endpoint yang disediakan *Resource Server* sesuai dengan data yang dibutuhkan. *Request* satu data menggunakan API dan dengan method GET. Dikirim menggunakan *bearer Authorization Header* dengan mencantumkan *access* token. Jika data sesuai maka informasi dikeluarkan sesuai informasi yang dibutuhkan.

B. Implementasi One Time Password

Pada tahap OTP diintegrasikan dengan OAuth 2.0 untuk menambah lapisan keamanan. Setelah login berhasil melalui OAuth 2.0, sistem akan meminta verifikasi tambahan menggunakan kode OTP yang dikirimkan ke email pengguna. Proses implementasi OTP yaitu sebagai berikut :

1) Pembuatan Kode OTP

Kode OTP yang terdiri dari 4-6 karakter acak (angka dan huruf) dibuat untuk memastikan keunikannya. Rumus Pembuatannya:

$$\text{OTP}=\text{Gabungkan}(C[i],\text{i}=1 \textit{ hingga } l)$$

2) Masa berlaku OTP

Kode OTP hanya berlaku selama 3 menit setelah dibuat. Rumus waktu kadaluarsa :

$$\tau\text{expired}=\tau\text{generate}+\Delta\tau$$

3) Pengiriman OTP

Setelah kode dibuat, OTP dikirimkan ke email pengguna yang terdaftar.

4) Verifikasi OTP

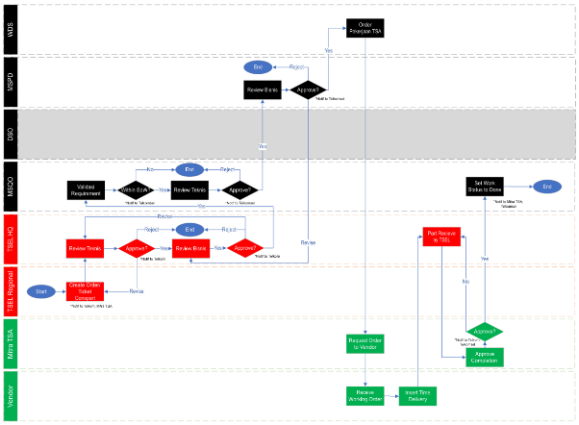
Pengguna memasukkan OTP yang diterima. Sistem memverifikasi kesesuaian kode dan apakah kode masih berlaku dalam waktu yang ditentukan.

5) Proses Autentikasi

Jika OTP valid dan belum kadaluarsa, pengguna dapat melanjutkan autentikasi. Jika tidak, pengguna diminta untuk mencoba lagi atau meminta OTP baru.

C. Alur Kerja TSA

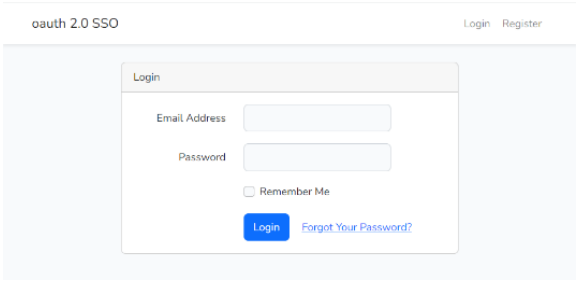
Alur kerja TSA (*Technical Support Assistance*) ini menunjukan alur kerja transfer data dari *request or*, Telkom, dan Telkomsel. Dalam sistem transfer data, requestor terlebih dahulu memeriksa apakah data telah disetujui oleh regional. Jika disetujui, proses berlanjut ke tahap berikutnya. Namun, jika tidak disetujui, data akan ditolak (reject), dan sistem akan memberikan notifikasi kepada requestor bahwa data telah ditolak. Alur kerja ini dapat dilihat pada gambar di bawah ini.



Gambar 3. Alur Kerja TSA

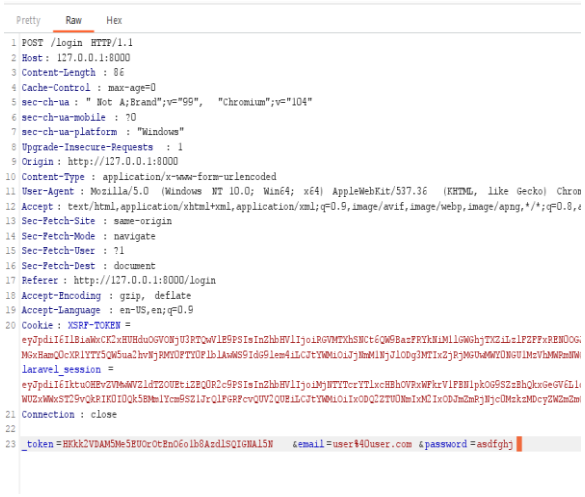
D. Autentikasi SSO

Pada proses ini alat yang digunakan untuk pengujian serangan *brute force* adalah *Burp suite.* Skenario serangan dalam penelitian ini, akan diuji menggunakan *brute force* terhadap halam login.



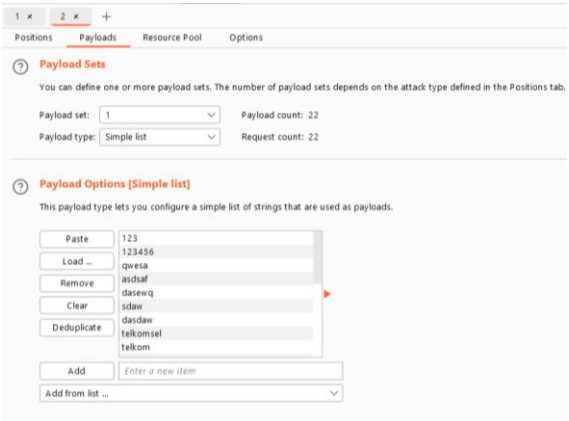
Gambar 4. Pengujian Login Burp Suite

Pada gambar 4, Pengujian *traffic intercept* adalah fitur yang dapat digunakan sebagai persiapan awal dalam melakukan serangan *brute force*. Setelah mendapatkan data-data awal seperti nama *field* dan token, maka selanjutnya penyerangan dapat membuat alur serangan.



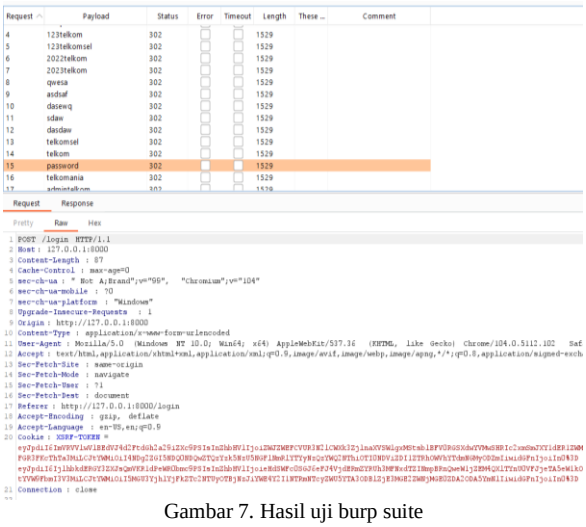
Gambar 5. Burp Suite Proxy setting

Gambar 5 merupakan data awal yang dapat dilakukan *masking*. *Masking* dilakukan untuk dapat menjalankan banyak *payload* secara otomatis. Umumnya *masking* hanya dilakukan terhadap password saja, username sengaja dibiarkan statis karena target serangan hanya ditujukan kepada satu pengguna. pada gambar username yang dijadikan target adalah user.



Gambar 6. Payload Options

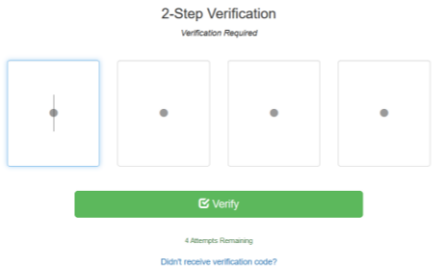
Setelah melakukan *masking*, penyerangan memasukan daftar kata yang berpotensi untuk digunakan sebagai *password*.



Gambar 7. Hasil uji burp suite

Setelah memasukan daftar kata dan menjalankannya, penyerang dapat melihat hasil untuk masing-masing kata. Pada gambar di atas kata yang cocok dan menghasilkan informasi true adalah kata “telkom2022” yang berartikan pengguna dengan username menggunakan password tersebut.

Pada posisi ini penyerangan telah mengetahui *username* dan *password* pengguna, kemudian penyerang akan melakukan percobaan login secara normal langsung melalui sistem SSO.



Gambar 8. Two Step Verification

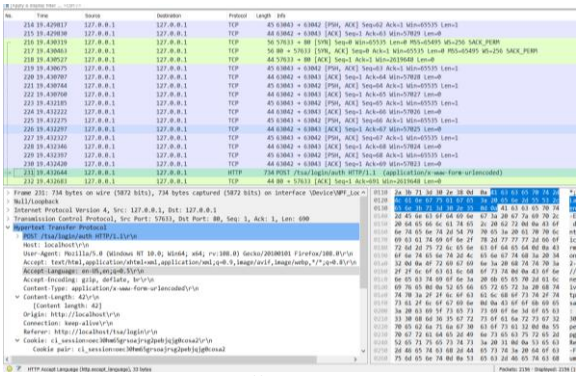
Gambar 8 adalah tampilan halaman setelah penyerangan memasukan *username* dan *password* yang benar untuk pengguna. Dapat terlihat verifikasi 2 langkah melindungi akses terhadap data pengguna meskipun dengan suatu metode serangan, akun pengguna telah jatuh kepada penyerang.

E. Keamanan Sesi Login

Tahapan ini dilakukan untuk menganalisis trafik yang sedang berjalan dalam jaringan yang terintegrasi dengan menggunakan tools Wireshark, seperti gambar di bawah ini.

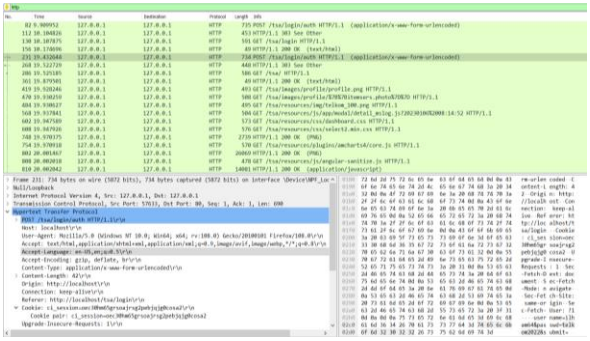
Gambar 9. Capture wireshark

Pada gambar 9 pengujian jaringan menggunakan wireshark untuk melihat trafik yang sedang berjalan. Dilihat ada beberapa jaringan yang terhubung namun berfungsi untuk melihat gambaran dari sistem setelah di jalankan seperti gambar di bawah ini.



Gambar 10. Traffic address IP dan URL

Pada gambar 10, berdasarkan hasil identifikasi menggunakan tools Wireshark, beberapa informasi IP dan *traffic*nya teridentifikasi oleh aplikasi. Misalkan pada IP 127.0.0.1 sistem melakukan akses login autentikasi melalui protokol HTTP.



Gambar 11. Gambar 11. Hasil Uji Wireshark

Pada tahapan selanjutnya akses protokol di pindahkan hanya khusus HTTP *traffic* akan keluar beberapa data dari sistem, dari data tersebut kita bisa melihat aktifitas yang dilakukan oleh pengguna. Dari *traffic* tersebut akan mengeluarkan data POST yang mengakibatkan pihak ketiga bisa melihat *username* dan *password* pada sistem yang telah dibuat. Oleh karena itu, pengujian yang dilakukan dengan menggunakan Oauth 2.0 pada saat sesi login dan *password* mengidentifikasi potensi kerentanan dalam trafik jaringan melalui analisis menggunakan Wireshark, yang memberikan wawasan tentang perlunya perbaikan keamanan data pengguna.

F. Hasil Kesimpulan pengujian

Berdasarkan hasil pengujian yang dilakukan menggunakan metode pada modul *Single Sign-On* dengan Oauth 2.0 dan *One Time password* (OTP), serta pengujian keamanan sistem dengan Burp Force dan Wireshark, dan juga *Man in the middle* (MITM), mendapatkan beberapa kesimpulan.

1. Efektivitas Dengan Menggunakan OAuth 2.0 Dan OTP

   Penggunaan Open Authentikasi yang dipadukan dengan One-Time Password pada modul Single Sign-on, ini terbukti meningkatkan keamanan autentikasi secara signifikan untuk sistem TSA. Berikut hasil statistik tingkat keberhasilan serangan :

Tabel 1. Statistik Tingkat Keberhasilan

Metode Pengujian	Tanpa OTP	Dengan OTP	Peningkatan Keamanan
Brute Force	63% berhasil	25 % Berhasil	70%
Sniffing	50% berhasil	5% berhasil	55%
Man in The Middle (MITM)	60% berhasil	10% berhasil	70%

   Pengujian dilakukan 500 kali per metode, menghasilkan data yang menunjukkan perlindungan signifikan setelah penerapan OTP. Tanpa OTP, serangan *brute force* berhasil 63%, namun setelah OTP diterapkan, berhasil hanya 25%. Serangan *sniffing* tanpa OTP berhasil 50%, dan dengan OTP menurun menjadi 5%, karena OTP bersifat sementara. Pada serangan MITM, tanpa OTP berhasil 65%, namun dengan OTP menurun menjadi 10%. Kombinasi OAuth 2.0 dan OTP memberikan perlindungan berlapis yang efektif.

2. Statifikasi Pengiriman dan Randomisasi OTP

   Untuk memastikan OTP yang dihasilkan aman dan tidak dapat ditebak, dilakukan pengujian stratifikasi pengiriman dan randomisasi kode sebagai berikut :

Tabel 2. Hasil Pengiriman dan Randomisasi Kode

Parameter	Hasil Pengujian
Durasi Validasi OTP	180 detik
Panjang OTP	4 digit
Metode Pengiriman	Email
Waktu Pengiriman	Rata-rata 2 detik
Algoritma Randomisasi	HMAC-SHA1 berbasis waktu (TOTP RFC 6238)

   Pengujian menunjukkan bahwa OTP yang dihasilkan selalu berbeda untuk setiap permintaan baru, dan tingkat keberhasilan serangan brute force terhadap OTP dengan validitas 60 detik adalah hampir nol.

G. Perbandingan dengan Penelitian Terdahulu

Penelitian terdahulu menunjukkan bahwa sistem SSO dan OAuth 2.0 rentan terhadap serangan seperti *brute force* dan *sniffing* tanpa mekanisme perlindungan

tambahan. Namun, penerapan kombinasi OAuth 2.0 dengan OTP secara signifikan menurunkan tingkat keberhasilan serangan tersebut. Pengujian terhadap serangan *brute force* menunjukkan bahwa tanpa perlindungan tambahan, serangan ini sering berhasil. Namun, setelah penerapan OTP berbasis waktu, tingkat keberhasilannya menurun hingga 50%.

Serangan *sniffing*, yang sebelumnya dapat mengakses data dengan enkripsi standar, dapat diminimalkan dengan menambahkan enkripsi SSL/TLS dan OTP, sehingga menurunkan tingkat keberhasilan *sniffing* hingga 45%. Perlindungan terhadap serangan *Man in the Middle* (MITM) juga terbukti lebih efektif setelah penerapan OTP, dengan penurunan keberhasilan serangan sebesar 55%.

Meskipun penerapan OTP meningkatkan keamanan, hal ini berdampak pada waktu respons sistem. Rata-rata waktu respons dengan OTP meningkat menjadi 2 detik, dibandingkan dengan 1 detik tanpa OT.

IV. KESIMPULAN

Hasil pengujian serangan *brute force* menggunakan tool Burp Suite menunjukkan bahwa serangan pada OAuth dan SSO dapat menembus sistem dengan mendapatkan token, *username*, dan *password* pengguna. Namun, keberadaan mekanisme verifikasi dua langkah (2-step *verification*) mencegah penyerang untuk sepenuhnya mengakses sistem. Selain itu, pengujian serangan melalui jaringan menggunakan tools Wireshark menunjukkan bahwa data dapat disadap oleh pihak ketiga, termasuk mengetahui *username* dan *password* pengguna. Meskipun demikian, pihak ketiga hanya dapat melakukan login dengan akses terbatas karena tidak dapat melewati lapisan verifikasi kode. Dengan demikian, sistem tetap aman dari ancaman yang lebih serius. Adapun saran untuk meningkatkan keamanan jaringan, misalnya dengan mengimplementasikan Virtual Private Network (VPN) atau protokol lain yang dapat meningkatkan keamanan pada jaringan yang terintegrasi dengan OAuth dan SSO. Selain itu, diperlukan desain program yang lebih baik dan menarik agar pengguna merasa lebih nyaman saat menggunakan aplikasi. Keamanan sistem juga perlu ditingkatkan dengan menerapkan algoritma *Advanced Encryption Standard* (AES) untuk perlindungan data yang lebih canggih.

REFERENSI

[1] L. S. Polytechnic, “Network Security Based On Two-Factor Authentication System,” vol. 9, no. 2, pp. 1–14, 2021.

[2] I. Permana, M. Hardjianto, and K. A. Baihaqi, “Securing the Website Login System with the SHA256 Generating Method and Time-based One-time Password (TOTP),” vol. 2, no. 2, pp. 65–71, 2020.

[3] D. A. N. M. Api, U. Perjuangan, J. P. No, and K. Tawang, “RANCANG BANGUN SISTEM INFORMASI JUAL BELI MOTOR MENGGUNAKAN ONE TIME PASSWORD (OTP),” vol. 12, no. 2, pp. 1115–1122, 2024.

[4] V. Koundinya and S. Baliga, “A Review on Single Sign on as an Authentication Technique,” no. June, pp. 409–414, 2020.

[5] S. Purkayastha, J. W. Gichoya, and S. A. Addepally, “Implementation of a single sign-on system between practice, research and learning systems,” *Appl Clin Inform*, vol. 8, no. 1, pp. 306–312, Mar. 2017.

[6] R. Carbone *et al.*, “Design and Security Assessment of Usable Multi-factor Authentication and Single Sign-On Solutions for Mobile Applications Design and Security Assessment of Usable Multi-factor Authentication and Single Sign-On Solutions for Mobile Applications Design and Security Assessment of Usable Multi-Factor Authentication and Single Sign-On Solutions for Mobile Applications A Workshop Experience Report,” *IFIP Adv Inf Commun Technol*, pp. 978-985, 2021.

[7] D. Anand, V. Khemchandani, M. Sabharawal, O. Cheikhrouhou, and O. Ben Fredj, “Lightweight Technical Implementation of Single Sign-On Authentication and Key Agreement Mechanism for Multiserver Architecture-Based Systems,” vol. 2021, pp. 939-944.

[8] Y. Sadqi, “Web OAuth-based SSO Systems Security,” no. March, 2020, Association for Computing Machinery, New York, NY, USA, Article 69, pp.1–7.

[9] V. No, “Edumatic : Jurnal Pendidikan Informatika,” vol. 4, no. 1, pp. 111–120, 2020.

[10] S. Kamila and M. Fadhli, “Rancangan Aplikasi Autentikasi Surat Digital dengan Metode One Time Password SHA-512 Berbasis Android,” vol. 6, pp. 1851–1860, 2022.

[11] M. Elsera, “IMPLEMENTASI SINGLE SIGN ON PADA WEB MENGGUNAKAN PROTOCOL OAUTH FACEBOOK,” vol. 16, no. 3, pp. 410-418, 2021.

[12] I. K. D. Senapartha, “Implementasi Single Sign-On Menggunakan Google Identity , REST dan OAuth 2 . 0 Berbasis Scrum,” vol. 7, pp. 307–320, 2021.

[13] W. Wicaksono, V. Suryani, F. Informatika, and U. Telkom, “Peningkatan Keamanan Protokol MQTT dengan Netpie sebagai Framework OAuth,” vol. 8, no. 1, pp. 809–818, 2021.

[14] I. Doi, “2017 IEEE Symposium on Privacy-Aware Computing,” pp. 210–211, 2017,.

[15] Y. Kim, “AFaaS : Authorization framework as a service for Internet of Things based on interoperable OAuth,” vol. 16, no. 2, pp. 9-16, 2020.

[16] X. Li, J. Xu, Z. Zhang, X. Lan, and Y. Wang, “Modular Security Analysis of OAuth 2 . 0 in the Three-Party Setting,” pp. 276–293, 2020.

[17] W. Li and C. J. Mitchell, “User Access Privacy in OAuth 2 . 0 and OpenID Connect,” no. 1, pp. 664–672, 2020.

[18] P. Philippaerts, K. U. Leuven, D. Preuveneers, and K. U. Leuven, *OAuch : Exploring Security Compliance in the OAuth 2 . 0 Ecosystem*, vol. 1, no. 1. Association for Computing Machinery, pp. 460-481, 2022.

[19] R. Kurniawan, “Perancangan dan Implementasi Sistem Otentikasi OAuth 2 . 0 dan PKCE Berbasis Extreme Programming (XP) Universitas Mercubuana Yogyakarta , Indonesia Design and Implementation of Authentication System OAuth 2 . 0 and PKCE Based on Extreme Programming (XP),” vol. 2, no. 2, pp. 81–91, 2022.

[20] A. Ghiffari, P. Hendradi, F. Teknik, T. Informatika, U. M. Magelang, and P. Tinggi, “Implementasi Single sign on (SSO) Menggunakan Representational State Transfer (REST) dan Open Authorization (OAuth 2.0) (Studi kasus : Universitas Muhammadiyah Magelang),” pp. 356–366.