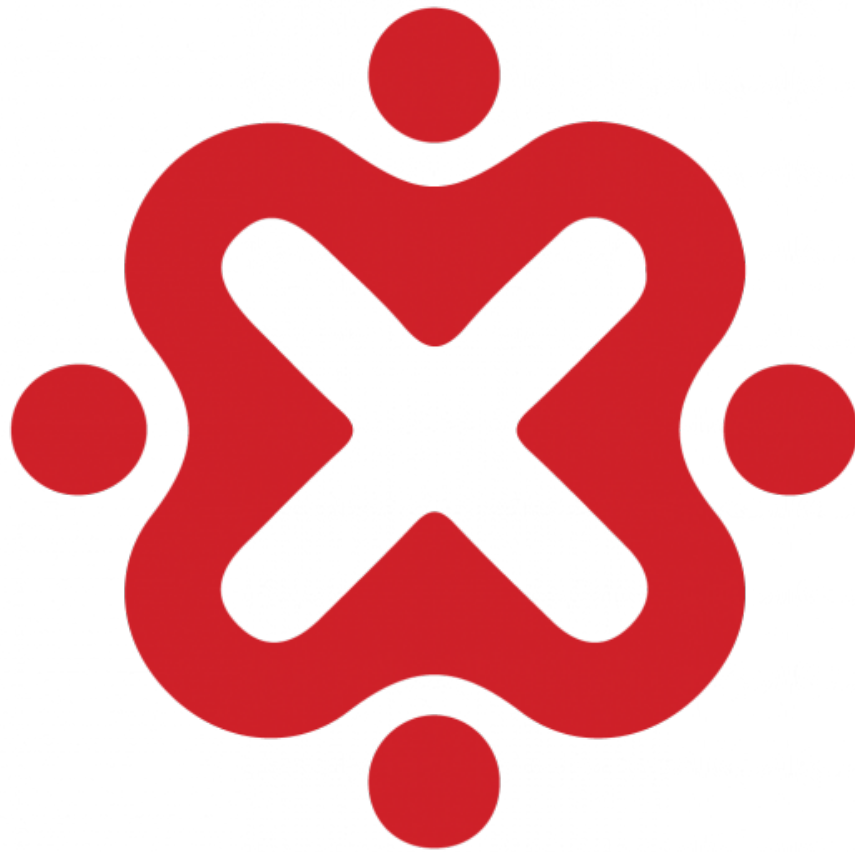


Praktikum

Pengujian Penetrasi terhadap Risiko Keamanan API pada

Laboratorium *Damn Vulnerable API (DVAPI)*



ID-Networkers
Indonesian IT Expert Factory



Dokumentasi Praktikum Pengujian Penetrasi

1. Informasi Umum

- **Nama Peserta:** Himawan Imtikhan Azmi
- **Tanggal Praktikum:** 23 – 29 Juni 2025
- **Nama Praktikum:** Pengujian Penetrasi terhadap Risiko Keamanan API pada Lab. DVAPI
- **Target Sistem:** Laboratorium DVAPI berbasis Mesin Virtual
- **IP/URL Target:** 192.168.56.12:3000 (Server Lab. DVAPI berbasis kontainer Docker)

2. Tujuan Praktikum

Untuk menguji pengetahuan dan keterampilan dalam mengidentifikasi dan mengurangi kerentanan keamanan umum dalam implementasi API.

3. Tools dan Bahan

- Tools utama: Burp Suite Community Edition v2025.5.3, jwt-encoder-decoder (Aplikasi berbasis Web), hashcat (Aplikasi berbasis CLI), Postman.
- VM/Lab environment: Ubuntu Server 24.04.2 LTS berisi Lab. DVAPI (Target) dan Kali Linux 2025.2 (Penyerang)

4. Metodologi Pengujian

Metodologi pengujian yang digunakan dalam praktikum ini menggunakan standar dan kerangka kerja uji penetrasi NIST SP 800-115 (*Technical Guide to Information Security Testing*) yang meliputi empat tahap yaitu:

- a. Perencanaan (*Planning*): Menetapkan aturan pengujian, menentukan batasan manajemen dan teknis.
- b. Penemuan (*Discovery*): Memindai mengidentifikasi potensi kerentanan. Fase ini mencakup pengumpulan informasi dan analisis kerentanan.
- c. Serangan (*Attack*): Mencoba untuk mengeksploitasi kerentanan untuk mengonfirmasi tingkat risiko. Seringkali, fase ini akan berulang kembali ke fase *Discovery* untuk menemukan target baru setelah sebuah sistem berhasil disusupi.
- d. Pelaporan (*Reporting*): Menyusun laporan hasil pengujian dan memberikan rekomendasi mitigasi

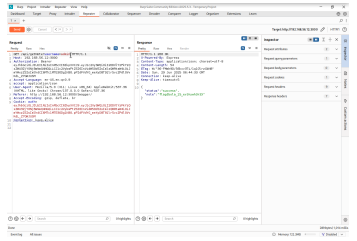


5. Langkah-Langkah Praktikum

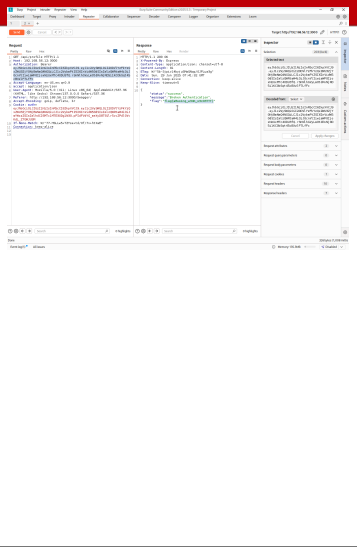
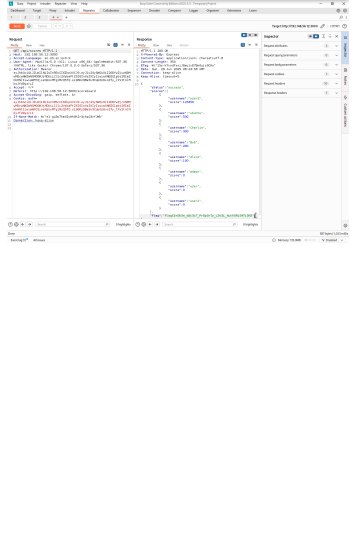
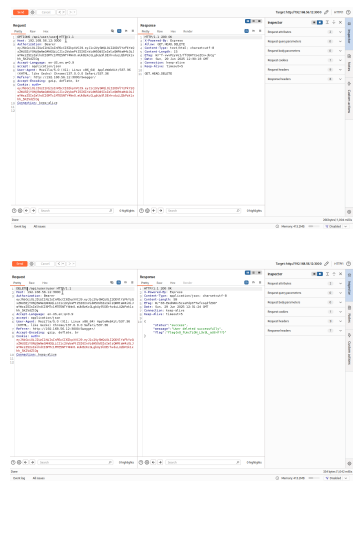
Langkah-langkah yang dilakukan dalam pengujian berdasarkan metodologi yang digunakan:

1. Mengakses *server* lab. DVAPI menggunakan peramban web yang tersedia di Burp Suite dengan fungsi pencegat (*intercept*) menyala (*on*) untuk memeriksa dan memodifikasi lalu lintas HTTP dan/ atau HTTPS antara peramban web dan *server* target.
2. Membuat akun baru untuk mendapatkan akses ke aplikasi web DVAPI sebagai pengguna baru melalui halaman *register*.
3. Melakukan pencegatan (*intercept*) lalu lintas HTTP melalui aplikasi Burp Suite saat proses *login* untuk mendapatkan token autentikasi berjenis *Bearer Token* dari pengguna yang telah dibuat sebelumnya.
4. Melakukan pengujian API berdasarkan *OWASP TOP 10 API Security Risks - 2023*

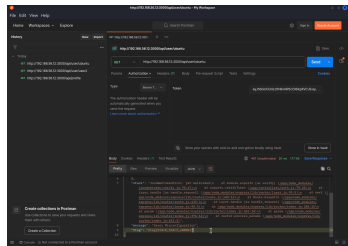
6. Temuan dan Analisis

No	Jenis Kerentanan	Deskripsi Temuan	Dampak	Bukti (Screenshot/Request)
1	API1:2023 Otorisasi Level Obyek yang Rusak (<i>Broken Object Level Authorization</i>)	URL API <code>/api/getNote?username=</code> . Parameter <i>query username</i> memiliki kerentanan otorisasi tingkat objek yang rusak. Isi <i>query</i> tersebut dapat dimanipulasi dengan mengubah nama pengguna yang lain.	Akses tanpa otorisasi ke obyek pengguna lain dapat mengakibatkan pengungkapan data ke pihak yang tidak berwenang, kehilangan data, atau manipulasi data. Dalam keadaan tertentu, akses tanpa otorisasi ke obyek juga dapat mengarah pada pengambilalihan akun total.	



2	API2:2023 Autentikasi yang Rusak (<i>Broken Authentication</i>)	URL API <i>/api/profile</i> yang berisi HTTP <i>authorization header</i> berupa <i>Bearer Token</i> pengguna memungkinkan untuk dimodifikasi menjadi pengguna lain atau pengguna fungsi administrasi dengan memanipulasi <i>JWT Token</i> pengguna. Proses autentikasi mengalami kerentanan sehingga perubahan <i>JWT Token</i> dapat diterima.	Penyerang dapat memperoleh kendali penuh atas akun pengguna lain dalam sistem, membaca data pribadi mereka, dan melakukan tindakan sensitif atas nama mereka. Sistem kemungkinan besar tidak dapat membedakan tindakan penyerang dari tindakan pengguna yang sah.	
3	API3:2023 Otorisasi Tingkat Properti Obyek yang Rusak (<i>Broken Object Property Level Authorization</i>)	URL API untuk proses pembuatan akun pengguna baru <i>/api/register</i> memuat parameter <i>username</i> dan <i>password</i> mengalami kegagalan dalam proses otorisasi masukan obyek pengguna. Sehingga ketika terdapat penambahan parameter yang tidak sah dalam proses permintaan HTTP <i>header</i> lolos dari pemeriksaan dan berhasil diproses oleh aplikasi.	Akses tanpa otorisasi ke properti obyek privat/sensitif dapat mengakibatkan pengungkapan data, kehilangan data, atau kerusakan data. Dalam keadaan tertentu, akses tanpa otorisasi ke properti obyek dapat mengarah ke eskalasi hak istimewa atau pengambilalihan akun parsial/total.	
4	API5:2023 Otorisasi Tingkat Fungsi yang Rusak (<i>Broken Function Level Authorization</i>)	Pengguna biasa dapat melakukan tindakan sensitif dengan memodifikasi metode HTTP. URL API dengan metode HTTP <i>GET</i> <i>/api/user/user</i> pada awalnya hanya untuk menampilkan data pengguna. Kemudian, melalui modifikasi metode HTTP seperti <i>OPTIONS</i> untuk menampilkan tindakan apa saja yang tersedia untuk pengguna biasa memperlihatkan	Kelemahan seperti itu memungkinkan penyerang mengakses fungsionalitas yang tidak sah. Fungsi administratif menjadi target utama untuk jenis serangan ini dan dapat menyebabkan pengungkapan data, kehilangan data, atau kerusakan data. Pada akhirnya, dapat menyebabkan	



		beberapa metode HTTP yang dapat dilakukan pengguna istimewa atau administratif seperti menghapus data pengguna lain.	gangguan layanan.	
5	API8:2023 Miskonfigurasi Keamanan (<i>Security Misconfiguration</i>)	Ketidaksesuaiin permintaan HTTP berupa <i>Bearer Token</i> yang telah dimodifikasi diproses oleh <i>server</i> dan tetap mengembalikan hasil berupa informasi kesalahan terperinci yang berpotensi dapat dieksploitasi. URL API <i>/api/user/user</i> yang diuji menampilkan informasi kesalahan yang terperinci kepada pengguna melalui permintaan HTTP.	Kesalahan konfigurasi keamanan tidak hanya mengekspos data pengguna yang sensitif, tetapi juga detail sistem yang dapat menyebabkan kompromi penuh <i>server</i> .	

7. Rekomendasi Perbaikan

Upaya yang dapat dilakukan sebagai langkah pencegahan berdasarkan *OWASP TOP 10 API Security Risks – 2023* adalah sebagai berikut:

- a) API1: 2023 Otorisasi Level Obyek yang Rusak (*Broken Object Level Authorization*)
 - 1) Terapkan mekanisme otorisasi yang tepat yang mengandalkan kebijakan dan hierarki pengguna.
 - 2) Gunakan mekanisme otorisasi untuk memeriksa apakah pengguna yang login memiliki akses untuk melakukan tindakan yang diminta pada catatan di setiap fungsi yang menggunakan input dari klien untuk mengakses catatan di basis data.
 - 3) Sebaiknya gunakan nilai acak dan tidak terduga sebagai GUID untuk ID catatan.
 - 4) Tulis tes untuk mengevaluasi kerentanan mekanisme otorisasi. Jangan terapkan perubahan yang membuat tes gagal.
- b) API2:2023 Autentikasi yang Rusak (*Broken Authentication*)
 - 1) Pastikan Anda mengetahui semua alur kemungkinan untuk mengotentikasi API (*mobile/web/tautan langsung yang mengimplementasikan otentikasi satu klik/dll.*). Tanyakan pada insinyur Anda alur apa yang Anda lewatkan.



- 2) Bacalah tentang mekanisme autentikasi Anda. Pastikan Anda memahami apa dan bagaimana mereka digunakan. OAuth bukan autentikasi, dan demikian pula kunci API.
 - 3) Jangan menciptakan ulang dalam autentikasi, pembuatan *token*, atau penyimpanan kata sandi. Gunakan standar.
 - 4) *Endpoint* pemulihan kredensial/lupa kata sandi harus diperlakukan seperti endpoint login dalam hal *brute force*, pembatasan tingkat, dan perlindungan penguncian.
 - 5) Haruskan autentikasi ulang untuk operasi sensitif (misalnya mengubah alamat email pemilik akun/nomor telepon 2FA).
 - 6) Gunakan *Cheat Sheet* Autentikasi OWASP.
 - 7) Jika memungkinkan, terapkan autentikasi multifaktor.
 - 8) Terapkan mekanisme anti-*brute force* untuk memitigasi *credential stuffing*, serangan kamus, dan serangan *brute force* pada *endpoint* autentikasi Anda. Mekanisme ini harus lebih ketat dari mekanisme pembatasan tingkat biasa pada API Anda.
 - 9) Implementasikan mekanisme penguncian akun/*captcha* untuk mencegah serangan *brute force* terhadap pengguna tertentu. Terapkan pemeriksaan kata sandi lemah.
 - 10) Kunci API tidak boleh digunakan untuk autentikasi pengguna. Mereka hanya boleh digunakan untuk autentikasi klien API.
- c) API3: 2023 Otorisasi Tingkat Properti Obyek yang Rusak (*Broken Object Property Level Authorization*)
- 1) Saat memaparkan obyek menggunakan endpoint API, selalu pastikan bahwa pengguna harus memiliki akses ke properti obyek yang Anda paparkan.
 - 2) Hindari menggunakan metode generik seperti *to_json()* dan *to_string()*. Sebaliknya, pilih properti obyek tertentu yang ingin Anda kembalikan.
 - 3) Jika memungkinkan, hindari menggunakan fungsi yang secara otomatis mengikat input klien ke dalam variabel kode, obyek internal, atau properti obyek ("Penugasan Massal").
 - 4) Izinkan perubahan hanya pada properti obyek yang seharusnya diperbarui oleh klien.
 - 5) Terapkan mekanisme validasi respons berbasis skema sebagai lapisan keamanan tambahan. Sebagai bagian dari mekanisme ini, tentukan dan paksa data yang dikembalikan oleh semua metode API.
 - 6) Pertahankan struktur data yang dikembalikan seminimal mungkin, sesuai persyaratan bisnis/fungsional untuk *endpoint* tersebut.
- d) API5: 2023 Otorisasi Tingkat Fungsi yang Rusak (*Broken Function Level Authorization*)
- 1) Aplikasi Anda harus memiliki modul otorisasi yang konsisten dan mudah dianalisis yang dipanggil dari semua fungsi bisnis Anda. Seringkali, perlindungan seperti itu disediakan oleh satu atau lebih komponen eksternal untuk kode aplikasi.
 - 2) Mekanisme penegakan harus menolak semua akses secara *default*, dibutuhkan hak akses yang eksplisit ke peran tertentu untuk mengakses setiap fungsi.



- 3) Tinjau *endpoint* API Anda terhadap kelemahan otorisasi tingkat fungsi, dengan tetap memperhatikan logika bisnis aplikasi dan hierarki grup.
 - 4) Pastikan semua pengendali administratif Anda mewarisi kendali abstrak administratif yang menerapkan pemeriksaan otorisasi berdasarkan grup/peran pengguna.
 - 5) Pastikan fungsi administratif di dalam pengendali reguler menerapkan pemeriksaan otorisasi berdasarkan grup dan peran pengguna.
- e) API8:2023 Miskonfigurasi Keamanan (*Security Misconfiguration*)

Siklus hidup API harus mencakup:

- 1) Proses pengerasan berulang yang menghasilkan penerapan lingkungan yang terkunci dengan benar dengan cepat dan mudah.
- 2) Tugas untuk meninjau dan memperbarui konfigurasi di seluruh *stack* API. Tinjauan harus mencakup: file orkestrasi, komponen API, dan layanan cloud (misalnya, izin bucket S3).
- 3) Proses otomatis untuk terus-menerus menilai efektivitas konfigurasi dan pengaturan di semua lingkungan.

Selain itu:

- 1) Pastikan semua komunikasi API dari klien ke server API dan komponen hulu/hilir terjadi melalui saluran komunikasi yang terenkripsi (TLS), tanpa memandang apakah itu API internal atau publik.
- 2) Lebih spesifik tentang verba HTTP mana pun yang dapat diakses oleh setiap API: semua verba HTTP lainnya harus dinonaktifkan (misalnya, HEAD).
- 3) API yang diharapkan diakses dari klien berbasis browser (misalnya, *front-end* WebApp) harus setidaknya:
 - mengimplementasikan kebijakan *Cross-Origin Resource Sharing (CORS)* yang tepat.
 - menyertakan *Header* Keamanan yang berlaku.
- 4) Batasi jenis konten/format data masuk hanya pada yang memenuhi persyaratan bisnis/fungsional.
- 5) Pastikan semua *server* dalam rantai server HTTP (misalnya, *load balancer*, *reverse and forward proxy*, serta *server backend*) memproses permintaan masuk dengan cara yang beragam untuk menghindari masalah desinkronisasi.
- 6) Jika memungkinkan, tentukan dan tegakkan semua skema muatan respons API, termasuk respons kesalahan, untuk mencegah pengecualian jejak dan informasi berharga lainnya dikirimkan kembali kepada pelaku serangan.



8. Evaluasi dan Refleksi

Jawab pertanyaan berikut:

- Apa tantangan utama dalam praktikum ini?

Tantangan utama dalam praktikum ini adalah keterbatasan sumber daya untuk membuat laboratorium praktikum. Selain itu, sedikitnya pengetahuan praktikan terhadap jenis kerentanan dan risiko keamanan yang ada pada *Application Programming Interface (API)* menjadi tantangan tersendiri untuk dipelajari.

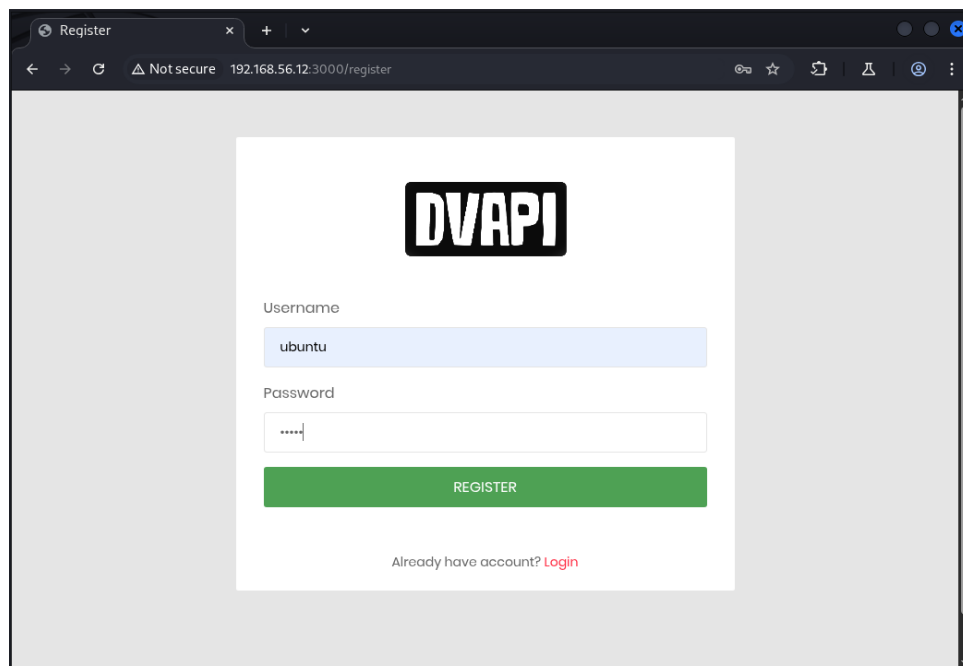
- Apakah ada tools/metode yang tidak berjalan sesuai ekspektasi?

Tidak ada, hanya saja ada sedikit kendala terkait alat yang tidak bisa berjalan karena RAM yang dialokasikan pada perangkat mesin virtual untuk penyerang (Kali Linux) terlalu kecil pada proses *Crack JWT Token*.

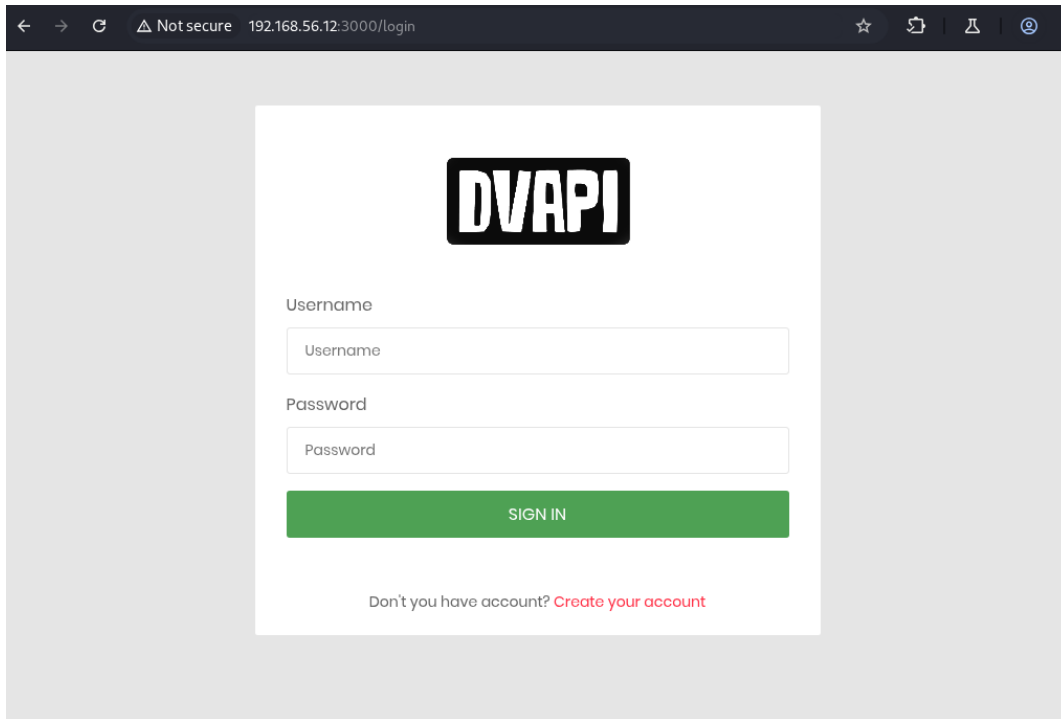
- Apa pelajaran paling penting yang dipelajari dari praktikum ini?

Pentingnya sikap teliti, cermat dan telaten dalam proses uji coba dengan sumber daya terbatas dan analisis kerentanan yang belum pernah dipelajari sebelumnya.

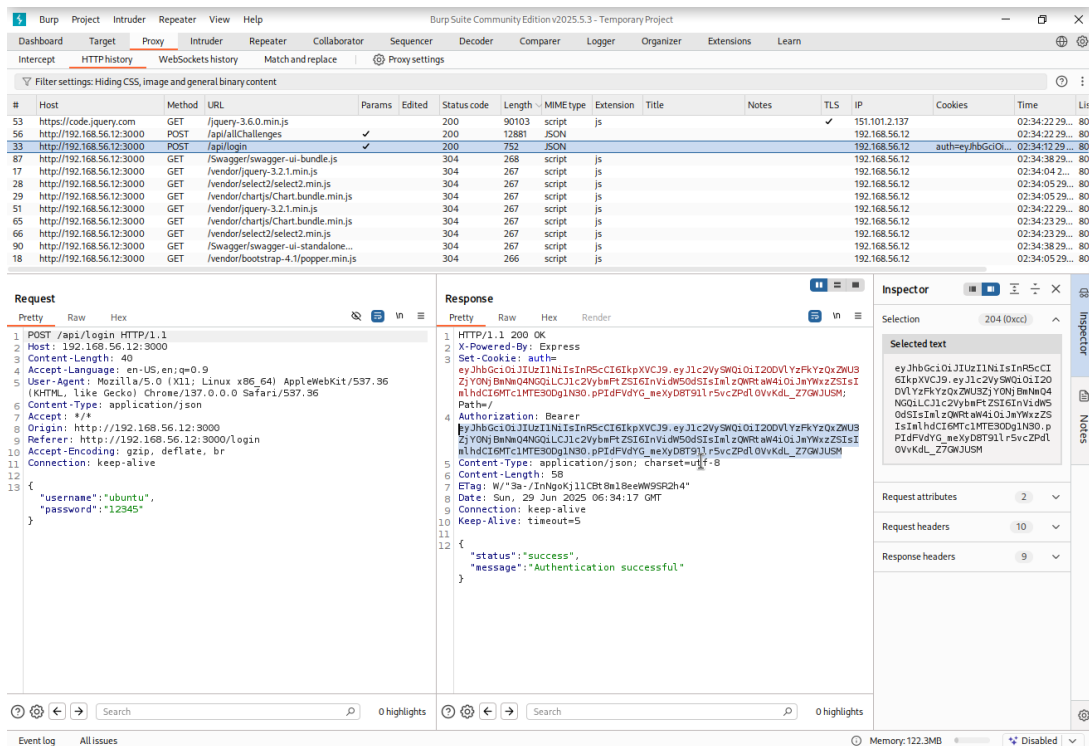
9. Lampiran



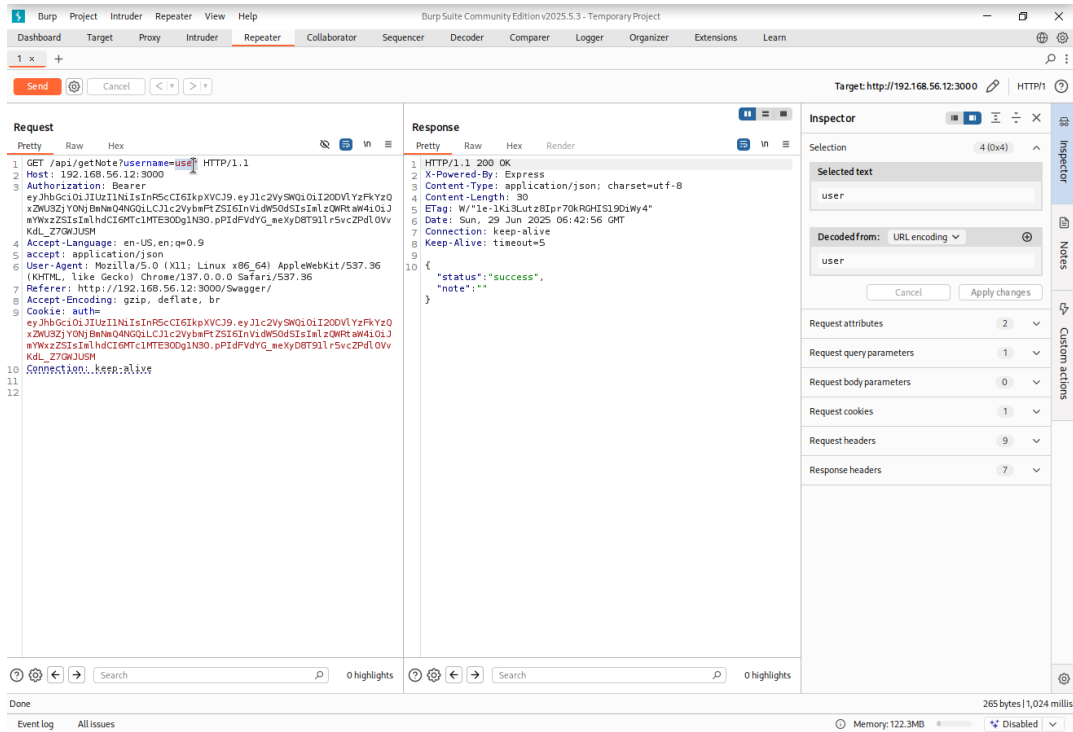
Gambar 1: Halaman Pendaftaran (*Register*)



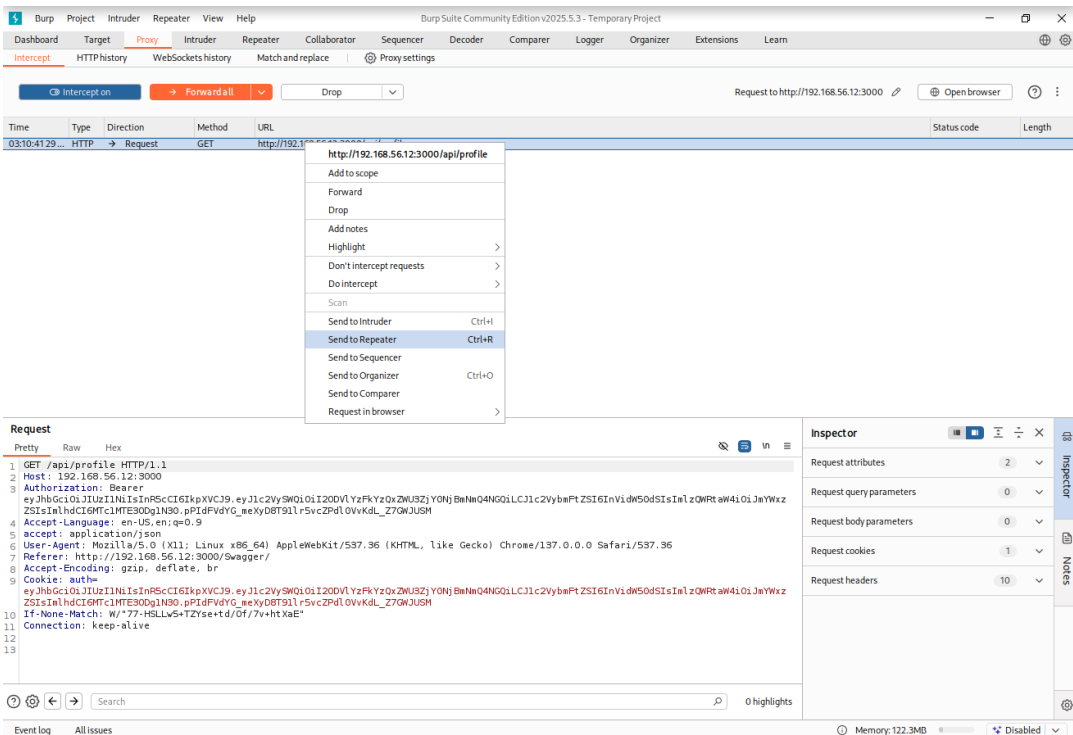
Gambar 2: Halaman Masuk (Login)



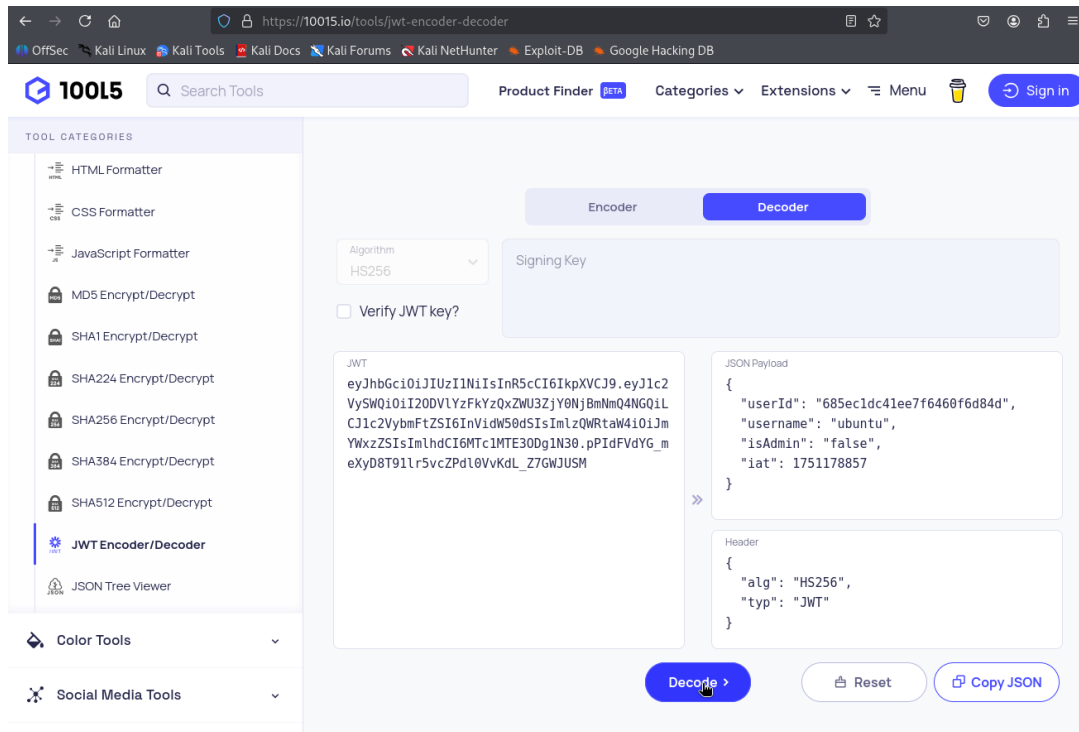
Gambar 3: Penyalinan Bearer Token



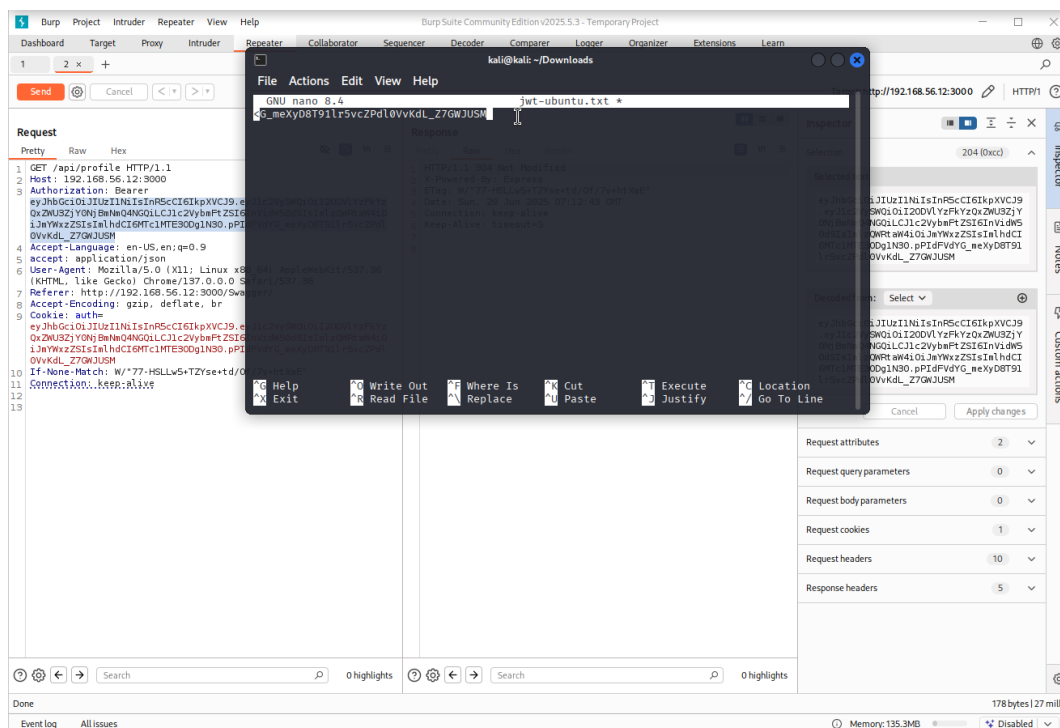
Gambar 6: Proses Uji Penetrasi dengan Repeater untuk API1:2023-BOLA



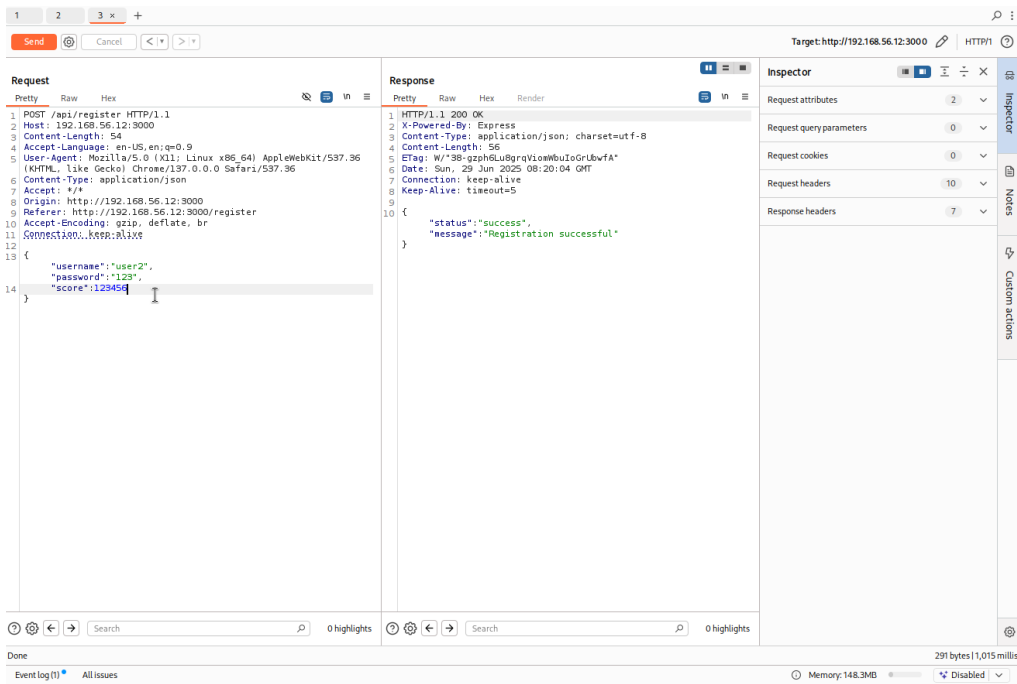
Gambar 7: Pencegatan Permintaan HTTP untuk Pengujian API2:2023-BA



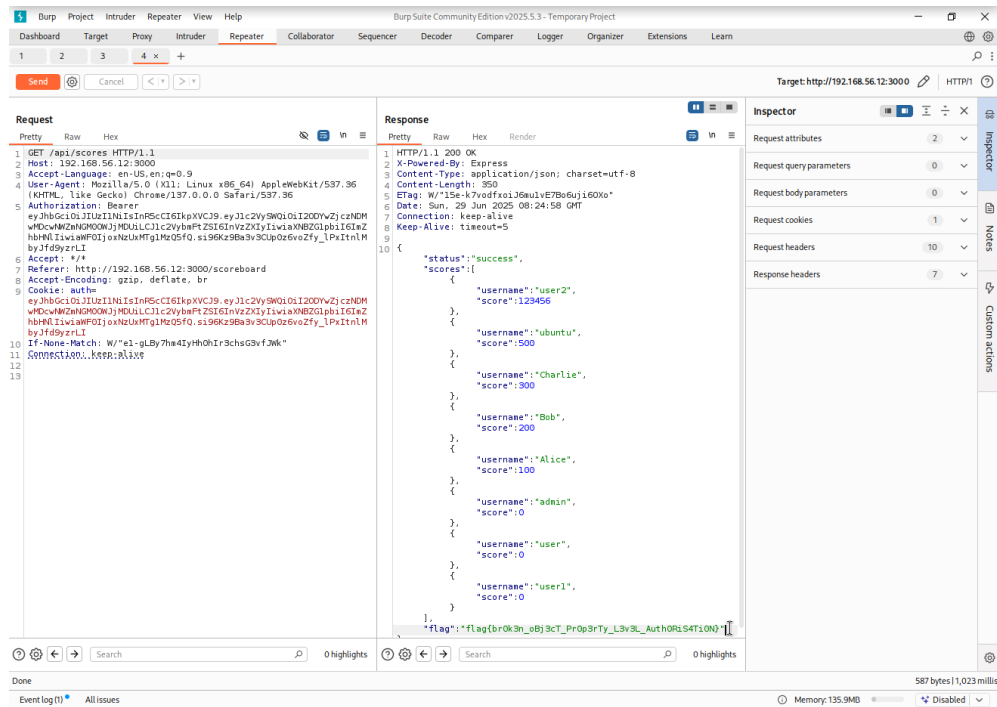
Gambar 8: Proses Decoder Bearer Token untuk Pengujian API2:2023-BA



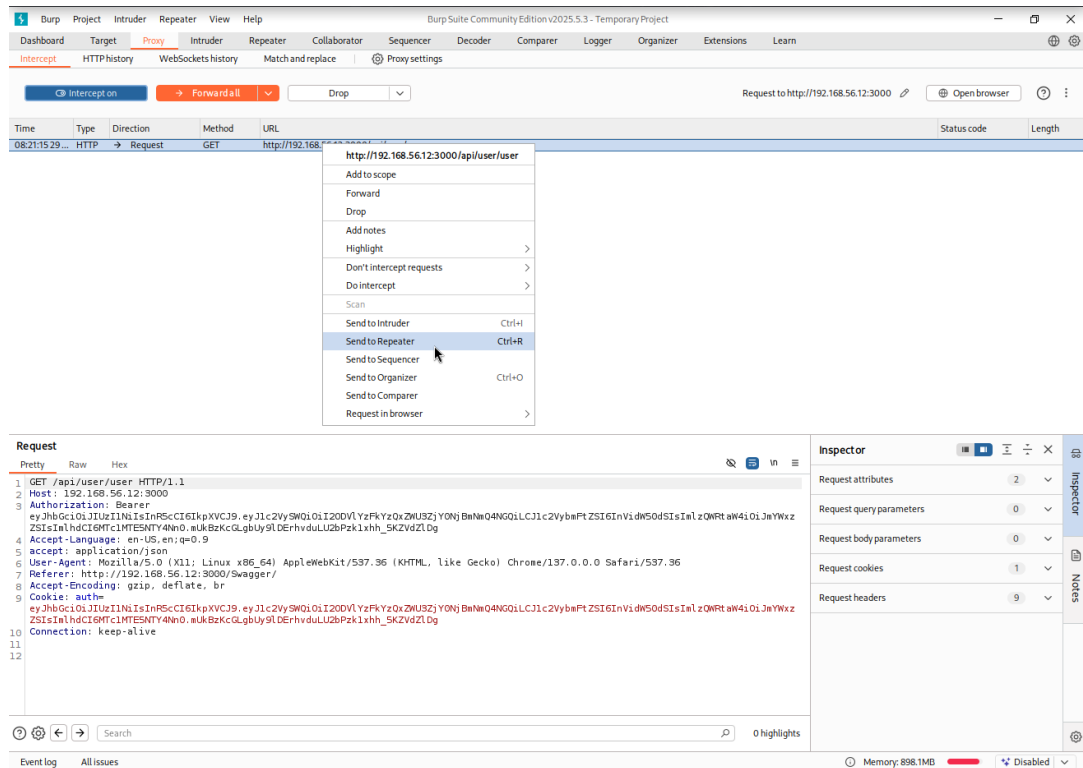
Gambar 9: Menyimpan Bearer Token untuk proses Crack Signing Key JWT Token



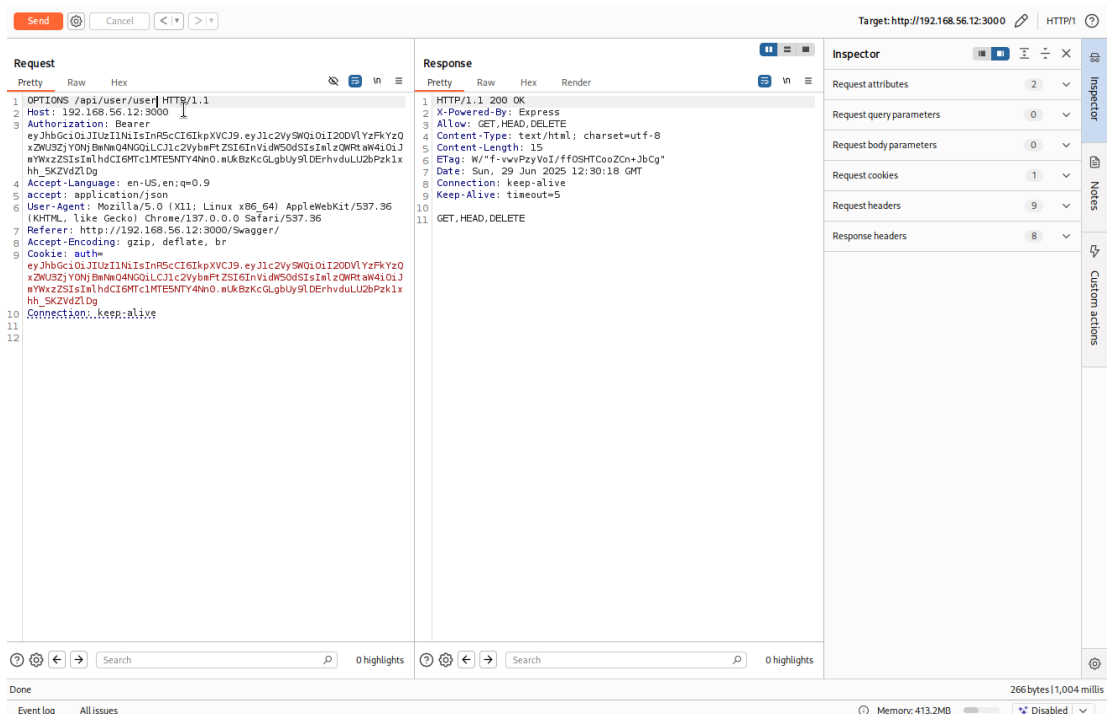
Gambar 12: Pembuatan Akun Pengguna Baru dengan Repeater untuk Pengujian API3:2023-BOPLA



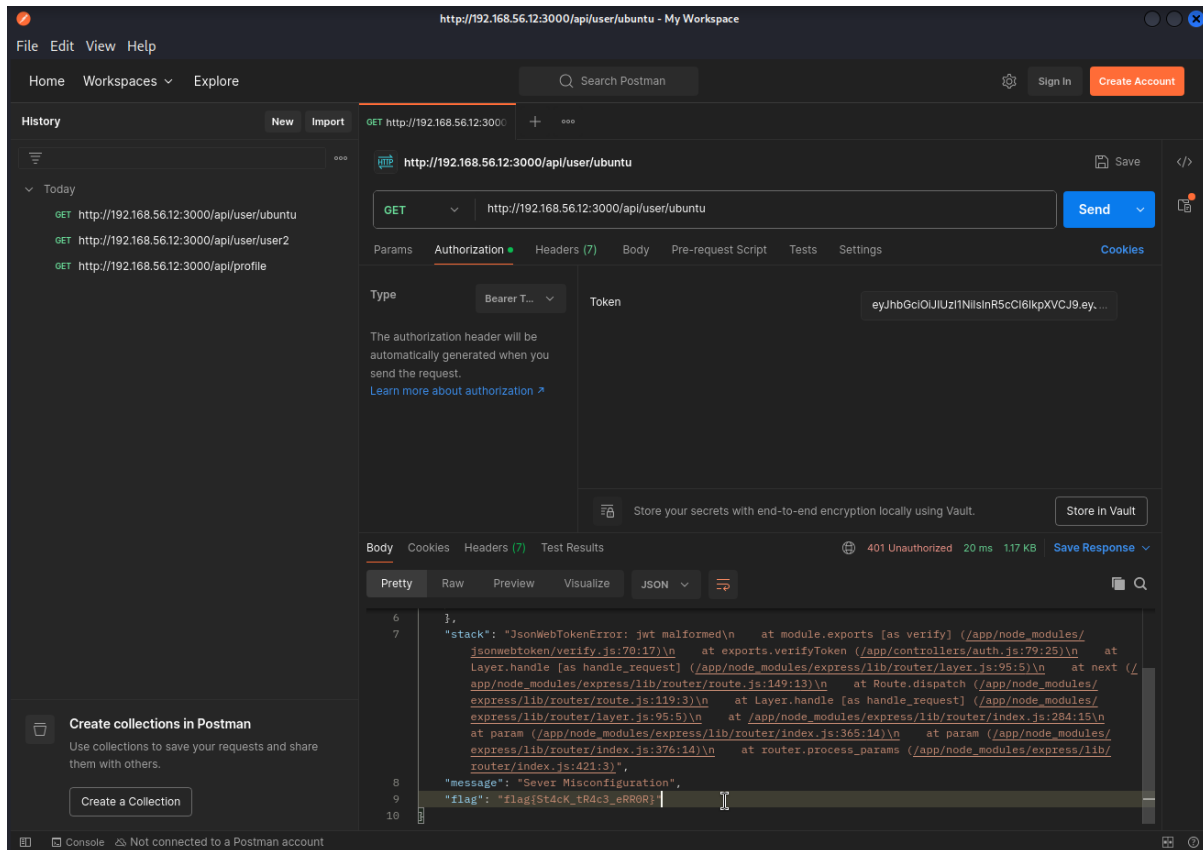
Gambar 13: Pemeriksaan Hasil Pembuatan Akun Pengguna Baru dan Hasil Pengujian API3:2023-BOPLA



Gambar 14: Pencegatan Permintaan HTTP untuk Pengujian API5:2023-BFLA



Gambar 15: Modifikasi Jenis Permintaan HTTP untuk Pengujian API5:2023-BFLA



Gambar 16: Pengujian API8:2023-SM dengan Postman

10. Referensi

- <https://owasp.org/API-Security/editions/2023/id/0x00-notice/>
- <https://github.com/payatu/DVAPI>
- <https://csrc.nist.gov/pubs/sp/800/115/final>