Laporan Penetration Testing
Coffee Portal
Mobile Application

Mei 2025 - Versi 1.0

Kelompok 10:

1. Michelle Tamara Hartono - 2702249461
2. Gendis Angel Trinichola - 2702269154
3. Bhremada Fevreano Khrisna Ardhi - 2702240241
4. Axel Nicholas - 2702229964
5. Ichwanul Ammar Al Fajri - 2702348890

Daftar Isi

Executive Summary

Dokumen tersebut adalah Laporan Pengujian Penetrasi untuk android app Coffee Portal.

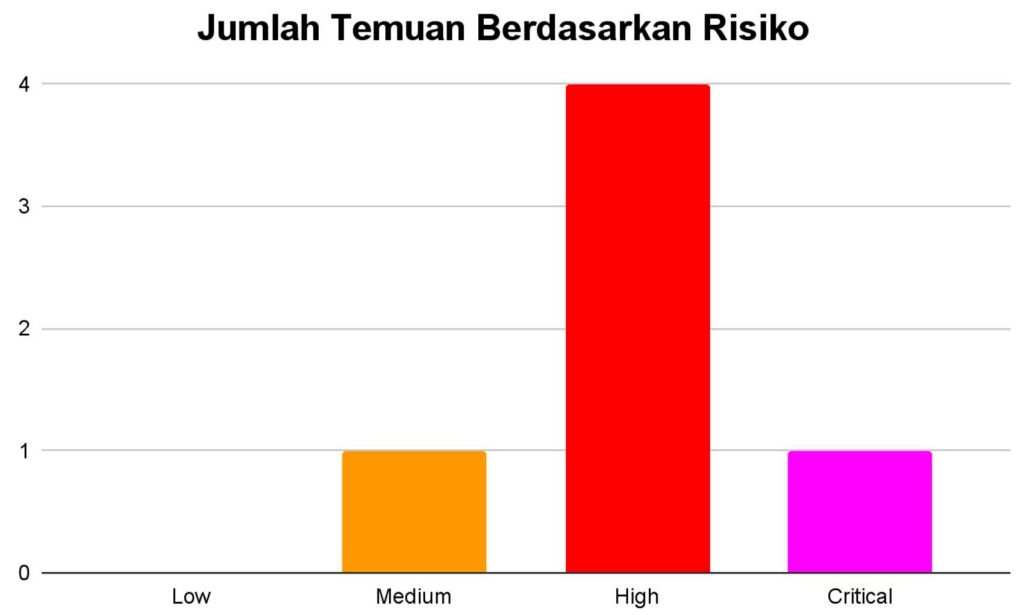
Pada laporan ini ditemukan 6 kerentanan dalam aplikasi Coffee Portal dengan tingkat yang bervariasi dari Medium hingga Critical. Berikut list kerentanan yang ditemukan:

1. Insecure Authorization (Medium): Attacker dapat melihat semua transaksi yang dilakukan oleh pengguna lain dalam aplikasi.
2. IDOR Lead to Unauthorized Cart Operations (High): Attacker dapat memanipulasi keranjang pengguna lain hanya dengan mengetahui ID pengguna tersebut.
3. Integer Overflow (High): Attacker dapat memanipulasi jumlah produk yang akan di beli, menyebabkan harga total menjadi negatif yang dapat membuat balance user menjadi sangat besar.
4. Root and Emulator Detection Bypass (High): Attacker dapat melewati pengecekan root dan emulator dengan menggunakan Frida, ini dapat menjadi celah utama untuk eksploitasi yang lebih lanjut.
5. Hardcoded Api-Key in Application (High): Kunci API terdapat di source code aplikasi dengan hal ini dapat digunakan untuk mengakses endpoint tanpa otorisasi.
6. Account Takeover (Critical): Informasi pengguna saat menggunakan aplikasi disimpan secara lokal dapat dimanipulasi untuk mengakses akun pengguna lain tanpa perlu kredensial login.

Laporan ini memberikan deskripsi terperinci tentang setiap kerentanan, dampak, dan rekomendasi untuk mitigasi. Ringkasan keseluruhan menunjukkan satu risiko Medium, empat High, dan satu Critical yang ditemukan selama pengujian penetrasi android app Coffee Portal.

Profil Temuan

Grafik di bawah menunjukkan gambaran visual hasil uji penetrasi.



Ringkasan Temuan

Satu (1) temuan ***Medium***, Empat (4) temuan ***High***, dan Satu (1) temuan ***Critical*** tercantum dibawah ini.

NO.	JUDUL TEMUAN	RISIKO
1	**Insecure Authorization**	**Medium**
2	**IDOR Lead to Unauthorized Cart Operations**	**High**
3	**Integer Overflow**	**High**
4	**Root and Emulator Detection Bypass**	**High**
5	**Hardcoded Api-Key in Application**	**High**
6	**Account Takeover**	**Critical**

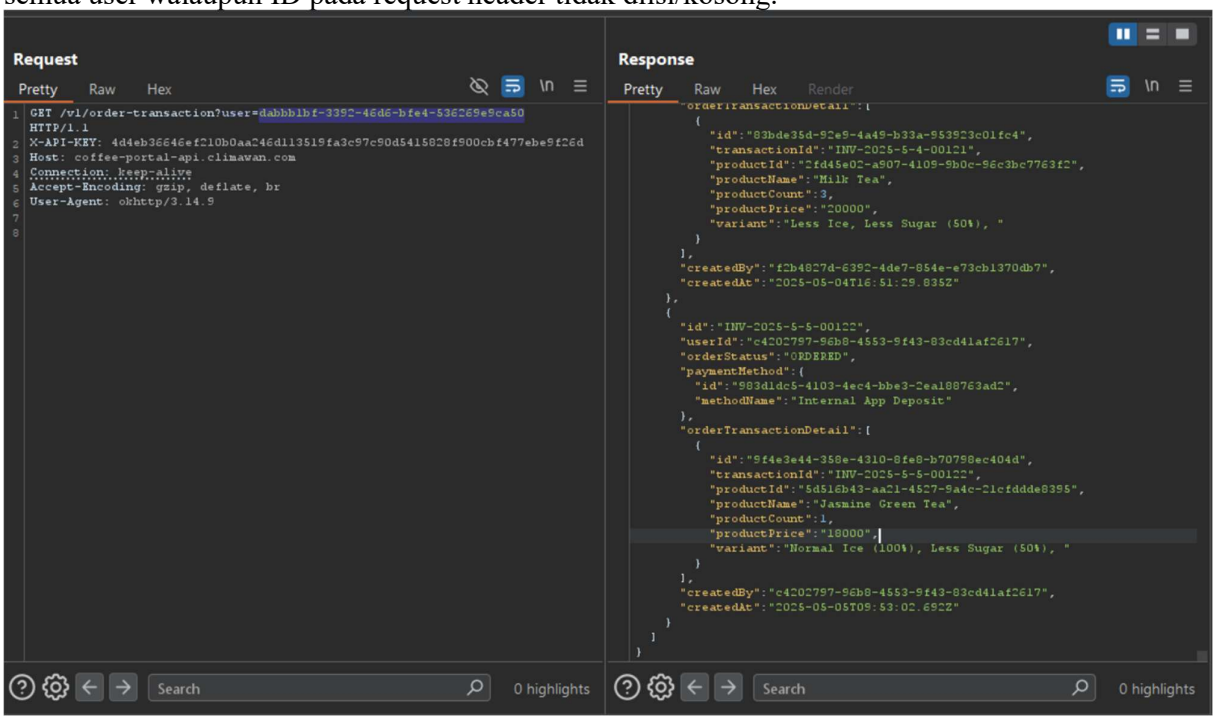
Detail Temuan

Temuan 1 – **Insecure Authorization**

Target: Coffee Portal Apps
Risiko: **CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N - 5.3 (Medium)**

Deskripsi

Terdapat bug pada bagian view transaction (pada path /v1/order-transaction?user=) dimana user bisa melihat semua transaksi yang dilakukan oleh semua user dalam aplikasi ini. Walaupun pada bagian kiri atas (yang berwarna hijau) dimasukkan ID milik Asep, kita tetap dapat melihat transaksi semua user walaupun ID pada request header tidak diisi/kosong.



Dampak

Vulnerability ini memungkinkan setiap user; bahkan tanpa autentikasi yang valid untuk mengakses seluruh data transaksi milik semua user yang menggunakan aplikasi. Hal ini merupakan pelanggaran pada prinsip kerahasiaan data (confidentiality) karena informasi pribadi seperti detail transaksi, waktu, atau jumlah pembelian dapat terekspos secara tidak sah yang jika dimanfaatkan oleh pihak yang tidak bertanggungjawab, informasi ini dapat digunakan untuk menyusun pola perilaku user, mengidentifikasi kebiasaan belanja, atau bahkan dapat mengarah ke serangan lain seperti social engineering.

Rekomendasi

Kami merekomendasikan agar aplikasi Coffee Portal:

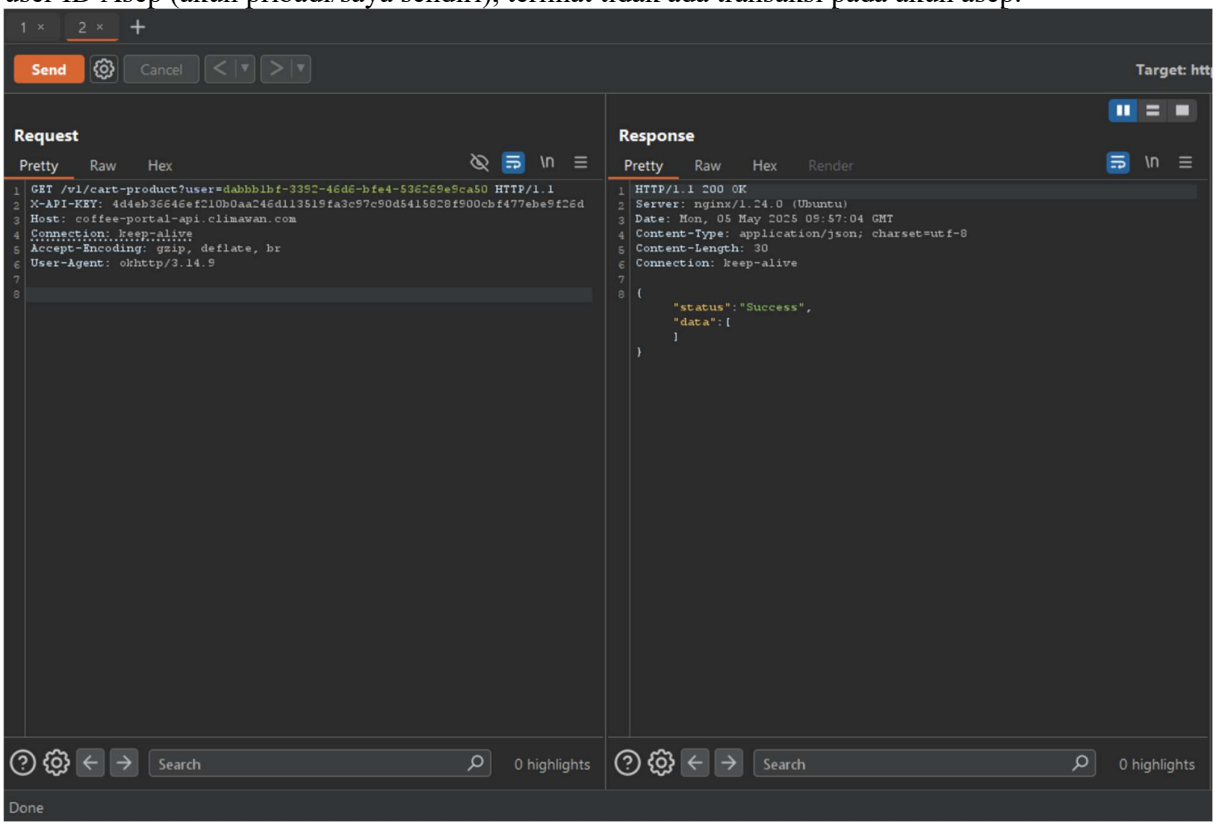
- Memastikan ulang bahwa setiap request ke endpoint yang menampilkan data transaksi harus divalidasi berdasarkan identitas user yang telah terautentikasi.
- Backend tidak boleh menerima parameter `user=` tanpa proses otorisasi yang ketat. Autentikasi dan otorisasi harus diterapkan secara menyeluruh dan dilakukan secara server-side.
- Hindari menyimpan data sensitif seperti seperti ID User, akses token, atau informasi transaksi secara hardcoded (pinning data/nilai langsung ke dalam source code) dalam aplikasi atau di local storage karena hal ini berisiko tinggi untuk dieksploitasi melalui reverse engineering.

Temuan 2 – **IDOR Lead to Unauthorized Cart Operations**

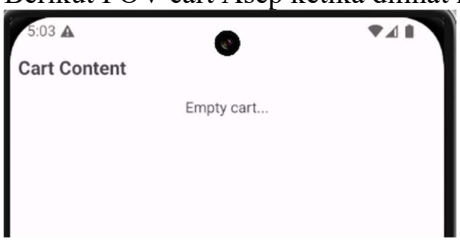
Target: Coffee Portal Apps
Risiko: **CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L- 7.3 (High)**

Deskripsi

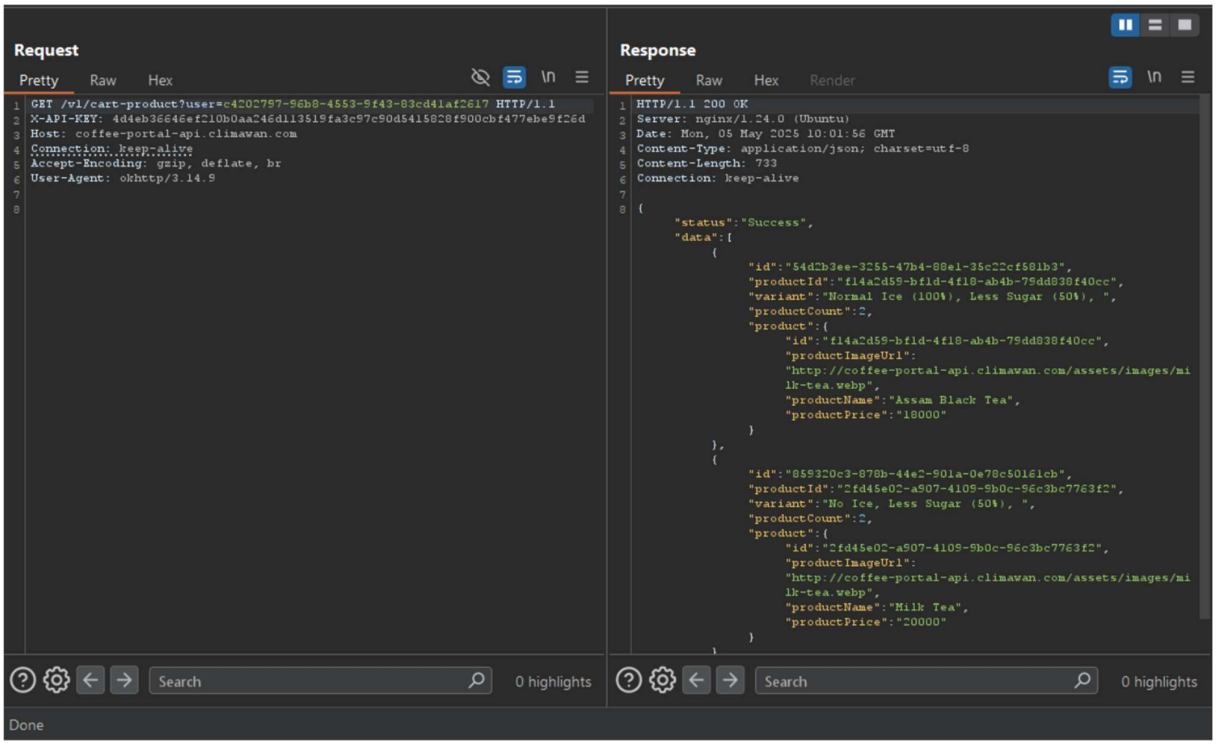
Terdapat bug IDOR dengan menggunakan user ID orang lain, pada gambar dibawah ini digunakan user ID Asep (akun pribadi/saya sendiri), terlihat tidak ada transaksi pada akun asep.



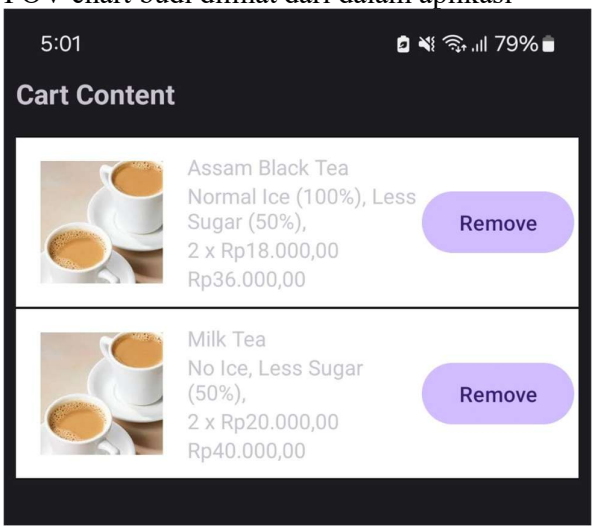
Berikut POV cart Asep ketika dilihat melalui aplikasi



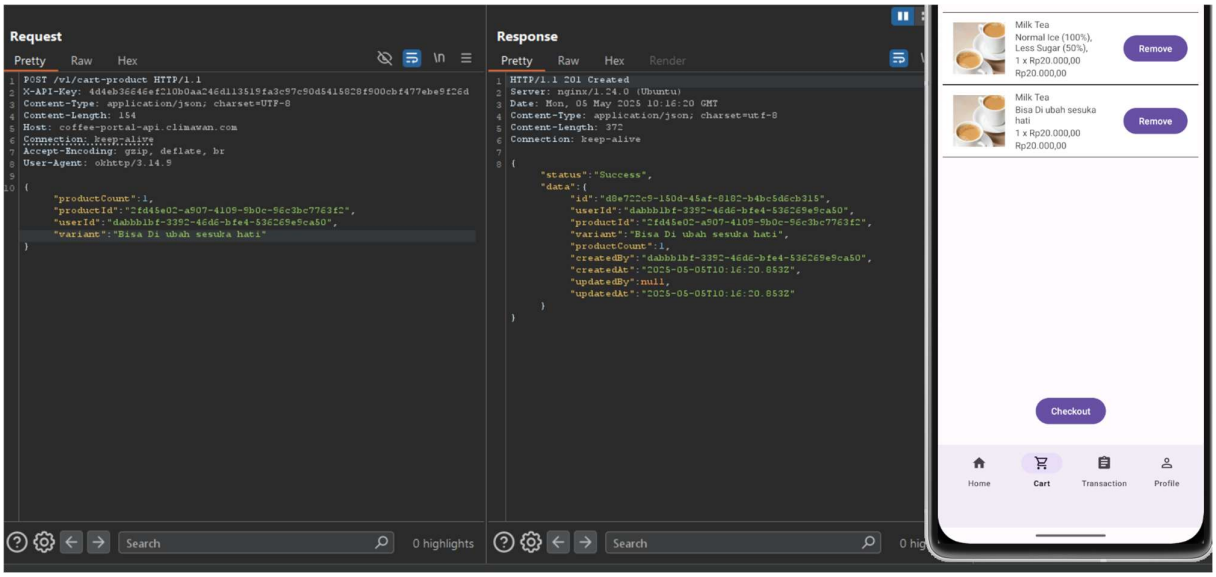
Kemudian coba request dengan ID Budi (sebagai target), maka bisa terlihat isi cart milik Budi.



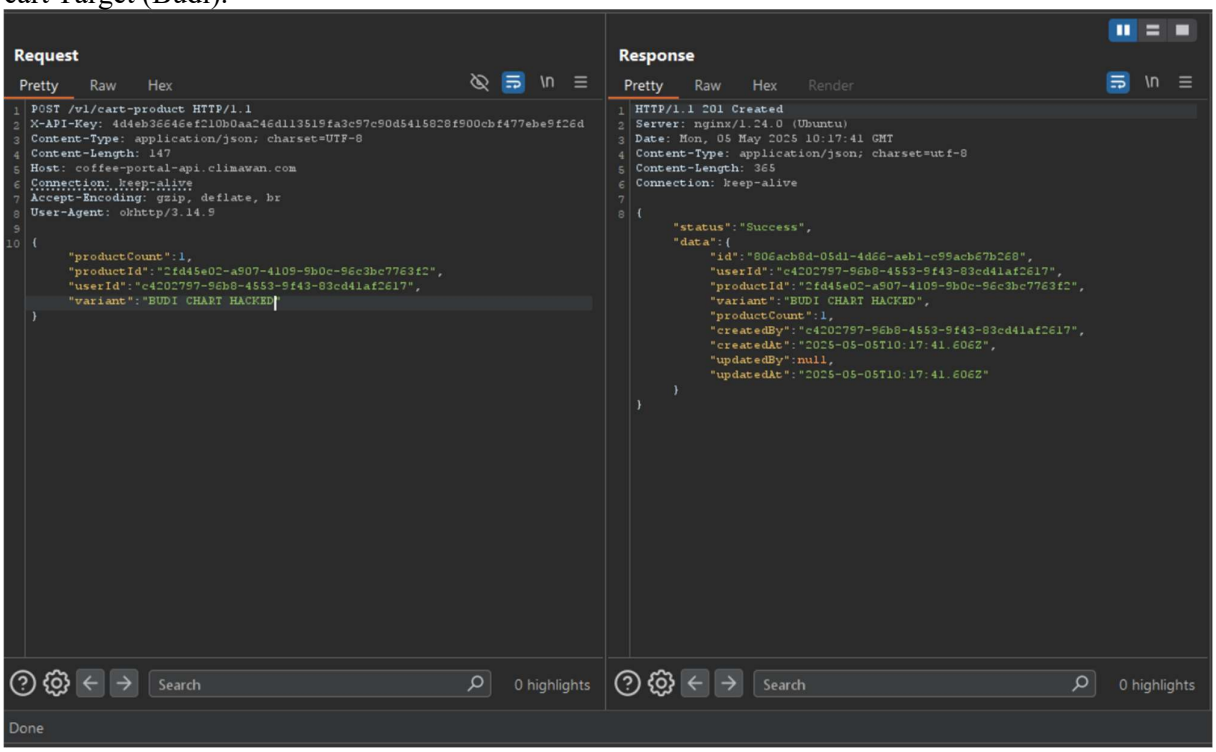
POV chart budi dilihat dari dalam aplikasi



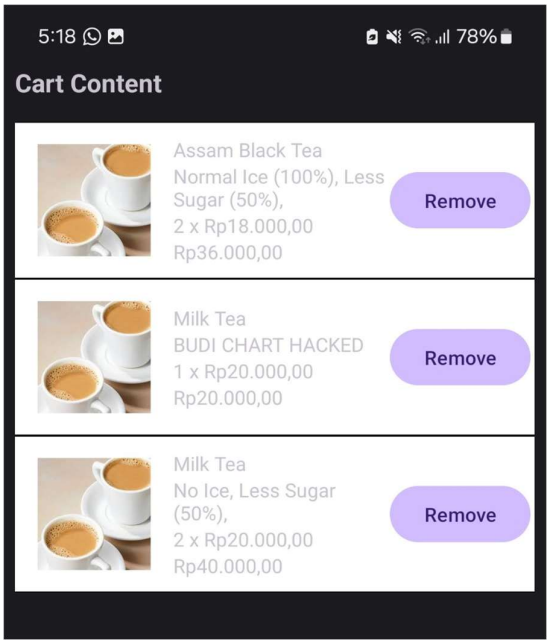
Pada /v1/cart-product, pada body request bisa kita ubah datanya sesuka hati, misal bagian variant pada saat mau menambahkan isi cart, user bisa mengubah isi variant.



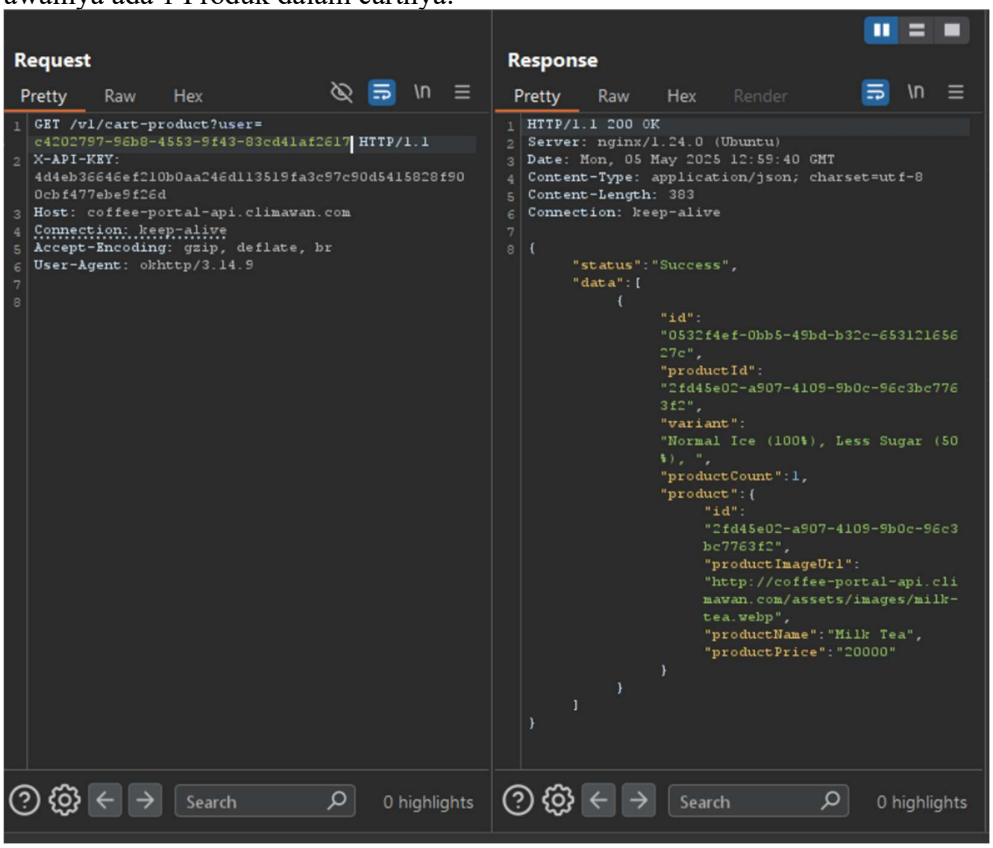
Dengan demikian, hanya dengan mengetahui user ID orang lain, attacker bisa menambahkan produk ke chart target pada endpoint /v1/cart-product. Pada gambar dibawah ini userID ganti menjadi milik Budi, dan respon dari server statusnya adalah “Success” yang berarti barang berhasil ditambahkan ke cart Target (Budi).



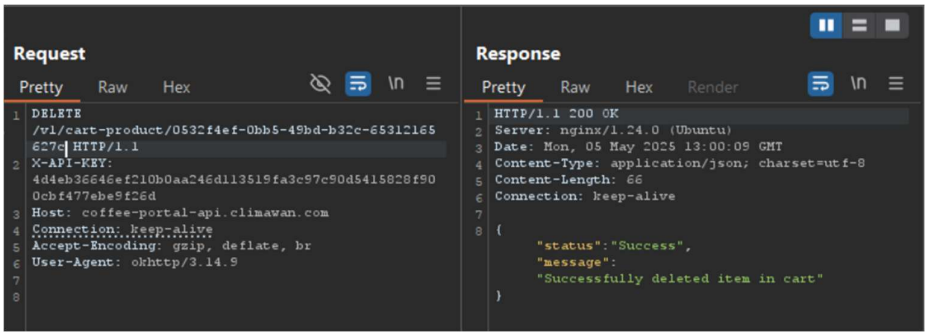
POV cart Budi setelah barang ditambahkan oleh Attacker hanya dengan mengetahui ID user Budi.



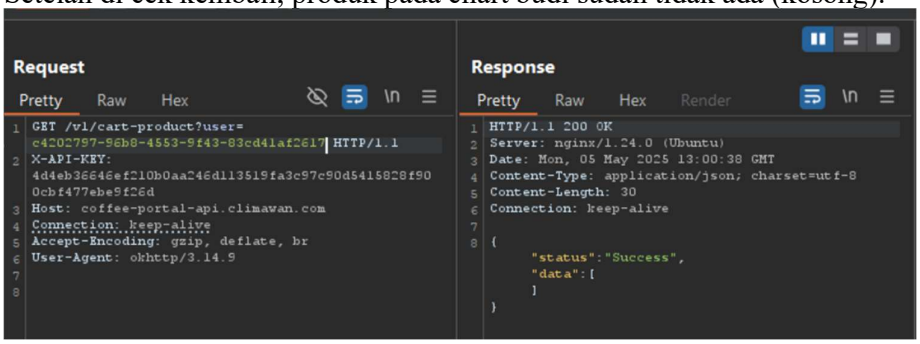
Kemudian kita coba untuk hapus produk yang ada di dalam cart Budi, berikut tampilan isi cart Budi, awalnya ada 1 Produk dalam cartnya.



Kemudian Attacker mengambil ID cart milik Budi untuk dilakukan delete pada endpoint /v1/cart-product/<id> dan request berhasil dilakukan.

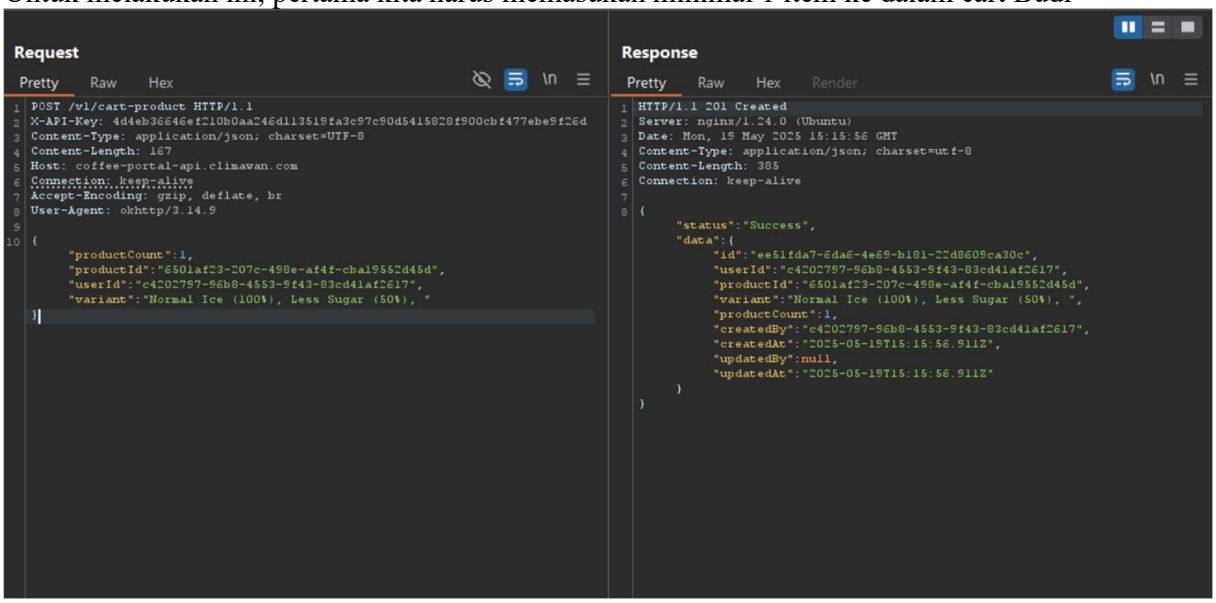


Setelah di cek kembali, produk pada chart budi sudah tidak ada (kosong).

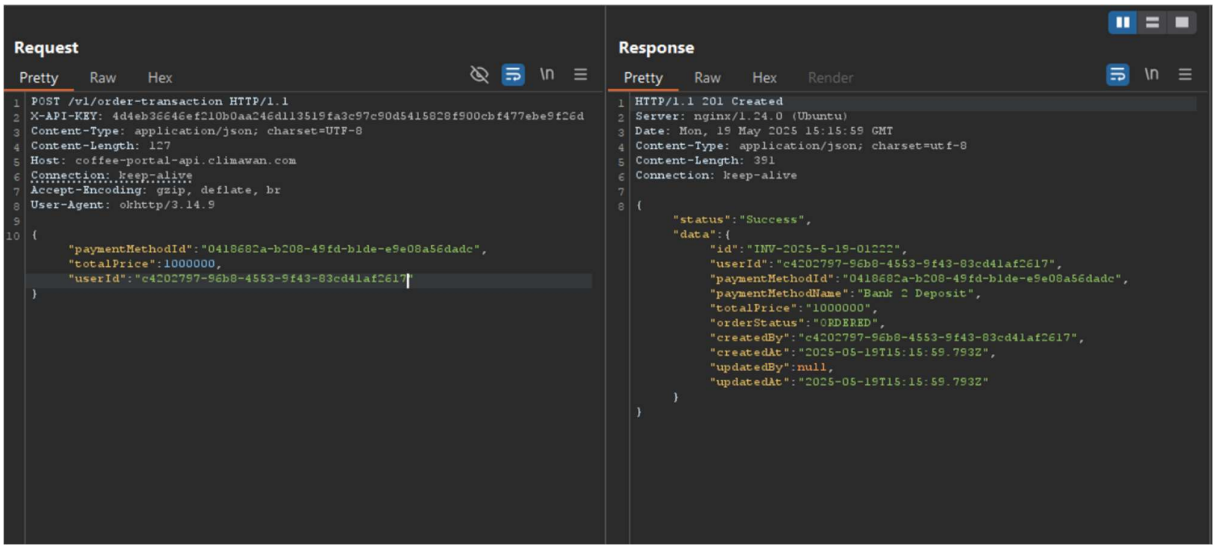


Disini kita juga bisa melakukan transaksi pembelian produk yang ada didalam cart orang lain, yang bisa dilihat disini kami berhasil melakukan pembelian product di dalam cart milik Budi, dan juga pembelian ini juga dibayar menggunakan saldo dari akun Budi, dengan total price yang bisa kita ubah sesuka hati, selama masih dalam rentang int.

Untuk melakukan ini, pertama kita harus memasukan minimal 1 item ke dalam cart Budi



Setelah itu kami pun masuk ke dalam endpoint untuk melakukan transaksi order yaitu “v1/order-transaction” dan mengganti userId yang awalnya milik asep, menjadi user id Budi.



Dan disini terlihat bahwa kita telah berhasil melakukan pembelian ke dalam product cart orang lain, dan melakukan pembayaran dengan menggunakan saldo uang si Budi

Dampak

Vulnerability ini cukup serius karena hanya dengan mengetahui ID User, attacker dapat melakukan berbagai operasi terhadap data chart. Attacker dapat membaca, menambah, menghapus, bahkan melakukan pembelian item pada cart milik user lain tanpa perlu melakukan autentikasi sebagai user tersebut. Celah keamanan ini dapat berdampak pada manipulasi pesanan, penyalahgunaan akun, kerugian finansial, hingga hilangnya kepercayaan user terhadap platform coffee-portal.

Rekomendasi

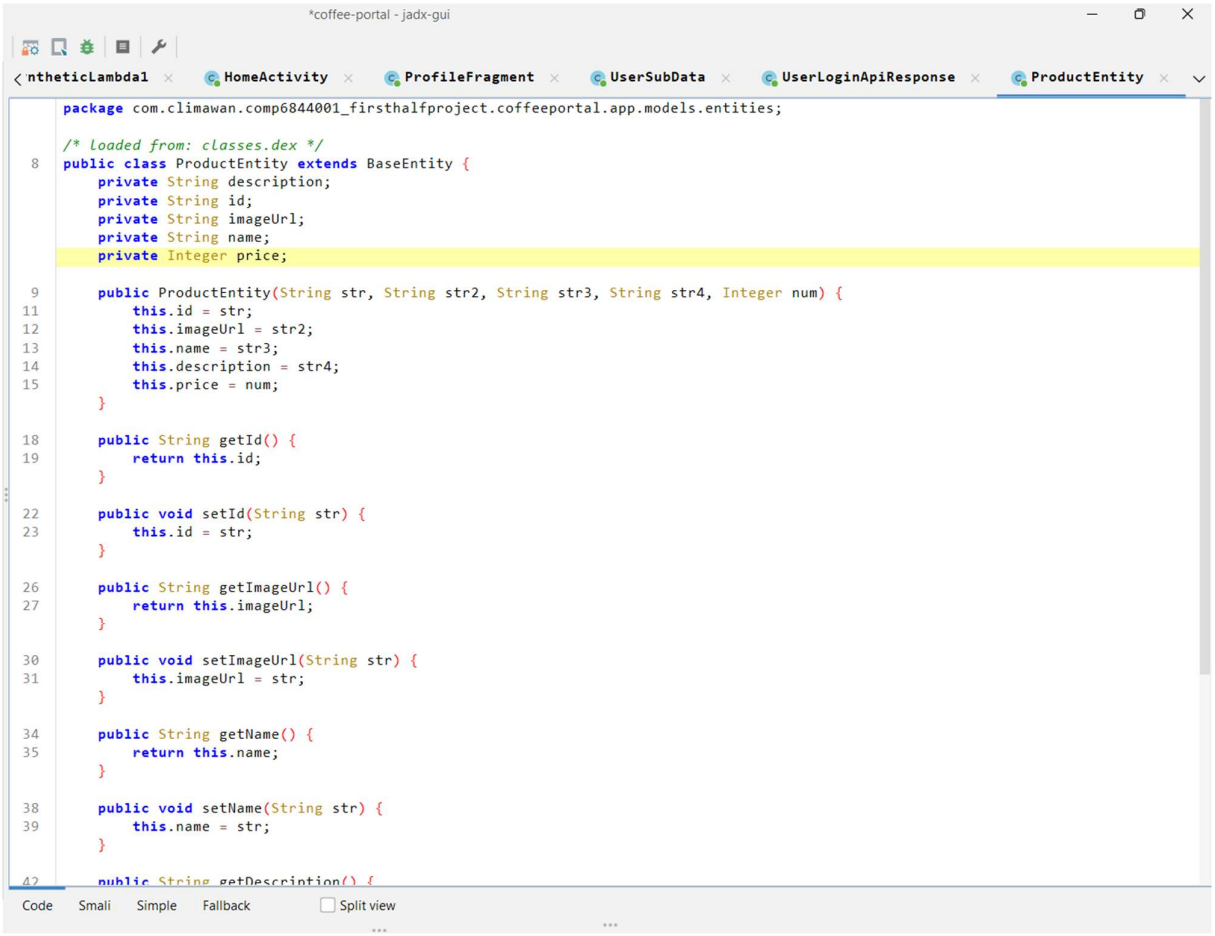
Kami merekomendasikan agar aplikasi coffee-portal ditambahkan kontrol otorisasi yang ketat pada setiap endpoint yang berinteraksi dengan resource milik user. Sistem backend harus memverifikasi bahwa ID user yang melakukan request memang memiliki akses terhadap resource yang dimaksud. Validasi ini harus dilakukan secara server-side dan tidak boleh bergantung pada data yang dikirim dari sisi klien. Audit dan logging terhadap operasi yang dilakukan pada cart user juga penting untuk deteksi dini dan forensik apabila terjadi penyalahgunaan.

Temuan 3 – **Integer Overflow**

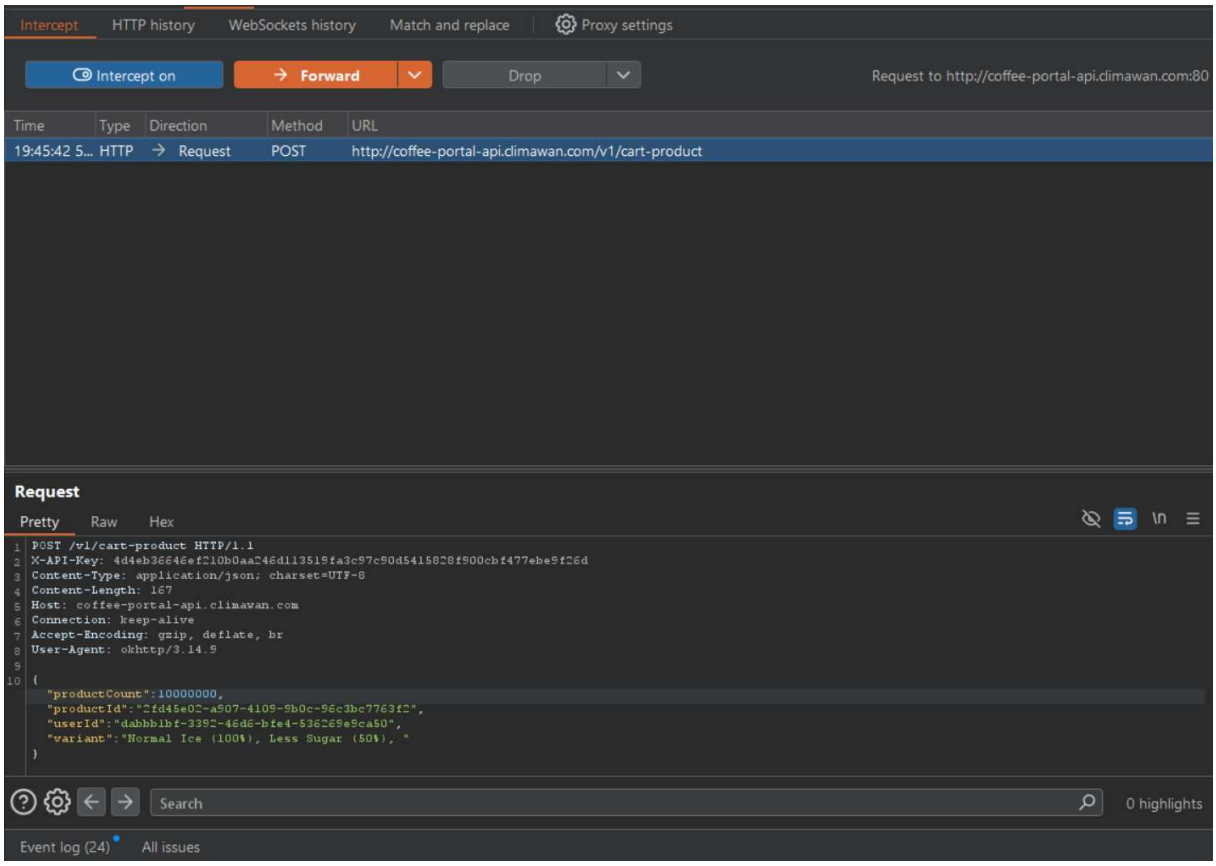
Target: Coffee Portal Apps
Risiko: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N **7.5 (High)**

Deskripsi

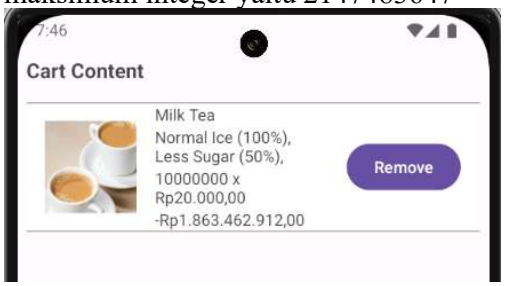
Saat decompile aplikasi pakai JAXD, bisa terlihat variabel price menggunakan tipe data integer. Dimana integer sendiri memiliki batas maksimum sebesar 2147483647 dan batas minimum -2147483647.



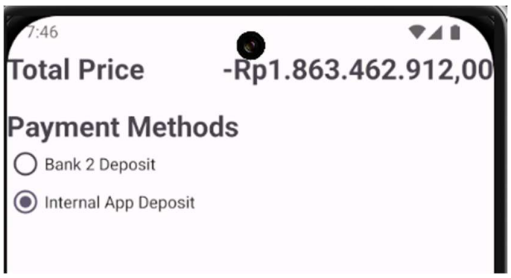
Dengan mengakses endpoint /v1/cart-product, kita bisa mengubah value dari productcount menjadi sangat banyak



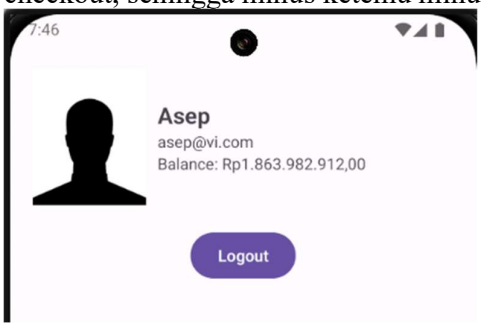
Kemudian kirim request, dan bisa terlihat di aplikasi bagian chart, sekarang kita memiliki jumlah produk yang sangat banyak sebelum di checkout. Terlihat pada jumlah biaya yang ingin di checkout menjadi negatif, karena quantity (10.000.000) dikali harga satuan (20.000) sudah melebihi batas maksimum integer yaitu 2147483647



Lakukan proses checkout dengan menggunakan internal app deposit (agar bisa melihat uang kita setelah selesai transaksi).



Disini uang Asep menjadi positif karena seharusnya prosesnya uang saat ini dikurang dengan harga checkout, sehingga minus ketemu minus jadi positif.



Dampak

Temuan ini menunjukkan adanya vulnerability integer overflow yang dapat dieksploitasi untuk memanipulasi logika transaksi finansial dalam aplikasi. Dengan mengubah nilai productcount pada endpoint /v1/cart-product menjadi jumlah yang sangat besar, perhitungan total harga (jumlah produk dikali harga satuan) akan melebihi batas maksimum tipe data integer (2.147.483.647). Akibatnya, nilai total harga menjadi negatif. Saat proses checkout dilakukan, sistem yang seharusnya mengurangi saldo justru memperlakukannya sebagai pengurangan terhadap angka negatif, yang menghasilkan penambahan saldo secara tidak sah. Hal ini berdampak langsung pada kerugian finansial aplikasi dan memungkinkan penyalahgunaan sistem deposit internal secara berulang.

Rekomendasi

Kami merekomendasikan agar aplikasi Coffee Portal:

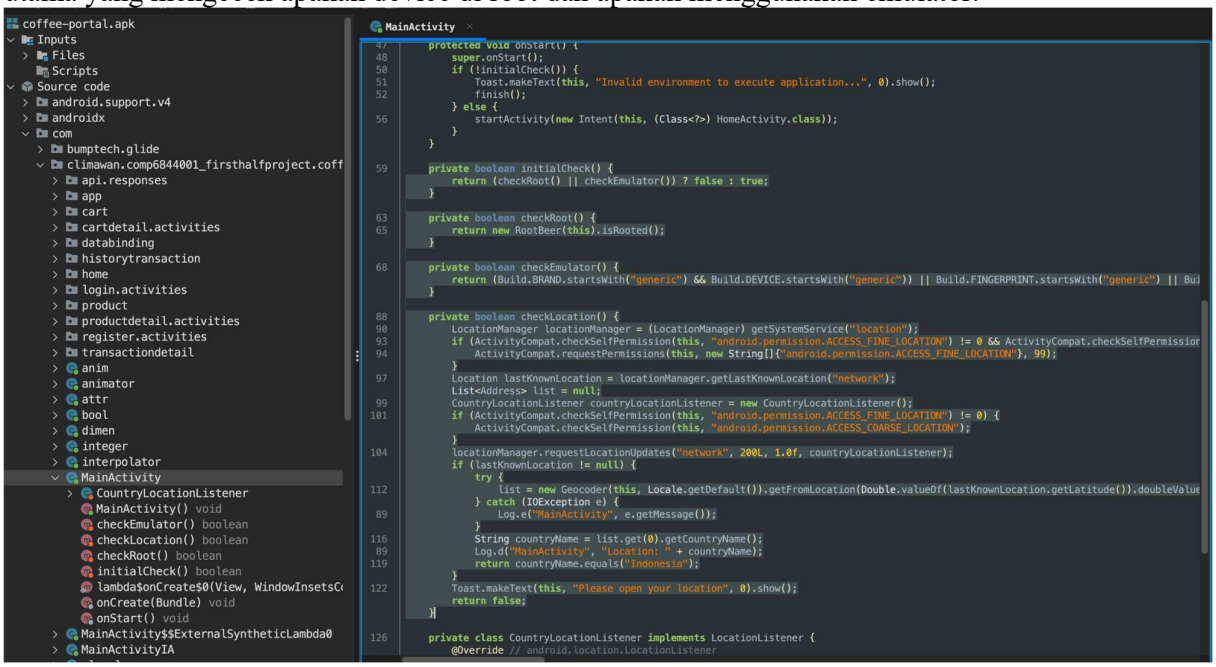
- Mengganti tipe data price dan seluruh variabel yang terlibat dalam perhitungan finansial dari integer 32-bit ke tipe yang lebih aman seperti long (64-bit) agar lebih akurat dan stabil.
- Menerapkan validasi sisi server untuk memeriksa jumlah maksimum produk yang dapat ditambahkan ke cart serta memastikan hasil perhitungan tidak melampaui batas tipe data yang digunakan.
- Menambahkan proteksi seperti pembatasan transaksi maksimum dan logging terhadap anomali harga juga penting untuk mendeteksi dan mencegah eksploitasi lebih lanjut.
- Menerapkan mekanisme anti abuse: rate limiting, threshold enforcement, logic validation, session mentoring dan anomali detection, dan audit log transaksi wajib diterapkan agar kejadian seperti temuan ini dapat ditelusuri dan dicegah dimasa mendatang.

Temuan 4 – **Root and Emulator Detection Bypass**

Target: Coffee Portal Apps
Risiko: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N **7.5 (High)**

Deskripsi

Saat melakukan decompile coffeeportal.apk menggunakan JADX, kami menemukan 2 function utama yang mengecek apakah device di root dan apakah menggunakan emulator.



Menggunakan frida, kami menjalankan script ini untuk memodifikasi hasil return function pengecekan root, emulator (pada function initialCheck), dan lokasi (pada function checkLocation) agar selalu mengembalikan nilai True.

```
Java.perform(()=>{
   let MainActivity =
Java.use("com.climawan.comp6844001_firsthalfproject.coffeeportal.MainActivity")


   MainActivity.initialCheck.implementation = () => {
       return true;
   }


   MainActivity.checkLocation.implementation = () => {
       return true;
   }
})
```

Sehingga aplikasi coffee portal bisa dibuka di emulator yang rooted.

Dampak

Dengan aplikasi yang *rooted*, kita bisa menggunakan burpsuite untuk melihat trafic dari aplikasi ke server dan sebaliknya, hal ini menjadi pintu utama bagi kami untuk menemukan kelemahan lain.

Rekomendasi

Kami merekomendasikan agar aplikasi Coffee Portal:

- Implementasi root dan emulator detection dengan code native (misal C/C++). Karena frida akan lebih sulit menangani code native dari pada code java/kotlin secara langsung
- Menerapkan teknik anti debug dan obfuscate code, agar sulit untuk memahami alur aplikasi
- Melakukan perhitungan checksum pada bagian aplikasi yang penting untuk menghindari perubahan.
- Melakukan pembaruan rutin pada library dan dependensi terkait deteksi serta implementasi validasi dan kontrol di sisi server. .

Temuan 5 – **Hardcoded Api-Key in Application**

Target: Coffee Portal Apps
Risiko: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:H/A:L **8.6 (High)**

Deskripsi

Pada saat melakukan decompile aplikasi menggunakan JADX, terlihat ada API_KEY yang langsung ditulis dalam aplikasi secara hardcode.

```
/* loaded from: classes.dex */
public class ApiRequestTask {
    public static final String API_KEY = "4d4eb36646ef210b0aa246d113519fa3c97c90d5415828f900cbf477ebe9f26d";
    private static ApiRequestTask instance;
    private BackendApiService api;
    private Retrofit retrofit;

    public ApiRequestTask() {
        Retrofit build = new Retrofit.Builder().baseUrl("http://coffee-portal-api.climawan.com").addConverterFactory(GsonConverterF
        this.retrofit = build;
        this.api = (BackendApiService) build.create(BackendApiService.class);
    }

    public static ApiRequestTask getInstance() {
        if (instance == null) {
            instance = new ApiRequestTask();
        }
        return instance;
    }

    public BackendApiService getApi() {
        return this.api;
    }
}
```

API key ini bisa kita gunakan untuk mengakses semua endpoint yang ada di aplikasi. Sehingga dapat membuka celah ke vulnerability lain di dalam API.

Request

Pretty Raw Hex

```
1 POST /v1/cart-product HTTP/1.1
2 X-API-Key: 4d4eb36646ef210b0aa246d113519fa3c97c90d5415828f900cbf477ebe9f26d
3 Content-Type: application/json; charset=UTF-8
4 Content-Length: 167
5 Host: coffee-portal-api.climawan.com
6 Connection: keep-alive
7 Accept-Encoding: gzip, deflate, br
8 User-Agent: okhttp/3.14.9
```

Dampak

Berdasarkan hasil analisis dan uji coba yang kami lakukan, temuan ini memberikan dampak serius terhadap integritas dan kerahasiaan sistem backend pada aplikasi coffee-portal. Dengan API_KEY yang ditulis secara hardcoded dalam aplikasi, attacker dapat melakukan interaksi langsung dengan seluruh endpoint API tanpa melalui proses autentikasi yang seharusnya dilakukan oleh user yang sah. Hal ini memungkinkan berbagai jenis eksploitasi seperti: akses tidak sah terhadap data pengguna lain, modifikasi data, penyalahgunaan fungsi aplikasi dari sisi client, hingga pengambilalihan penuh atas layanan backend.

Rekomendasi

Kami merekomendasikan agar aplikasi coffee-portal:

- Menghapus seluruh API key yang ditanam langsung dalam kode aplikasi (hardcoded) dan menggantinya dengan mekanisme otentikasi yang dinamis, seperti token berbasis OAuth 2.0 atau sistem otorisasi berbasis session dari sisi server.
- Menyimpan kredensial dan konfigurasi sensitif hanya di lingkungan server-side atau melalui penggunaan secure vault environment (seperti AWS Secrets Manager, Azure Key Vault, atau Google Secret Manager).
- Melakukan rotasi API key secara berkala dan mencatat seluruh aktivitas akses API dalam sistem logging yang aman dan terpusat untuk kebutuhan audit dan deteksi anomali.
- Menambahkan rate limiting dan validasi asal request (misalnya melalui IP whitelisting atau signed request) sebagai kontrol tambahan terhadap potensi penyalahgunaan API key yang bocor.

Temuan 6 – **Account Takeover**

Target: Coffee Portal Apps
Risiko: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N **9.1 (Critical)**

Deskripsi

Untuk kerentanan ini dikarenakan kepanjangan untuk step-by-step untuk melakukan attack ini, jadi kami membuat sebuah video untuk step yang kami lakukan untuk melakukan eksploitasi ini.
link PoC:
https://drive.google.com/file/d/1txrAcCZOhDd0lpZiRnJa7hGwE1JCGw7U/view?usp=sharing

Disini kami memanfaatkan cara dari coffe portal ini menyimpan informasi terhadap user yang sudah melakukan login, yaitu dengan menyimpannya di folder “shared_preferences”, dengan nama file “sp_coffee_portal.xml”, yang dimana menyimpan userid, dan juga email dari user.

```
emu64xa:/data/data/com.climawan.comp6844001_firsthalfproject.coffeeportal/shared_prefs # ls
sp_coffee_portal.xml
at sp_coffee_portal.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="logged_in_id">dabbb1bf-3392-46d6-bfe4-536269e9ca50</string>
    <string name="logged_in_email">asep@vi.com</string>
</map>
emu64xa:/data/data/com.climawan.comp6844001_firsthalfproject.coffeeportal/shared_prefs # ls
emu64xa:/data/data/com.climawan.comp6844001_firsthalfproject.coffeeportal/shared_prefs # cd ..
emu64xa:/data/data/com.climawan.comp6844001_firsthalfproject.coffeeportal # ls
cache  shared_prefs
emu64xa:/data/data/com.climawan.comp6844001_firsthalfproject.coffeeportal # cd shared_prefs/
emu64xa:/data/data/com.climawan.comp6844001_firsthalfproject.coffeeportal/shared_prefs # ls
emu64xa:/data/data/com.climawan.comp6844001_firsthalfproject.coffeeportal/shared_prefs # vi sp_coffee_portal.xml
emu64xa:/data/data/com.climawan.comp6844001_firsthalfproject.coffeeportal/shared_prefs #
```

Disini dengan kita mengganti isi dari shared preferences yang ada, kita bisa masuk sebagai user yang lain, tanpa perlu untuk login sebagai user tersebut, dan itu berarti password itu tidak diperlukan untuk seseorang melakukan login dikarenakan adanya kerentanan ini.

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="logged_in_id">c4202797-96b8-4553-9f43-83cd41af2617</string>
    <string name="logged_in_email">budi@vi.com</string>
</map>
~
```

Untuk user id sendiri bisa kita lakukan scraping data, dengan memanfaatkan vuln di api “/v1/order-transaction”, dan untuk email kita bisa saja melakukan social engineering untuk mendapatkannya, seperti misal seseorang baru memesan kopi dari aplikasi ini, dan masuk ke dalam order-transaction, kita bisa saja membuat link survey palsu yang meminta email yang dipakai untuk coffee portal ini, sehingga kita bisa mendapatkan account orang tersebut.

Dampak

Berdasarkan temuan yang kami eksplorasi dan uji, kerentanan ini memiliki dampak yang sangat kritikal terhadap keamanan akun pengguna pada aplikasi Coffee Portal. Dengan memanfaatkan file shared_preferences/sp_coffee_portal.xml yang menyimpan data sensitif seperti user_id dan email tanpa adanya mekanisme validasi atau enkripsi tambahan, penyerang dapat melakukan manipulasi langsung terhadap isi file tersebut dan mengakses akun pengguna lain tanpa memerlukan autentikasi ulang.

Rekomendasi

Kami merekomendasikan agar aplikasi Coffee Portal:

- Hindari penyimpanan data sensitif secara langsung di shared_preferences, khususnya user_id dan email, terutama jika tanpa enkripsi.
- Gunakan token autentikasi yang valid dan aman, yang dikelola dari sisi server untuk memastikan keaslian sesi pengguna.
- Simpan token atau data penting menggunakan Android Keystore agar tidak dapat dimodifikasi secara lokal oleh pengguna.
- Pastikan seluruh proses login dan akses data tervalidasi oleh server, dan tidak mengandalkan data lokal untuk menentukan identitas pengguna.