# DEMO CORP
# Security Assessment Findings Report

## Business Confidential

*Date: May 8th, 2024*
*Project: DC-001*
*Version 1.0*

# Table of Contents

# Confidentiality Statement

This document is the exclusive property of Demo Corp and TCM Security (TCMS). This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both Demo Corp and TCMS.

Demo Corp may share this document with auditors under non-disclosure agreements to demonstrate penetration test requirement compliance.

# Disclaimer

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.

Time-limited engagements do not allow for a full evaluation of all security controls. TCMS prioritized the assessment to identify the weakest security controls an attacker would exploit. TCMS recommends conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

# Contact Information

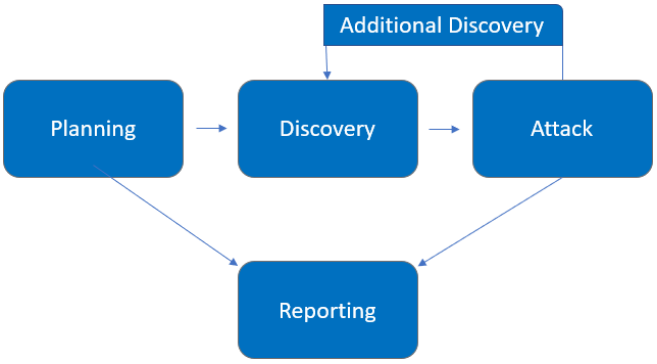
| Name | Title | Contact Information |
|---|---|---|
| Demo Corp | | |
| Agas Ananta Wijaya | Praktikum 3 Ethical Hacking | Email: agasananta04@gmail.com |
| TCM Security | | |
| Aslab Teknologi Informasi ITS | SQL Injection | Email: - |

# Assessment Overview

Kita para praktikan praktikum ethical hacking Teknologi Informasi ditugaskan melakukan penetration testing terhadap aplikasi mockup bank yang masih dalam tahap development, yang disebut Jay's Bank. Tujuan dari praktikum ini adalah untuk menemukan kerentanan yang mungkin ada dalam aplikasi dan melaporkannya untuk perbaikan sebelum aplikasi diluncurkan ke publik.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.




# Assessment Components

## Internal Penetration Test

An internal penetration test emulates the role of an attacker from inside the network. An engineer will scan the network to identify potential host vulnerabilities and perform common and advanced internal network attacks, such as: LLMNR/NBT-NS poisoning and other man- in-the-middle attacks, token impersonation, kerberoasting, pass-the-hash, golden ticket, and more. The engineer will seek to gain access to hosts through lateral movement, compromise domain user and admin accounts, and exfiltrate sensitive data.

# Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

| Severity | CVSS V3 Score Range | Definition |
|---|---|---|
| Critical | 9.0-10.0 | Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately. |
| High | 7.0-8.9 | Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible. |
| Moderate | 4.0-6.9 | Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved. |
| Low | 0.1-3.9 | Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window. |
| Informational | N/A | No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation. |

# Risk Factors

Risk is measured by two factors: Likelihood and Impact:

## Likelihood

Likelihood measures the potential of a vulnerability being exploited. Ratings are given based on the difficulty of the attack, the available tools, attacker skill level, and client environment.

## Impact

Impact measures the potential vulnerability's effect on operations, including confidentiality, integrity, and availability of client systems and/or data, reputational harm, and financial loss.

# Scope

| Assessment | Details |
|---|---|
| 1. Semua fungsi aplikasi.<br>2. Mekanisme akun pengguna dan autentikasi.<br>3. Antarmuka web dan API.<br>4. Interaksi database dan proses penanganan data. | IP Address Aplikasi: 167.172.75.216 |

## Scope Exclusions

Per client request, TCMS did not perform any of the following attacks during testing:

- Denial of Service (DoS)
- Phishing/Social Engineering

All other attacks not specified above were permitted by Demo Corp.

## Client Allowances

Demo Corp provided TCMS the following allowances:

- Internal access to network via dropbox and port allowances

# Executive Summary

Kita para praktikan praktikum ethical hacking Teknologi Informasi ditugaskan melakukan penetration testing terhadap aplikasi mockup bank yang masih dalam tahap development, yang disebut Jay's Bank. Tujuan dari praktikum ini adalah untuk menemukan kerentanan yang mungkin ada dalam aplikasi dan melaporkannya untuk perbaikan sebelum aplikasi diluncurkan ke publik.

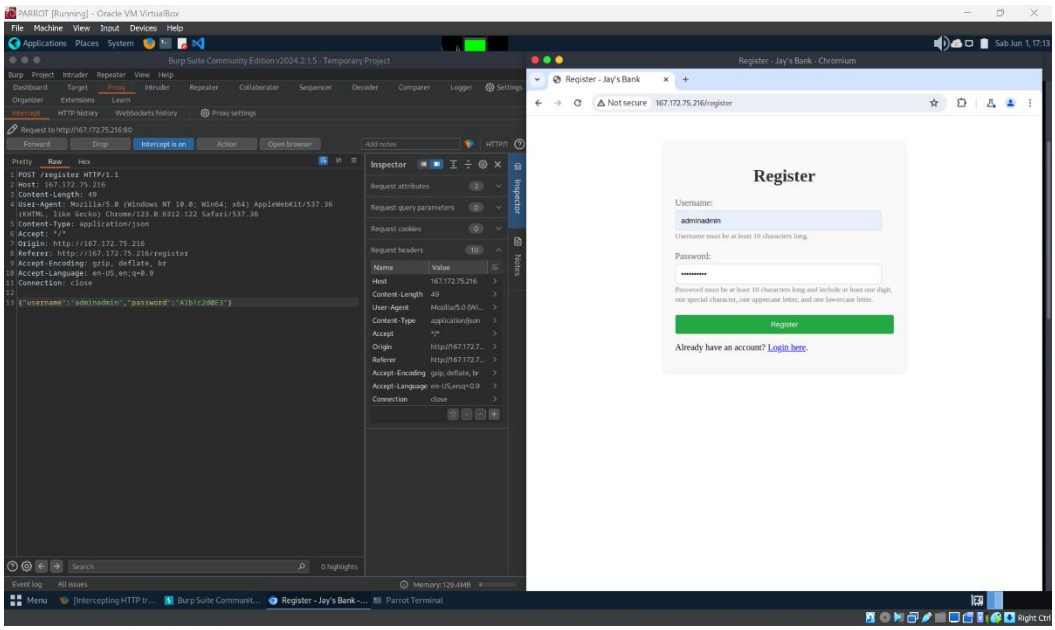
# Technical Findings

## Burpsuite Intercept

Mendapatkan POST menggunakan Burpsuite

| | |
|---|---|
| Description: | Intercept menggunakan Burp Suite adalah salah satu fitur inti yang memungkinkan pengguna untuk menangkap, memodifikasi, dan menganalisis lalu lintas HTTP(S) antara browser dan server web. Ini sangat berguna untuk pengujian keamanan aplikasi web karena memungkinkan penguji untuk melihat dan mengubah permintaan dan respons sebelum mencapai server atau browser. |
| Risk: | Menjalankan Burp Suite sebagai proxy dapat menambah overhead pada kinerja jaringan dan server. Hal ini bisa memperlambat waktu respons aplikasi web dan mempengaruhi pengguna lain. Memodifikasi permintaan dan respons HTTP secara tidak tepat dapat menyebabkan aplikasi web berfungsi tidak semestinya atau bahkan rusak. Hal ini bisa mengganggu layanan yang sedang berjalan, terutama jika dilakukan pada lingkungan produksi. |
| System: | All |
| Tools Used: | Burpsuite |
| References: | Stern Security - Local Network Attacks: LLMNR and NBT-NS Poisoning<br>NIST SP800-53 r4 IA-3 - Device Identification and Authentication<br>NIST SP800-53 r4 CM-6(1) - Configuration Settings |

Evidence



Selanjutnya kita akan mengcopy isi RAW dari yang tertera di Burpsuite lalu menuliskannya dalam txt.

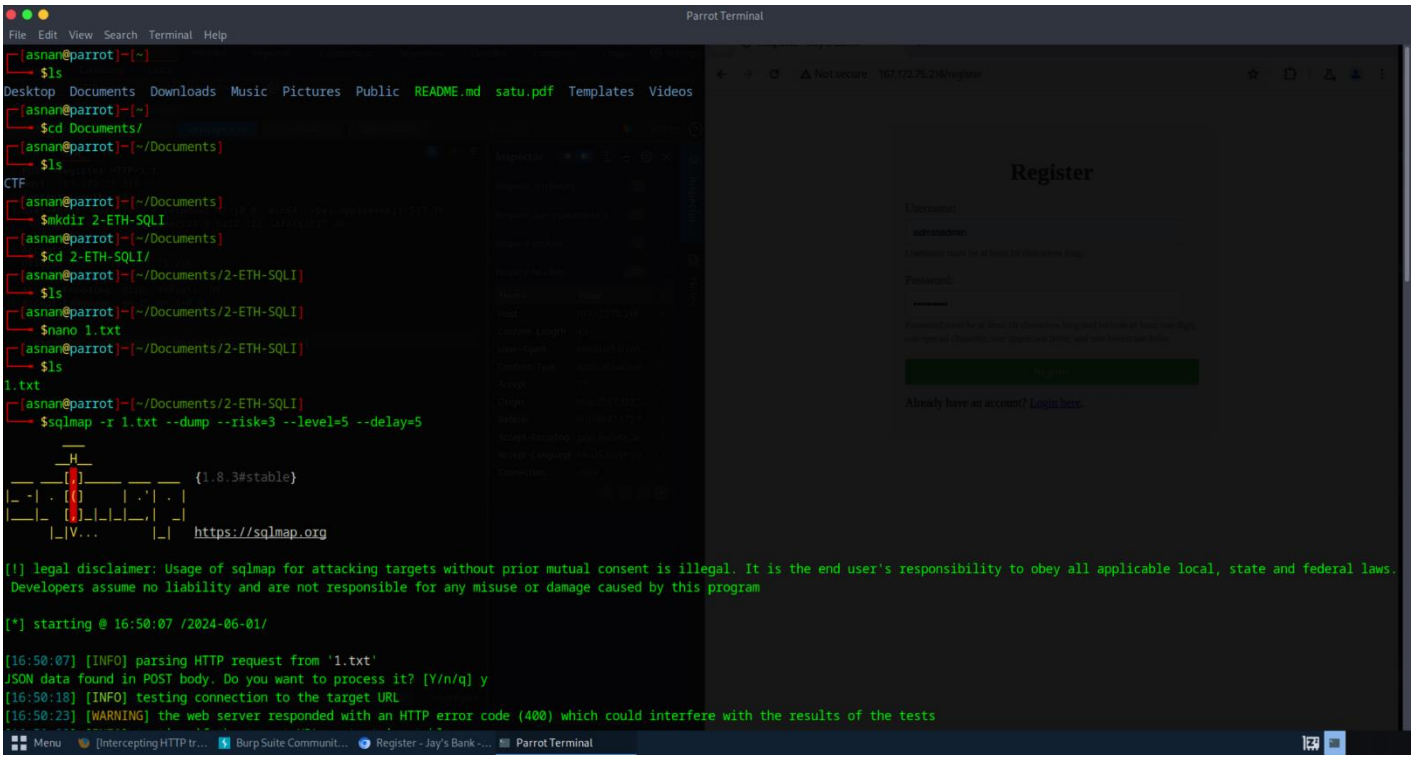
## Membuat file 1.txt

```
POST /register HTTP/1.1
Host: 167.172.75.216
Content-Length: 49
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/123.0.6312.122 Safari/537.36
Content-Type: application/json
Accept: */*
Origin: http://167.172.75.216
Referer: http://167.172.75.216/register
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close
{"username":"adminadmin","password":"A1b!c2d@E3"}
```

Melakukan SQLMAP: `sqlmap -r 1.txt --dump --risk=3 --level=5 --delay=5`

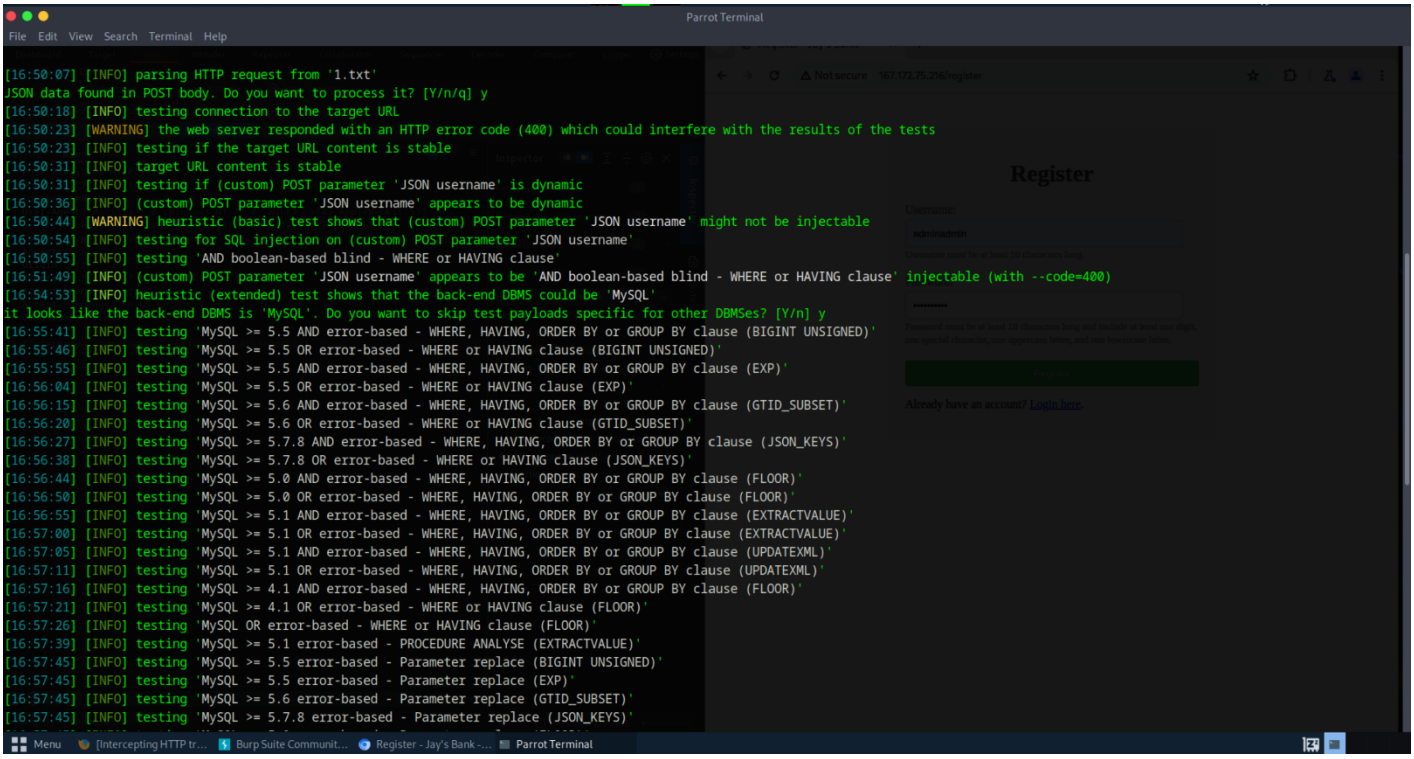
| | |
|---|---|
| Description: | Perintah ini menjalankan sqlmap untuk menguji endpoint `/register` pada server `167.172.75.216` dengan data dari `1.txt` untuk kemungkinan adanya kerentanan injeksi SQL. Dengan pengaturan risiko tertinggi (`--risk=3`), kedalaman pengujian tertinggi (`--level=5`), dan jeda 5 detik antara setiap permintaan (`--delay=5`), sqlmap akan melakukan pengujian yang sangat komprehensif dan agresif, serta mencoba mengambil data dari basis data jika injeksi SQL berhasil ditemukan. |
| Risk: | Menggunakan opsi `--dump` untuk mengambil data dari basis data dapat menyebabkan beban yang signifikan pada basis data, yang bisa mengakibatkan penurunan kinerja atau bahkan crash pada server database. Permintaan yang terus menerus dan intensif dengan delay hanya 5 detik dapat menyebabkan penurunan kinerja pada aplikasi web, terutama jika server tidak cukup kuat untuk menangani beban tersebut. |
| System: | All |
| Tools Used: | sqlmap |
| References: | https://capec.mitre.org/data/definitions/644.html<br>https://tcm-sec.com/pentest-tales-001-you-spent-how-much-on-security/ |

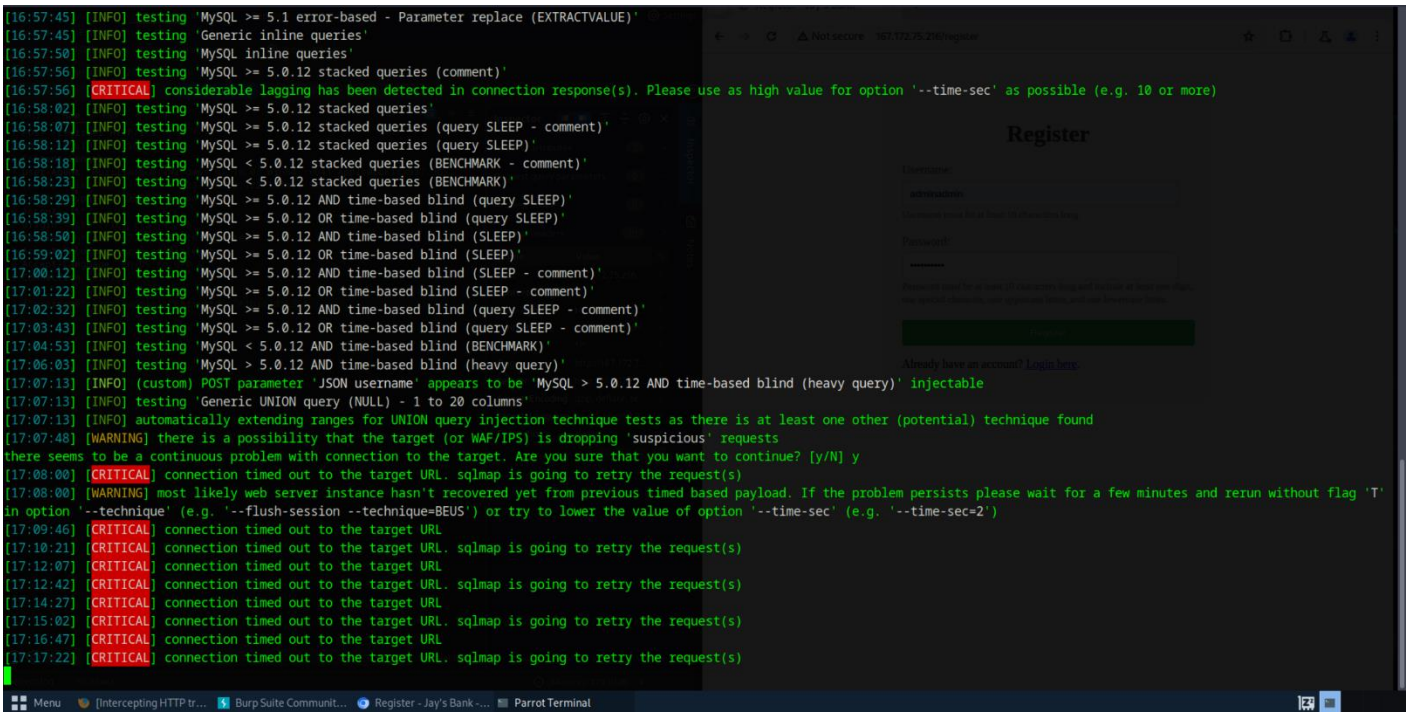
Evidence



Selanjutnya kita akan menunggu sqlmap dalam melakukan scanning. Namun sayangnya, beberapa kali saya mengalami connection timed out, sehingga menghambat dalam melakukan sql injection.

Evidence



## Security Strengths:

☐ Stabilitas Konten URL:
- Konten URL tampaknya stabil, yang berarti bahwa beberapa mekanisme pengamanan dapat bekerja untuk menjaga kestabilan aplikasi.

☐ Beberapa Tes Tidak Berhasil:
- Beberapa teknik injeksi tidak berhasil, yang menunjukkan adanya tingkat perlindungan atau konfigurasi yang mungkin telah mengurangi beberapa jenis serangan.

☐ Deteksi Lag dan Time-based Tests:
- Aplikasi mengalami lag ketika menggunakan payload berbasis waktu, yang menunjukkan bahwa ada beberapa mekanisme yang mempengaruhi kinerja ketika permintaan tidak wajar diajukan. Ini bisa menunjukkan perlindungan terhadap serangan DoS atau rate limiting.

.

## Kerentanan:

1. **SQL Injection pada Parameter 'username':**
   - Hasil menunjukkan bahwa parameter `username` pada `JSON` POST request rentan terhadap SQL Injection.
   - Sqlmap menemukan bahwa parameter `username` bisa di-exploit dengan teknik 'boolean-based blind' dan 'time-based blind' injection.
   - Meski parameter `username` ini tampaknya tidak dapat dieksploitasi lebih lanjut untuk 'UNION-based' injections, kemungkinan ini masih ada.
2. **HTTP 400 Error:**
   - Beberapa tes menunjukkan bahwa server memberikan respon kode error HTTP 400 (Bad Request). Ini menunjukkan bahwa server mungkin memiliki beberapa perlindungan atau validasi input yang tidak sempurna yang menyebabkan permintaan tertentu ditolak.

## Saran:

1. **Validasi Input yang Ketat:**
   - Terapkan validasi input yang ketat di sisi server. Pastikan semua input pengguna di-escape atau di-sanitasi sebelum digunakan dalam query SQL.
   - Gunakan prepared statements dan parameterized queries untuk menghindari injeksi SQL.
2. **Penggunaan WAF (Web Application Firewall):**
   - Implementasikan WAF untuk memantau dan memblokir permintaan mencurigakan atau berpotensi berbahaya.
   - WAF dapat membantu mendeteksi dan menghalangi serangan berbasis injeksi secara real-time.
3. **Pemeriksaan Error Handling:**

- Tingkatkan error handling agar tidak mengungkapkan informasi yang berlebihan. Sebagai contoh, kembalikan pesan kesalahan umum seperti "Invalid request" tanpa memberikan detail yang dapat digunakan oleh penyerang.

4. **Rate Limiting dan Throttling:**
   - Terapkan rate limiting untuk menghindari serangan brute force atau DoS. Batasi jumlah permintaan yang dapat dilakukan oleh satu pengguna dalam jangka waktu tertentu.
5. **Regular Security Audits:**
   - Lakukan audit keamanan secara berkala. Gunakan alat analisis kerentanan dan melakukan pengujian penetrasi untuk menemukan dan memperbaiki kerentanan baru.
6. **Pembaruan dan Patching Rutin:**
   - Pastikan semua perangkat lunak, termasuk framework dan database, selalu diperbarui dengan patch keamanan terbaru.
7. **Log dan Monitoring:**
   - Aktifkan logging untuk semua aktivitas mencurigakan dan pastikan ada sistem pemantauan yang dapat memberi tahu administrator tentang potensi serangan atau anomali.

---


Last Page