

# LAPORAN RESMI KERENTANAN KEAMANAN (SQL INJECTION)

## 1. Ringkasan Eksekutif

Laporan ini mendokumentasikan temuan kerentanan SQL Injection (SQLi) pada salah satu endpoint milik Kampus. Kerentanan ini memungkinkan penyerang memodifikasi query database, mengakses data sensitif, atau melakukan tindakan tidak sah lainnya. Temuan dikategorikan High Severity karena berdampak pada confidentiality, integrity, dan availability.

## 2. Informasi Temuan

Jenis: SQL Injection

Kategori: OWASP Top 10 – A03 Injection

Severity: High

Tanggal Temuan: 1 Desember 2025

Reporter: Andi Fitra Furqan,S.T; CEH,ECIH / AFIF

## 3. Lokasi Kerentanan

Endpoint:

<https://ujimaterial.polman-bandung.ac.id/>

Parameter Rentan: id

## 4. Langkah Reproduksi (Proof of Concept)

4.1 Error-Based Test

Payload: ' OR '1' = '1'

Respon: True ( Berhasil Login)

## 5. Bukti Pendukung

The screenshot shows the 'Applikasi Pengujian Polman' dashboard. On the left is a sidebar with navigation links: 'MASTER' (Jenis Usaha, Pengujian, Pelanggan), 'TRANSAKSI' (Order Pengujian, Invoice), 'LAPORAN' (Order Pengujian), and 'USERS' (Ubah Password, Logout). The main area displays a blue banner with the text 'Selamat Datang di Aplikasi Pengujian'. At the bottom left of the sidebar, it says 'Developed © 2016 - UPT PUSKOMEDIA'.

The screenshot shows a browser window with a 'Login' form for 'ujimaterial.polman-bandung.ac.id'. The 'Username' field contains the value "' OR '1'='1 -- ". The 'Network' tab in the developer tools shows a list of requests, with the 'index.php' file having a status of 200 and a duration of 19 ms. Other files listed include 'reset.css', 'structure.css', 'jquery.min.js', 'jquery-1.5.2.min.js', 'logo2.png', 'logo\_blu\_speed.png', 'favicon.ico', and 'user\_cek\_login.php'.

## 6. Analisis Dampak

- Potensi bocornya data sensitif (PII, data pelanggan, Mahasiswa)
- Potensi bypass authentication

- Potensi modifikasi atau penghapusan data
- Potensi RCE pada konfigurasi tertentu
- Potensi gangguan layanan

## **7. Root Cause**

- Input tidak divalidasi
- Query menggunakan dynamic SQL (concatenation)
- Tidak ada prepared statements
- Error database terlihat secara publik

## **8. Rekomendasi Mitigasi**

Teknis:

1. Gunakan prepared statements/parameterized queries.
2. Terapkan input validation dengan allowlist.
3. Sembunyikan error database dari publik.
4. Aktifkan WAF rule untuk SQLi.

Operasional:

1. Lakukan secure code review.
2. Implementasi DevSecOps.
3. Lakukan penetration testing berkala.
4. Monitoring SIEM untuk pola SQL Injection.

## **9. Penutup**

Kerentanan ini perlu segera ditangani karena berdampak signifikan pada keamanan data dan operasional. Laporan ini disampaikan dalam semangat Responsible Disclosure untuk peningkatan keamanan sistem.