SQL Injection 1 : Bypass password saat login

Kenapa berhasil?

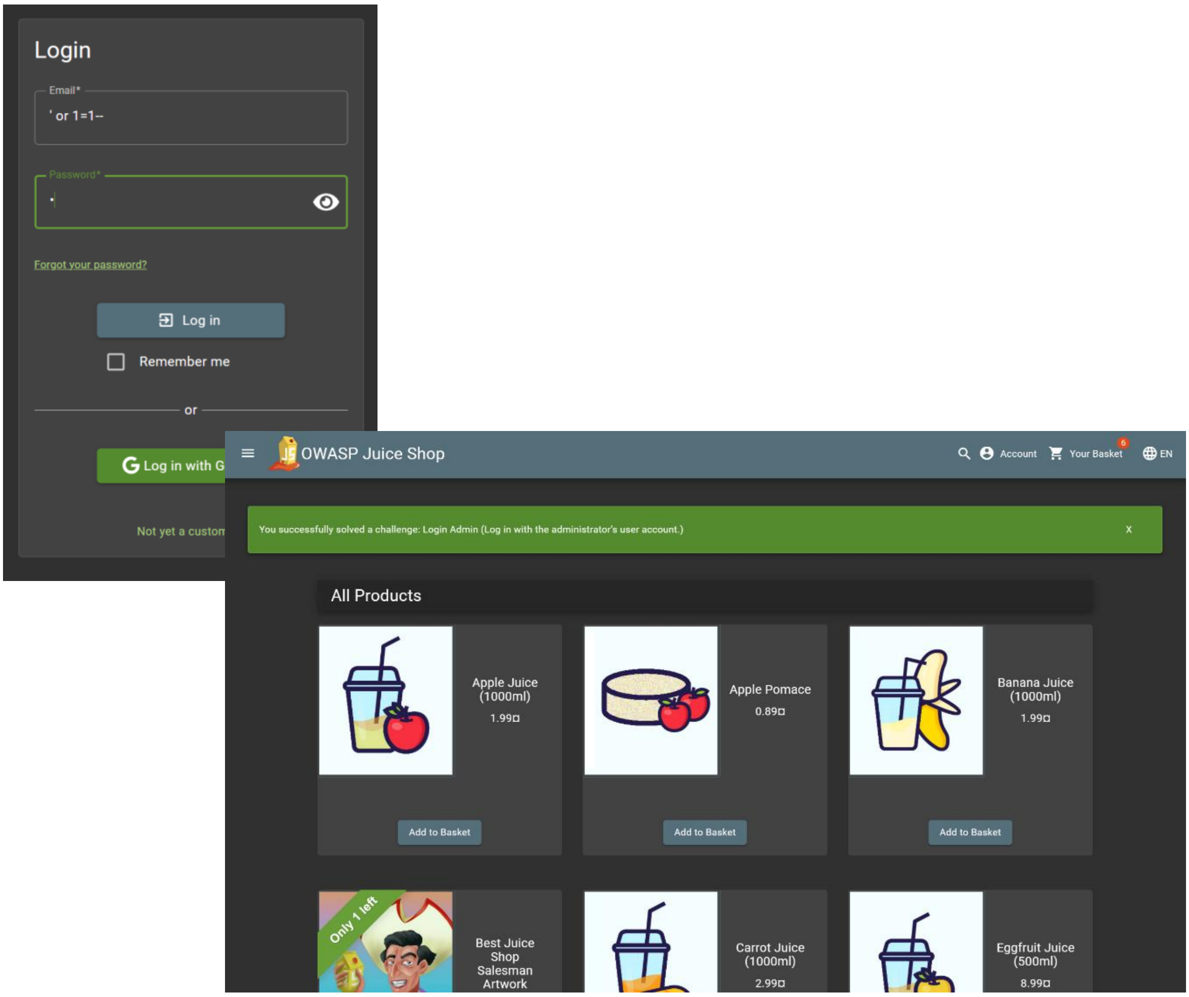
1. Data inputan tidak disanitasi.
2. Tidak ada validasi format email
3. ‘ mengeluarkan inputan dari parameter dan menambahkan kondisi baru menggunakan OR
4. 1=1 = true
5. ‘--‘ menyebabkan kondisi where di belakangnya menjadi comment, memungkinkan pemeriksaan password diabaikan.

Risiko dampak

1. Akses user lain secara illegal
2. Pencurian data

Rekomendasi perbaikan

1. Sanitasi data inputan.
2. Gunakan validasi format email.
3. Gunakan mekanisme login yang lebih aman.



Login

Email*: ' or 1=1--

Password*: •

You successfully solved a challenge: Login Admin (Log in with the administrator's user account.)

SQL Injection 2 : Menambahkan karakter yang menyebabkan query error

Kenapa berhasil?

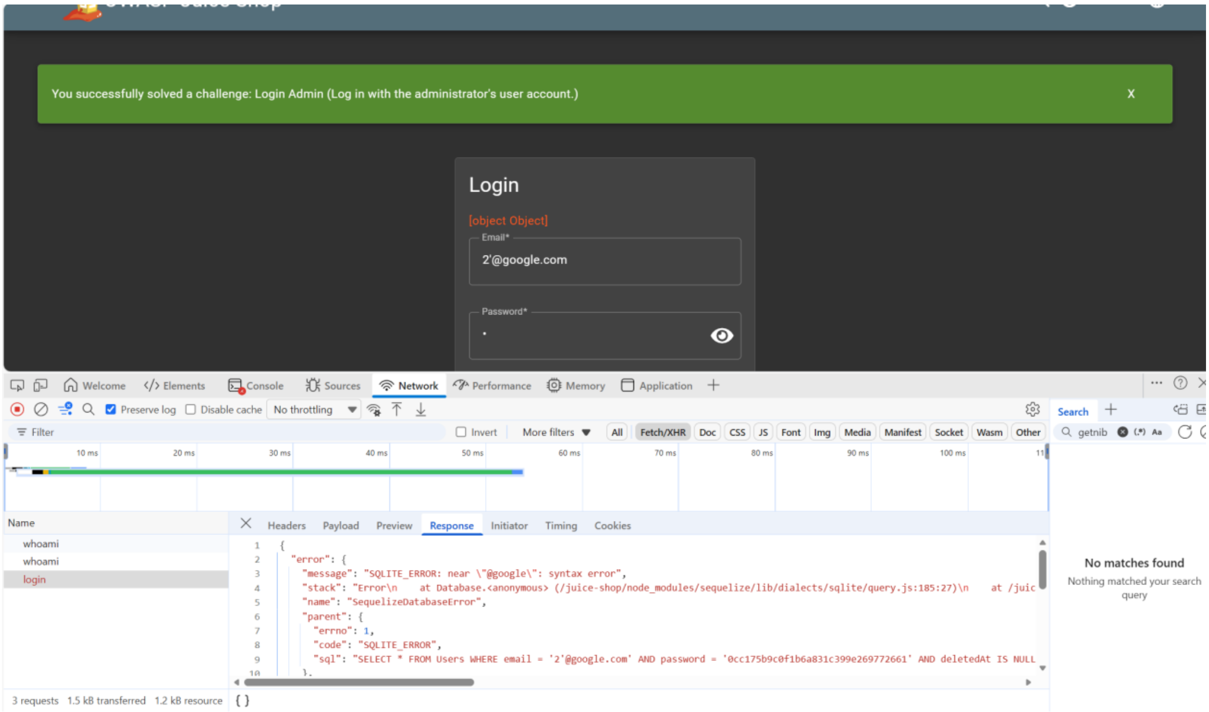
1. Data inputan tidak disanitasi.
2. Tidak ada validasi format email.
3. ‘ menyebabkan error pada query yang tidak dihandle dengan benar.

Risiko dampak

1. Bila error handling kurang baik, bisa jadi pesan error mengekspose kerentanan lebih jauh

Rekomendasi perbaikan

1. Sanitasi data inputan.
2. Gunakan validasi format email.
3. Perbaiki error handling agar tidak mengekspos informasi terlalu banyak.



You successfully solved a challenge: Login Admin (Log in with the administrator's user account.)

Login

[object Object]

Email*: 2'@google.com

Password*: ·

```
{
  "error": {
    "message": "SQLITE_ERROR: near \"@google\": syntax error",
    "stack": "Error\n    at Database.<anonymous> (/juice-shop/node_modules/sequelize/lib/dialects/sqlite/query.js:185:27)\n    at /juic
    "name": "SequelizeDatabaseError",
    "parent": {
      "errno": 1,
      "code": "SQLITE_ERROR",
      "sql": "SELECT * FROM Users WHERE email = '2'@google.com' AND password = '0cc175b9c0f1b6a831c399e269772661' AND deletedAt IS NULL
    },
```