NicolasPauferro / **sqli-ecclesia.md** Secret

Created last week

Code Revisions 1

SQL Injection in EcclesiaCRM <8.0

`sqli-ecclesia.md`

VULNERABILITY REPORT: Authenticated SQL Injection in EcclesiaCRM <8.x

Executive Summary

- **Vulnerability:** SQL Injection (SQLi) via Parameterized Query Template Substitution
- Product: EcclesiaCRM (https://github.com/phili67/ecclesiacrm)
- Affected Version: <v8.0.0
- CWE: CWE-89 (Improper Neutralization of Special Elements used in an SQL Command)
- CVSS 3.1 Score: 8.8 (High) — `CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H`
- Prerequisite: Authenticated user with access to the "Query Viewer" component.
- Impact: Full Database Exfiltration, Administrative Credential Theft, and Unauthorized Data Access.

1. Vulnerability Description

EcclesiaCRM is vulnerable to a critical SQL Injection in its **Query Viewer** component. The application allows users to execute pre-defined queries with custom parameters. However, it fails to properly sanitize these user-provided parameters before inserting them into SQL query templates using string substitution.

This flaw allows an authenticated attacker to inject arbitrary SQL commands, bypassing intended query logic to extract sensitive information from any table in the database.

---

2. Affected Components

- **Endpoint:** `/v2/query/view/{id}`
- **File:** `src/v2/templates/query/queryview.php`
- **Functions:** `ValidateInput()` and `ProcessSQL()`
- **Secondary Issue:** Information Disclosure (Full SQL query leakage within HTML comments).

---

3. Technical Analysis & Root Cause

The vulnerability resides in the workflow used to process parameterized queries. When a user runs a pre-defined query (e.g., Query ID 200 - Custom Search), the application accepts parameters via POST (e.g., `~value~` or `~custom~`).

Root Cause Analysis:

1. **Ineffective Validation**: In `src/v2/templates/query/queryview.php`, the `ValidateInput` function contains a `default` case that accepts raw POST data without filtering or escaping:

```
78: default:
79:     $vPOST[$qrp_Alias] = $POST[$qrp_Alias];
80:     break;
```

2. **Template Substitution**: The `ProcessSQL` function then uses `str_replace` to merge this raw input directly into the SQL query template:

```
103: $qry_SQL = str_replace('~' . $qrp_Alias . '~', $vPOST[$qrp_Alias], $
```

3. **Execution**: The resulting unescaped SQL string is executed via `mysqli_query()` through the `MiscUtils::RunQuery()` helper.

Information Disclosure:

The application explicitly leaks the full constructed SQL query in HTML comments at line 100 of `queryview.php`:

```
100: <?= "--" . $qry_SQL ?>
```

4. Proof of Concept (PoC)

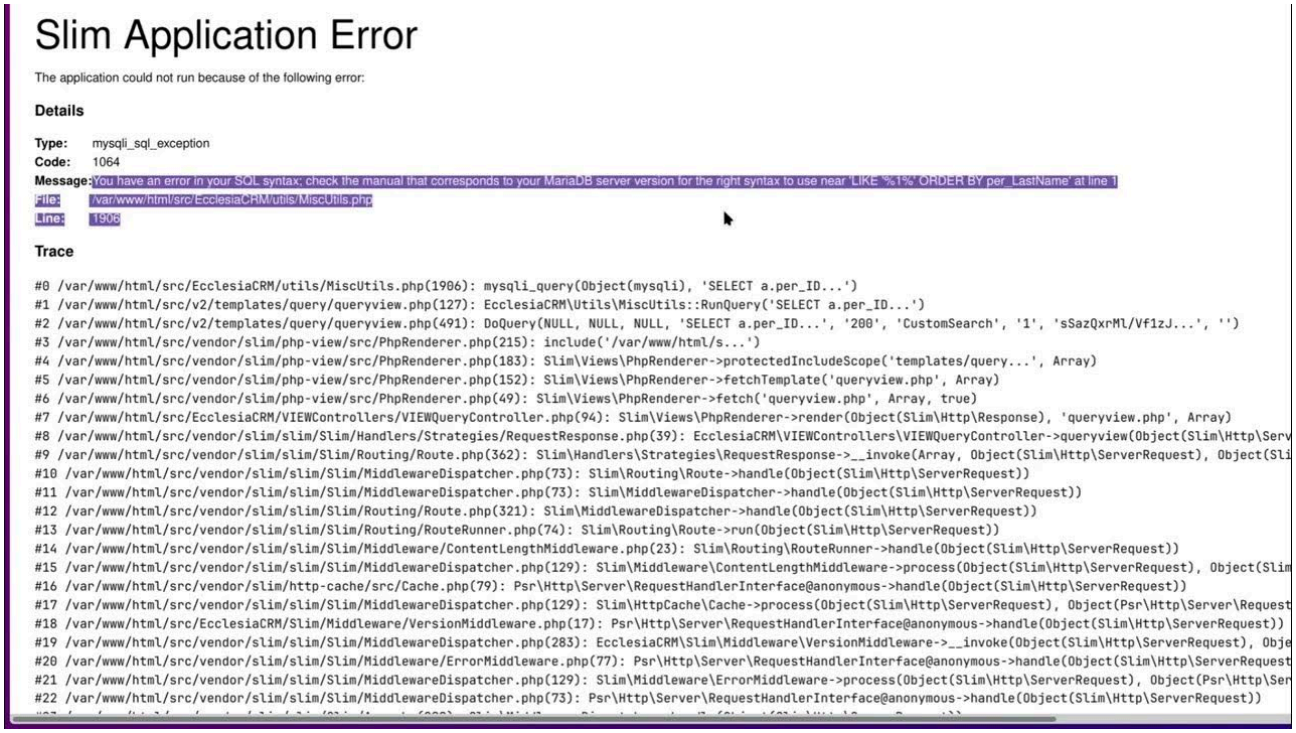
An attacker can use the `custom` parameter to perform a `UNION`-based injection to extract usernames and password hashes from the `user_usr` table.

Request:

```
POST /v2/query/view/200 HTTP/1.1
Host: [TARGET_HOST]
Content-Type: application/x-www-form-urlencoded
Cookie: [AUTH_COOKIES]

custom=per_ID AND 1=0 UNION SELECT 1, CONCAT(usr_UserName, ':', usr_Password)
```

4.5 Video Poc



5. Impact Assessment

Confidentiality: High. Attackers can access all database tables, including member personal information, financial records, and pastoral notes. Integrity: High. Depending on the database user permissions, an attacker might be able to modify or delete records. Availability: Low/Medium. Risk of database disruption through heavy queries or data deletion.

6. Recommended Remediation

- Implement Prepared Statements: Transition from manual string substitution to Parameterized Queries (Prepared Statements) using PDO or MySQLi. This is the only definitive fix for SQL Injection.
- Strict Input Validation: Update ValidateInput() to sanitize all inputs using mysqli_real_escape_string() or type-casting (e.g., (int)) as a temporary mitigation.
- Disable Verbose Debugging: Remove the code that echoes $qry_SQL into HTML comments to prevent sensitive information disclosure.]

7. Researcher Information

- Name: Nicolas Pauferro
- Discovery Date: 2026-03-27
- Disclosure Status: Reported to Vendor via e-mail (and vuln was corrected in 29/03/2026 commit)