SQL Injection via Unvalidated sort_by Query Parameter in Multiple Repository Classes

High **daveearley** published **GHSA-2qcp-24fh-fx6p** last week

Package

php **hi-events/hi-events** (Composer)

Affected versions

>=0.8.x

Patched versions

>=v1.7.1-beta

Description

Summary

Multiple repository classes pass the user-supplied `sort_by` query parameter directly to Eloquent's `orderBy()` without validation, enabling SQL injection. The application uses PostgreSQL which supports stacked queries.

Root Cause

10+ repository files including `AttendeeRepository.php`, `EventRepository.php`, `OrderRepository.php` pass `$sortBy` directly to `orderBy()`. `AttendeeRepository` is worst-case — it concatenates: `'attendees.' . $sortBy`.

An `getAllowedSorts()` whitelist exists and IS used correctly in the admin endpoint (`getAllEventsForAdmin()`), proving awareness of the issue — but it is not applied in non-admin repositories.

```
// AttendeeRepository.php — no validation
$query->orderBy('attendees.' . $sortBy, $params->sort_direction);

// Correct pattern (admin endpoint):
if (in_array($params->sort_by, DomainObject::getAllowedSorts()->keys(), true)) {
```

```
    $query->orderBy($params->sort_by, ...);
}
```

Proof of Concept

```
# Authenticated organizer — time-based blind SQL injection
GET /api/events/{id}/attendees?sort_by=id;SELECT+pg_sleep(5)-- HTTP/1.1
Authorization: Bearer <organizer_token>
# Response delayed ~5 seconds confirms injection
```

Impact

- Data exfiltration via stacked queries (PostgreSQL)
- Attendee PII, payment data, promo codes accessible
- Requires authenticated organizer role

Remediation

Apply the existing `getAllowedSorts()` whitelist in all repository classes, matching the pattern already used in `getAllEventsForAdmin()`.

Credit

Discovered by **@tikket1**, 2026-03-28.

Severity

High

CVE ID

CVE-2026-34455

Weaknesses

▶ CWE-89

Credits

tikket1 Reporter