

Security report: Requests Portal

Illustrative example: the organisation, the application and the flaws are fictional and do not correspond to any client.

ORGANISATION	Example Organisation (fictional)
APPLICATION	Requests Portal, version 3.2
SCOPE	Web application, API and source code
APPROACH	Grey box, with test accounts and access to the code
METHODOLOGIES	OWASP WSTG, OWASP ASVS and CVSS 4.0
STATUS	Retest completed

Executive summary

Four flaws were found: one critical, one high, one medium and one low. The most serious allowed anyone, without signing in, to reach the database through the public search. Three flaws were fixed and confirmed in the retest; the low-severity one is planned for the next release.

We recommend keeping code review in new releases and monitoring the application's dependencies, so that known flaws are fixed as soon as they are published.

Findings by severity

Critical		1
High		1
Medium		1
Low		1
Fixed and confirmed		3
Fix planned		1

Findings

Every finding was reproduced in a test environment before going into the report.

A-01 · SQL injection in the public search

Critical · 9.3

STATUS	Fixed and confirmed
LOCATION	search/results.php, line 42 · parameter "term"
CVSS 4.0	CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:N/SI:N/SA:N

Description

The search term is joined directly into the database query, without parameterisation. The search page is public, so no account is needed.

Evidence

In a test environment, a harmless value in the parameter changed the structure of the query and the order of the results, which confirms that the user's text is interpreted as part of the query. The full evidence was handed to the technical team and is not reproduced here.

Impact

An attacker could read, change or delete data in the database, including users' personal data, without any authentication.

Proposed fix

BEFORE

```
$sql = "SELECT id, subject FROM requests
        WHERE subject LIKE '%" . $_GET['term'] . "%'";
$rows = $db->query($sql)->fetchAll();
```

AFTER

```
$stmt = $db->prepare(
    'SELECT id, subject FROM requests WHERE subject LIKE ?'
);
$stmt->execute(['%' . $_GET['term'] . '%']);
$rows = $stmt->fetchAll();
```

Retest

Parameterised queries in every search. The retest confirmed that the parameter no longer changes the query.

A-02 · Access to other users' requests

High · 7.1

STATUS

Fixed and confirmed

LOCATION

API · GET /api/requests/{id}

CVSS 4.0

CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N

Description

The API returns a request from its identifier, but does not check whether the request belongs to the signed-in user.

Evidence

With test account A, requesting a record created by test account B returned that record's data (redacted in the report).

Impact

Any user with an account could look up other users' requests, including personal data and attached documents.

Proposed fix

BEFORE

```
$request = Request::find($id);
return response()->json($request);
```

AFTER

```
$request = Request::where('id', $id)
    ->where('user_id', $user->id)
    ->firstOrFail();
return response()->json($request);
```

Retest

Ownership check applied to every request in the API. The retest confirmed that account A can no longer see account B's records.

A-03 · Text editor library with a known flaw

Medium · 5.3

STATUS	Fixed and confirmed
LOCATION	package.json · text editor library, version 4.1
CVSS 4.0	CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:P/VC:N/VI:N/VA:N/SC:L/SI:L/SA:N

Description

The application uses a version of a text editor library with a vulnerability published by its maintainers, which allows code to be included in messages. The vulnerability's identifier is omitted in this example.

Evidence

The version was identified by the dependency analysis and the behaviour described by the maintainers was confirmed in a test environment.

Impact

A crafted message could run code in the browser of whoever opened it, for example a member of staff in the back office.

Proposed fix**BEFORE**

```
"dependencies": {  
  "example-editor": "4.1.0"  
}
```

AFTER

```
"dependencies": {  
  "example-editor": "4.3.2"  
}
```

Retest

Library updated to the version fixed by its maintainers. The dependency is now monitored under the annual subscription.

A-04 · Session cookie without protective attributes

Low · 2.3

STATUS	Fix planned
LOCATION	Session configuration · config/session.php
CVSS 4.0	CVSS:4.0/AV:N/AC:H/AT:P/PR:N/UI:P/VC:L/VI:L/VA:N/SC:N/SI:N/SA:N

Description

The session cookie lacks the Secure and HttpOnly attributes, which prevent it from being sent over unencrypted connections and from being read by code in the browser.

Evidence

The headers of the sign-in response show the session cookie without either attribute.

Impact

Combined with another flaw, it would make stealing a session easier. On its own, the risk is low.

Proposed fix**BEFORE**

```
"secure" => false,  
"http_only" => false,
```

AFTER

```
"secure" => true,  
"http_only" => true,  
"same_site" => "lax",
```

Retest

Fix planned for the application's next release, with a new retest at that point.