Horizontal Privilege Escalation via IDOR & Identifier Leakage

Project Information

- **Analyst:** Mduduzi William Radebe
- **Date:** 07 January 2026
- **Platform:** PortSwigger Web Security Academy
- **Vulnerability Type:** CWE-639: Access Control Bypass Through User-Controlled Key (IDOR)
- **Severity: High (8.1/10)**

1. Executive Summary

This report documents a successful **Horizontal Privilege Escalation** attack. The application utilizes Globally Unique Identifiers (GUIDs) to reference user accounts; however, these identifiers are leaked in public profile pages. Due to a lack of server-side object-level authorization, I was able to use a leaked GUID to access a peer user's private account page and exfiltrate their sensitive API key.

2. Technical Analysis & Discovery

The "Security through Obscurity" Flaw

The application designers assumed that because a GUID (e.g., 8ecbd9ca...) is statistically impossible to guess or brute-force, the account pages were secure. This ignores the fact that identifiers are often **leaked** through normal application use.

Information Disclosure (Reconnaissance)

During a crawl of the public-facing blog, I inspected the metadata associated with user comments.

- **Observation:** Clicking a user's name leads to a "Public Profile."
- **Leakage:** The URL for the public profile discloses the user's "Internal ID."
- **Harvested Identifier (Carlos):** 8ecbd9ca-56d3-4f54-8de8-0c0b1f58b3c8

3. Exploitation Methodology (Proof of Concept)

Step 1: Baseline Establishing

I logged in with my standard credentials (wiener) and accessed my account page:

GET /my-account?id=f1ef206f-2d18-4a51-934a-f86e2ac855d6

The server correctly rendered my API key and profile details.

Step 2: IDOR Attempt (Horizontal Escalation)

I performed "Parameter Tampering" by replacing my own ID with the identifier harvested from the user carlos.

Target URL:

https://[LAB-ID].web-security-academy.net/my-account?id=8ecbd9ca-56d3-4f54-8de8-0c0b1f58b3c8

Step 3: Verification of Unauthorized Access

The server responded with a 200 OK and rendered the private dashboard for **Carlos**.

- **Data Exfiltrated:** API Key T34I0VxlEW6oNrxpAtr9CnTQmUWfFZ3j

4. Root Cause Analysis

Attack Component	Status	Security Risk
Identifier Complexity	**High**	GUIDs are used, preventing traditional "number-incrementing" attacks.
Identifier Confidentiality	**Failed**	Internal IDs are exposed in public URLs and metadata.
Authorization Check	**Broken**	The server checks if the user is *logged in*, but fails to check if the user *owns* the requested ID.

5. Security Impact & Business Risk

In a production environment, this vulnerability is a high-risk event:

- **GDPR/POPIA Violation:** Unauthorized access to Personal Identifiable Information (PII).
- **Account Takeover:** If the account page allows changing passwords or emails, IDOR leads directly to a full account takeover.
- **API Abuse:** Stolen API keys can be used to perform actions in the name of the victim, potentially leading to financial or reputational damage.

6. Remediation & Hardening Recommendations

A. Implement Object-Level Authorization (Primary)

The application must perform an **ownership check** on the server side for every request. The logic should be:

IF (Logged_In_User_ID == Requested_Object_Owner_ID) THEN { Grant Access } ELSE { Deny Access }

B. Use Indirect Reference Maps

Avoid exposing internal database IDs or GUIDs in the URL. Instead, use a temporary, per-session "Reference Map" that translates a random UI value to the actual database ID only for the duration of that user's session.

C. Secure Identifier Handling

Identifiers used for private actions should not be the same as identifiers used for public profile views.

7. Conclusion

This lab demonstrates that **Obfuscation (using GUIDs) is not a substitute for Authorization.** Despite the complexity of the identifiers, the lack of a simple ownership check on the backend allowed for the total compromise of a peer user's private data.

How to add this to your GitHub (IDOR_Unpredictable_Identifiers.md)

1. **In WSL:**
   cd ~/Labs/Network-Security-Analysis-Portfolio/Portswigger_Labs
2. **Create the file:**
   nano IDOR_Unpredictable_Identifiers.md
3. **Paste** the content above.
4. **Save and Push:**
   git add .
   git commit -m "Added IDOR lab writeup with GUID exploitation"
   git push origin main