[Lab Report] Broken Access Control: Unprotected Administrative Panel Discovery

Project Information

- **Analyst:** Mduduzi William Radebe
- **Date:** 07 January 2026
- **Target:** PortSwigger Web Security Academy (Apprentice Lab)
- **Vulnerability Type:** CWE-284: Improper Access Control / **OWASP A01:2021 – Broken Access Control**
- **Severity: Critical (9.1/10)**

1. Executive Summary

A critical security flaw was identified where the administrative interface of the web application was accessible to unauthenticated users. The sensitive endpoint was discovered via an insecure configuration in the robots.txt file. I successfully accessed the management console without credentials and performed an unauthorized administrative action (user deletion), demonstrating a total failure of the application's access control mechanisms.

2. Technical Analysis & Reconnaissance

The "Security through Obscurity" Fallacy

The application relied on the assumption that if an administrative URL is not linked on the homepage, it cannot be found. This is a common but dangerous security misconception.

Information Disclosure via robots.txt

Web crawlers use robots.txt to identify paths that should stay out of search results. However, attackers use it as a "treasure map" for sensitive directories.

Reconnaissance Command:

curl https://[lab-id].web-security-academy.net/robots.txt

Findings:

Plaintext
User-agent: *
Disallow: /administrator-panel

3. Exploitation Workflow (Proof of Concept)

Step 1: Endpoint Navigation

I navigated directly to the discovered path:

https://[lab-id].web-security-academy.net/administrator-panel

Step 2: Access Control Verification

The server responded with a 200 OK and rendered the full Administrative Dashboard. No session cookie, login prompt, or Multi-Factor Authentication (MFA) was requested.

Step 3: Unauthorized Action (Impact Verification)

To demonstrate the severity of the access, I targeted a specific user account (carlos) for deletion.

- **Action:** Clicked "Delete" next to the user profile.
- **Request Intercepted:** GET /administrator-panel/delete?username=carlos
- **Result:** Server executed the request, and a "User deleted successfully" message was displayed.

4. Detailed Findings

Risk Factor	Details
Vulnerable Endpoint	/administrator-panel
Root Cause	Missing server-side authorization checks.
Exploitation Ease	Trivial (Requires only a browser).
Impact	Full Administrative Compromise / Data Loss.

5. Security Impact & Business Risk

In a production environment, this vulnerability allows any internet user to:

- **Delete entire databases** or user registries (Denial of Service).
- **Exfiltrate User Data** (GDPR/POPIA compliance violation).
- **Privilege Escalation:** An attacker could promote their own account to Admin, gaining permanent control.

6. Remediation & Hardening Recommendations

A. Enforce Server-Side Authorization (Primary Defense)

The application must check if a user is authenticated **and** holds an 'Administrator' role before rendering the page or executing any logic on /administrator-panel.

JavaScript
// Example Secure Middleware (Node.js/Express)
app.use('/administrator-panel', ensureAuthenticated, ensureAdminRole);

B. Remove Sensitive Paths from robots.txt

Sensitive directories should never be listed in robots.txt. Instead, use server-side authentication to block crawlers and unauthorized users simultaneously.

C. Use Non-Predictable Administrative URLs

While not a primary defense, using complex, non-standard paths (e.g., /admin_a7b2_99/) can slow down automated scanners, though it must be paired with strong MFA.

7. Conclusion

The successful completion of this lab highlights the critical need for **Defense in Depth**. Hiding a sensitive page is not a substitute for securing it. By demonstrating the unauthorized deletion of a user, I have proven that the current architecture lacks the basic access controls required for a secure web application.