CYBERTECH CLUB @ PSU | CTF WORKSHOP SERIES

MODULE 02

Access Control

Participant Handout • Spring 2026

HOW TO USE THIS HANDOUT

This packet mirrors the live workshop deck so you can study async. Each section maps to a slide.

1. Hook

SET THE SCENE

If a website hides its admin page but never checks who is accessing it, anyone can become an admin just by changing the URL.

We are going to do exactly that, on a real lab, in under five minutes.

2. Learning Objectives

By the end of this topic, a student can:

- ☐ **Identify** access control weaknesses in a web application.
- ☐ **Bypass** access restrictions by manipulating URLs or requests.
- ☐ **Distinguish** between horizontal and vertical privilege escalation.

3. Core Concept

Access control

Access control determines what actions a user is allowed to perform. A vulnerability occurs when the application relies on client-side behavior (like hidden links or UI restrictions) instead of enforcing rules on the server. Even if a user is not shown an admin page, they might still reach it directly if the server skips the permission check.

Two flavors of escalation

Horizontal escalation means accessing another user's data at your own privilege level (for example, viewing /user/124 when you are user 123). Vertical escalation means gaining higher privileges, like reaching /admin as a regular user. Both come from the same root cause: missing server-side authorization.

The rule to remember

Security must be enforced on the server, not just hidden in the interface. If the only thing keeping a user out of a page is the absence of a link, the page is not protected.

FIGURE 1: BYPASSING ACCESS CONTROL BY EDITING THE URL

Users

- wiener [Delete]
- carlos [Delete]
- administrator [Delete]

Changing the URL to /admin grants access to restricted functionality because the server does not enforce permission checks. The interface never showed a link to this page, but the page itself is not protected.

4. How It Works

Four steps that match how a tester actually probes for broken access control.

STEP 01

Access the Application Normally

Use it like a regular user first.

- Log in or browse the app as a normal user.
- Note every page and feature the UI exposes to you.

STEP 02

Discover Hidden Endpoints

Look for paths the UI does not advertise.

- Try common admin paths: /admin, /dashboard, /manage, /settings.
- Manually edit the URL in the browser address bar.

STEP 03

Request the Restricted Resource

Go straight at the endpoint.

- Send a request to the modified URL (for example, /admin).
- Bypass the missing UI link by hitting the endpoint directly.

STEP 04

Exploit Missing Server Checks

Confirm the server never validated you.

- If the server fails to verify your role, the restricted content loads.
- Try a privileged action (delete, edit, promote) to confirm the bypass works end to end.

5. Demo Walkthrough

Run this end to end on the PortSwigger lab. The live demo follows the same script.

Prerequisites

LAB	BROWSER	STATE
Unprotected admin functionality	**Chrome or Edge**	**PortSwigger lab launched**

Lab URL: `https://portswigger.net/web-security/access-control/lab-unprotected-admin-functionality`

Step 1: Access the lab as a normal user

Action. Open the lab URL in the browser without logging into anything privileged.

Expected. Normal homepage loads. No admin link or admin functionality is visible in the UI.

Step 2: Modify the URL to reach the admin page

Action. In the address bar, append /admin to the lab URL and press Enter.

Expected. The admin panel loads successfully even though you were never shown a link to it.

Step 3: Interact with admin functionality

Action. Click Delete (or any admin action) on one of the listed users.

Expected. The action succeeds without any authentication or authorization check.

Example URL change

```
# Before
https://<lab-id>.web-security-academy.net/

# After (append /admin)
https://<lab-id>.web-security-academy.net/admin
```

Fallback if the live demo fails

- Screenshot of the lab homepage with no admin link visible (proof the UI hides the admin area).
- Screenshot of the same lab after appending /admin, showing the unprotected admin panel and the list of users with Delete buttons.

6. Common Pitfalls

When something does not work, check these first before asking for help.

SYMPTOM

Accessing /admin returns Not Found

Cause: Wrong URL or wrong lab instance.

Fix: Copy the exact lab URL from PortSwigger and append /admin to it.

SYMPTOM

Cannot reach admin despite correct URL

Cause: Lab not properly loaded or session expired.

Fix: Relaunch the lab from PortSwigger and try again.

SYMPTOM

Students think the attack is too simple

Cause: Misunderstanding that real systems sometimes rely on hidden URLs instead of real checks.

Fix: Emphasize this represents real-world misconfigurations across many production apps.

SYMPTOM

No visible change after editing the URL

Cause: Typo in the endpoint (for example, /admins instead of /admin).

Fix: Use the exact endpoint /admin, no trailing s, no extra slash.

7. Your Challenge

5 MIN BEGINNER

THE TASK

Use the provided PortSwigger lab to identify and access a restricted admin page by manipulating the URL.

SUCCESS CRITERION

Submit a screenshot showing the admin panel successfully accessed (URL bar must include /admin).

Hints

1. Try exploring common hidden paths in web applications.
2. Modify the URL manually instead of relying on the interface.
3. Append /admin to the base URL.

8. Cheatsheet

Keep this page next to you while you practice. Endpoints, status codes, and shortcuts only, no prose.

ENDPOINT / SIGNAL	WHAT IT MEANS	HOW TO USE
`/admin`	Most common admin endpoint to test for vertical escalation.	Append to the base URL and reload.
`/dashboard, /manage`	Alternate admin paths used by many web apps.	Try when /admin returns 404.
`/api/admin/*`	Admin actions exposed over an API.	Test with the same credentials a normal user has.
`/user/<id>`	Per-user resource path used for IDOR (horizontal escalation).	Swap the id with another user's id.
`?role=admin`	Parameter manipulation hint.	Add or change a role / privilege query parameter.
`403 Forbidden`	Server is enforcing access control. Good sign for defenders.	Means the endpoint exists but checks are working.
`200 OK on hidden URL`	Broken access control. The endpoint loaded without a check.	This is the bug. Capture proof and submit.
`POST / PUT / DELETE`	Some apps protect GET but forget mutating verbs.	Re-send the same path with a different HTTP method.

Quick mental model

```
# Vertical escalation (gain higher privilege)
regular user  →  GET /admin  →  loads admin panel

# Horizontal escalation (other user, same privilege)
user 123    →  GET /user/124  →  loads user 124's data
```

9. Further Reading

Go deeper on your own time with these three resources.

01

PortSwigger: Access Control

Interactive labs and explanations covering real-world access control vulnerabilities.

https://portswigger.net/web-security/access-control

02

OWASP Top 10: Broken Access Control

High-level overview of the most critical access control risks in modern applications.

https://owasp.org/Top10/2021/A01_2021-Broken_Access_Control/

03

OWASP Authorization Cheat Sheet

Practical guidelines for implementing secure access control correctly.

https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html