Perfect 👌 now we move to **WSTG-ATHZ-02: Testing for Bypassing Authorization Schema**.

**🧾 Testing for Bypassing Authorization Schema**

**1. Why It Matters**

Authorization is about **what an authenticated user can do**.

* If access controls are weak or poorly enforced, attackers can **bypass them** to access admin panels, other users’ data, or restricted functionality.
* Often caused by relying only on **client-side controls** (hidden fields, JavaScript checks, predictable URLs).

**2. Common Authorization Bypass Scenarios**

**🔹 Direct Object Reference (IDOR)**

* URL:
* https://target.com/user/profile?id=1001
* Change id=1001 → id=1002 to see another user’s profile.

**🔹 Parameter Modification**

* Example:
* role=user

→ Change to role=admin.

**🔹 Forced Browsing**

* Guessing/brute forcing URLs of restricted resources.
* /admin
* /superuser
* /config
* Tools: dirb, gobuster, ffuf.

**🔹 Method Tampering**

* Some endpoints allow **GET vs POST confusion**.
* Example: using GET /deleteUser?id=1 when UI only allows POST.

**🔹 Cookie Manipulation**

* Cookies storing roles/permissions in plaintext or weak encoding:
* Cookie: role=guest

Change → role=admin.

**🔹 Header Tampering**

* Some apps trust headers like X-Forwarded-For, X-Original-URL.  
  Example: bypass restrictions with:
* X-Original-URL: /admin

**🔹 Client-Side Enforcement**

* Hidden form fields or disabled buttons can be re-enabled via Burp Suite/DevTools.

**3. How to Test**

**🛠 Manual (Burp Suite is key here)**

* **Proxy Intercept:** Modify parameters (IDs, roles, cookies).
* **Repeater:** Replay requests with altered values.
* **Intruder:** Automate brute force on IDs or tokens.

**🛠 Automated**

* **OWASP ZAP** – scans for forced browsing and auth bypass.
* **Nmap NSE (http-auth-finder)** – can detect auth endpoints.
* **wfuzz/ffuf** – brute force directories.

**4. Examples**

**IDOR Exploit**

GET /user/profile?id=102 HTTP/1.1

Host: target.com

Cookie: session=abc123

If logged in as user A, but id=102 reveals user B → auth bypass.

**Cookie Role Tampering**

Before:

Cookie: user=John; role=user

After tampering:

Cookie: user=John; role=admin

**Forced Browsing**

GET /admin/config.php HTTP/1.1

Even though no admin link was shown in the UI.

**5. Impact**

* Unauthorized access to sensitive information.
* Privilege escalation → gaining admin rights.
* Tampering with other users’ accounts.
* Full system compromise in case of weak server-side enforcement.

**6. Mitigations**

* **Always enforce authorization on server side** (not just UI).
* Use **role-based access control (RBAC)** consistently.
* Use **indirect references** (UUIDs instead of sequential IDs).
* Securely validate cookies & headers (signed/encrypted tokens like JWT).
* Implement **least privilege principle**.

✅ **Summary**:  
Bypassing Authorization Schema happens when an attacker can trick the system into granting higher privileges or accessing restricted resources through **IDOR, forced browsing, parameter tampering, or cookie/header manipulation**. Testing involves using tools like **Burp Suite** and **ffuf** to probe for weak access control enforcement.

👉 Do you want me to give you a **Burp Suite practical lab flow** for simulating this (intercept → modify → escalate → test access)?