Authorization Flaws

Broken Access Control

Access control is supposed to prevent that users can act outside of their intended permissions.

Possible Impact of Broken Access Control

- Access unauthorized functionality and/or data, such as access other users' accounts
- View sensitive files
- Modify other users' data
- Change access rights

Common Attacks

- Modifying URL, internal application state, or HTML page

- Changing the primary key to another users record

- Elevation of privilege
    - Acting as a user without being logged in

    - Acting as an admin when logged in as a user

ℹ️ *Obtaining a higher level of access is also referred to as **Vertical** Privilege Escalation while same-level access to another user's data is called **Horizontal** Privilege Escalation.*

- Metadata manipulation
  - Replaying or tampering with access control tokens
  - Cookie or hidden field manipulation
- Force browsing to authenticated pages as an anonymous user or to privileged pages as a standard user
- Accessing API with missing access controls for `POST`, `PUT` and `DELETE`

Risk Rating

Broken Access Control

Exploitability	Prevalence	Detecability	Impact	Risk
🔶 Average	🔶 Common	🔶 Average	🔴 Severe	A5

Exercise (📌)

Assuming no access control is in place, which privilege escalations are possible by tampering with the following URLs?

1. `https://logistics-worldwi.de/showShipment?id=40643108`

2. `https://my-universi.ty/api/students/6503/exams/view`

3. `https://document-warehou.se/landingpage?content=index.html`

Prevention

- **Access control** is only effective if **enforced in trusted server-side code**

- With the exception of public resources, **deny by default**

- **Implement** access control mechanisms **once and re-use** them throughout the application

- **Enforce record ownership**

- **Disable web server directory listing** and ensure file metadata and backup files are not present within web roots

- **Log access control failures**, alert admins when appropriate

- Rate limit API and controller access to minimize the harm from automated attack tooling

- Access tokens should be invalidated on the server after logout

- Developers and QA staff should include functional access control unit and integration tests

Exercise

1. Access the *Admin Section* of the store (⭐⭐)

2. (🎓) *View* some other user's shopping *Basket* (⭐⭐)

3. Get rid of all submitted *Five-Star Feedback* (⭐⭐)

4. (🎓) Post some *Forged Feedback* for another user (⭐⭐⭐)

5. It's *Payback Time*: Place an order with a negative total (⭐⭐⭐)

6. Access the *Forgotten Developer* and/or *Sales Backup* (⭐⭐⭐⭐)