Web Architecture

A web site usually is a collection of web pages that are:

– Accessed by users over a network through the HTTP or HTTPS protocol
– Coded in a browser-supported programming language (i.e. JavaScript, HTML, etc.)
– Used through a common web browser (IE, Firefox, etc.) to render the pages executable, with usually the help of some cookies
– Managed by a web application with a client–server architecture (i.e. 3-tiers) in which Presentation, Logic, and Data tiers are logically separated
HTTP Proxy

• An intercepting Proxy:
  – inspect and modify traffic between your browser and the target application
  – Burp Intruder, OWASP ZAP, etc.
Web Access Control/Authorization

- Access control implement **fine-grained** logic to allow or deny user access to particular URLs and web sites
  - Rules can be configured with a number of different criteria that has to match in order for the rule to apply

- The **complexity** of access control is a frequent source of security vulnerabilities
  - E.g. Web designers often assume that users will progress sequentially through pages and authenticate them only on the first page
Broken Access Control

- **Vertical** Access Control Attacks
  - A standard user accessing administration functionality

- **Horizontal** Access Control Attacks
  - Same role, but accessing another user's private data

- **Business Logic** Access Control Attacks
  - Abuse of one or more linked activities that collectively realize a business objective
Access Controls Impact

• Loss of **accountability**
  – Attackers can execute maliciously:
    • actions as other users
    • higher level actions

• **Confidential data disclosure**
  – Access to admin level accounts usually allows to access to user’s data

• **Data tampering**
  – Privilege levels do not distinguish users who can only view data and users permitted to modify data
Common Vulnerabilities

• Completely unprotected functionality
  – Only URL is necessary to perform actions that should be restricted
    • “No user will ever know this URL, because it is hidden in the menu"

• Identifier based functions
  – Access to resource is mitigated by a parameter that is only handed out to a given user
    • https://cs.brown.edu/ViewDocument.php?docid=122342214
    • Monitoring application logs will reveal this type of functionality
Common Vulnerabilities II

• Use static files
  – Just a static resource is protected a difficult to guess name files
    • cs.brown.edu/downloads/solHW5-2974358195.pdf
  – This is a static resource that cannot verify the rights again
Common Vulnerabilities III

• Logic Flaws/False Assumptions
  – Multistage functionality
  – Example:
    1. User authenticates on “Menu” and clicks on “Add User”
       – Page verifies that user has privileges to add users
    2. Forwards user to the “Add User” page
       – But this one is not protected
       – Attack by going directly to this page
Common Vulnerabilities IV

• Insecure access control model based on request parameters submitted by the client
  – Parameter-Based
    • http://cs.brown.edu/login/home.jsp?admin=true
  – Location-Based
    • Many companies have regulatory requirement to restrict access to resources depending on the user’s geographic location
    • Bypassing with VPN or web proxy