ZROC102

Module 4 – Cross-site Scripting (XSS)

Topics

- What is XSS?
- Types of XSS
  - Reflected XSS
  - Stored XSS
  - DOM XSS*
- How to test for XSS
- Getting around Filters
- Raising our impact

What is XSS

Allows attacker to inject client-side script

Happens because developer does not sanitize input

XSS attacks are possible in VBScript, ActiveX, Flash, and even CSS

Most common in JavaScript

According to PWK guide(2018)

XSS by nature, attack the client and not the server.

"XSS vulnerabilities are caused due to unsanitized user input that is then displayed on a web page in HTML format.

They don't directly compromise a machine, these attacks can still have significant impacts,
such as cookie stealing
and authentication bypass,
redirecting the victim’s browser to a malicious HTML page, and more."

Types of XSS

- Three main attack types
  - Reflected XSS
  - Stored XSS, this includes blind XSS
  - DOM XSS
- Many smaller less known ones

Types of XSS - Reflected

- User input gets reflected
  - Into an attribute of an html tag
  - Into the HTML page
  - Into the javascript context
- User input does not gets stored into database
- User input is not properly sanitized
- User input can contain javascript code

Types of XSS - stored

- User input gets stored into database
- Database value gets reflected
  - Into HTML page
  - Into HTML tag attribute
  - Into the javascript context
- User input is not sanitized properly
  - At input
  - And at write to the database
  - And at read from database
  - User input can contain malicious javascript code

Testing for XSS

- Attack vectors
  - Depend on the context
    - JS: ‘”`…
      - Single quote, double quote, backtick... to break out of JS function
    - HTML: <img src=x>
      - First test for HTML injection, then expand to XSS
    - HTML attribute: '>">`>…
      - Single quote, double quote, backtick... to break out of html tag attribute
    - Basically if they break the page or insert an image, then we probably have to look deeper.
    - Replace the <img src> with our own tag. Eg <script>

Bypassing filters

- Different filters exist out there
- Blacklist based
  - Try fuzzing all possible HTML tags and javascript handlers
  - Try URL encoding XSS attack vector
  - Try double or triple encoding it
- Whitelist based
  - Hard to get around
  - Only allow certain words
- Pattern based
  - Tries to look for things that look like a XSS attack
  - Depends on the configuration

Impact of XSS

- Steal cookies
  - Very hard recently
- Execute a keylogger
  - Getting harder to smuggle out data
- Steal data from the page
  - Same as keylogger
- Execute JS functions on page