Web Security (continued)

Cross Site Scripting (XSS)

Stored XSS
Attacker can send a message to a victim which is stored on the server, when the victim visits the server the server sends the message to them which may contain script
- These scripts would be executed in the context of the server so they would have full access to the cookies for that website
- They can get the script to send the cookies to another server by various means such as iframes

Reflected XSS
Attacker sends a URL to a victim which contains the script code, the victim's client sends the bad message to the server and it is echoed back to the victim on the webpage

Defenses
- Kill all script tags iteratively (prevent <<script>script> too)
  - Doesn’t prevent stuff like onClick
- &lt;pre&gt;&lt;/pre&gt;
- Escape all &lt; and &gt;
  - We can still allow selected HTML by having a different markup language, like [b]mytext[/b]
- Blueprint
  - Kind of like prepared statements
    - Because it guarantees browser will parse document the same way the server does
  - Server takes input and parses it, looks for script nodes in the parse tree
  - Encodes parse tree as string that contains only letters and numbers
  - Small JS program sent back too to decode the encoded parse tree
- Challenges in dealing with invalid input
  - Properly recognizing invalid input
    - Have all input coming from the same cleanse function
  - Ensuring you always use your defense
    - Have auth attribute (random number) for open and close tags
  - Use noscript tags around output which may come from a user
    - Have auth attribute (random number) for open and close tags
    - You need to know the random number to close the tag, so attackers can't close the noscript tag in their input

Content Sniffing XSS Attacks
- Server is supposed to provide an accurate mime type
  - Sometimes doesn't happen, so browsers inspect the image themselves
  - Browser can decide that an image is actually HTML so it runs it as HTML
- Solutions
  - Don't upgrade to HTML (if file is declared as image/jpeg do not ever interpret as text/html)
  - Can be stricter about classifying as HTML when originally classified as non-html