INTRO TO WEB SECURITY
PART 1
INPUT SANITIZATION
UPCOMING EVENTS

Intro to Web Security Part 2 and 3
Oct. 19th and 26th

EKOPARTY CTF
Oct. 27th - 29th

Hack the Vote
Election weekend
DISCLAIMER

THE THINGS WE TEACH YOU DURING THESE PRESENTATIONS ARE NOT FOR USE ON RANDOM WEBSITES.... THAT’S ILLEGAL
TOOLS FOR WEB EXPLOITS

• A browser like Firefox or Chrome (or IE if you’re desperate)

• Browser extensions
  • Live HTTP Headers
  • Tamper Data

• Burp Suite
  • Software suite for testing web applications
THE ENVIRONMENT

• Our target: Web Server – serves web pages
• The client: Typically a web browser – sends requests to server for pages
LANGUAGES OF THE TRADE

• PHP
• JavaScript
• SQL
• HTML
PHP

- PHP: Hypertext Preprocessor
- General purpose scripting language used primarily for web development
- Interpreted server-side
- Can be embedded in HTML
JAVASCRIPT

• Dynamic (makes your web pages fancy and interactive)
• Also embedded in HTML
• Interpreted client-side
SQL

• Structured Query Language

• Used to query databases

• Access important information and sensitive user data
HTML

• Describes layout of a page
• May contain helpful comments or debugging info
• Generally uninteresting
CLIENT-SIDE VS SERVER-SIDE

• Client-Side Environment
  • Scripts run in the browser
  • Poses security risk if server sends pages containing malicious code

• Server-Side Environment
  • User request fulfilled and information is sent to web server
  • Web server returns page containing customized user information
GARBAGE IN, GARBAGE OUT

• Applications process whatever data is passed into them, regardless of whether it is data the application actually wanted

• Passing in bad data can lead to bad output

• Can manipulate the computer’s trust to access valuable info
HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.

OH, DEAR - DID HE BREAK SOMETHING? IN A WAY—

DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?

OH, YES. LITTLE BOBBY TABLES, WE CALL HIM.

WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.

AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.
INPUT VALIDATION VS SANITIZATION

• **Validation** is accepting or rejecting input based on the data you’ve requested
  • Ex: Program wants an integer, I enter a double; program exits with error

• **Sanitization** is the modification of user input, the application changes the input into an acceptable format
  • Ex: Program wants a 10-digit phone number, I submit (555) 666 – 777; program strips out all non-numeric values to create: 555666777
INPUT SANITIZATION

• Sanitization Whitelist
  • Any values not part of the approved “list” within an application are removed from
    the input

• Sanitization Blacklist
  • Eliminate or translate characters into “safe” values to be used by the application
  • Generally incomplete and requires a lot of maintenance
  • Simpler and faster to validate instead of sanitize
WHY IS THIS IMPORTANT?

• Without input sanitization, it’s possible for hackers to inject unexpected input to manipulate the program into providing information standard users should not be able to access (bank accounts, ssn, credit card numbers)
EXAMPLES

See at: 10.176.169.10/
A SIMPLE FORM

Welcome Sasserfras
Your email address is: email@example.com
Source code trusts the user’s input and blindly accepts it

```html
<html>
<body>
<form action="posted.php" method="post">
  Name: <input type="text" name="name"><br>
  E-mail: <input type="text" name="email"><br>
  <input type="submit">
</form>
</body>
</html>
```
MANIPULATING INPUT

Adding HTML tags to the code causes them to be run on the web page.

Name: <b>Sasserfras</b>
E-mail: 
Submit Query

Welcome Sasserfras
Your email address is:
WHAT ABOUT SCRIPTS?

By including JavaScript between the `<script>` tags, we can run code on the client.
WHAT ELSE CAN WE DO?

Embed a website in a page through use of iframe

<iframe src="https://csg.utdallas.edu/"></iframe>
MITIGATION

- Escape everything
A SIMPLE WEBSITE

GOAL: OBTAIN THE FLAG

We can see that there is a flag in the / directory of the web server.
DIRECTORY TRAVERSAL

We know we can move up a directory by adding “..” before the file path. Let’s try traversing to this url:

http://10.176.169.10/traverse/?page=../..//flag.txt
WE’RE IN
MITIGATION

- chroot jails
- blacklist characters like "./" and "*"
TO BE CONTINUED?

• Common vulnerabilities: Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), SQL Injection, etc.

• These attacks will be covered in PART 2
PRACTICE ON YOUR OWN

http://10.176.169.11/app/