Introduction to Languages
Languages

- PHP
- Javascript
- SQL
- HTML
PHP

- Interpreted Server Side
- Dynamic
- Handles GET/POST
- Manages Sessions
- Has Own Set of Vulnerabilities
  - Not Covered Here
```php
<?php
$q = intval($_GET['q']);

$con = mysql_connect('localhost','peter','abc123','my_db');
if (!con)
{
    die('Could not connect: ' . mysql_error($con));
}

mysql_select_db($con,"ajax_demo");
$sql="SELECT * FROM user WHERE id = "".$q."";

$result = mysql_query($con,$sql);

echo "<table border='1'>
<tr>
<th>Firstname</th>
<th>Lastname</th>
<th>Age</th>
<th>Hometown</th>
<th>Job</th>
</tr>";

while($row = mysql_fetch_array($result))
{
    echo "<tr>";
    echo "<td>" . $row['FirstName'] . "</td>";
    echo "<td>" . $row['LastName'] . "</td>";
    echo "<td>" . $row['Age'] . "</td>";
    echo "<td>" . $row['Hometown'] . "</td>";
    echo "<td>" . $row['Job'] . "</td>";
    echo "</tr>";
}
echo "</table>";

mysql_close($con);
?>```
PHP

- Session Demo
- 10.176.169.7/web_demo/sample.php
Javascript

- Dynamic
- Embedded in HTML
- Interpreted Client Side!!!
Demo time!

10.176.169.7/web_demo/js.html
SQL

- Query Databases
- Most Common for CTFs
- Used to Access Data
  - Usernames
  - Passwords
  - Credit Card #s
  - Fun Stuff
To select a user:
SELECT * from users WHERE name = 'Bob';

The username is determined at runtime, so let’s make it:
SELECT * from users WHERE name = '$name';

For example, if $name is “Joe”:
SELECT * from users WHERE name = 'Joe';
HTML

- Describes Layout of Webpage
- Sometimes Contains Debug Info
- Usually not very interesting...
HTTP

- Protocol that provides the way to communicate over the web
- It is stateless and asynchronous
  - Simulate state with sessions
  - Your browser keeps session information
  - The server uses this to keep track of your state
- Example: Shopping Cart
  - Session has an ID tied to a cart in database
  - Every page you visit has to establish your identity
HTTP Requests

- **Methods**
  - GET – asks server for information
  - POST – gives server data
  - PUT – tells server to modify or create data
  - DELETE – tells server to delete data

- **Examples**
  - GET shows your profile on a webpage
  - POST is used to upload your picture
  - PUT changes your bio
  - DELETE gets rid of the embarrassing picture
Along with URL and method, requests carry data in the form of parameters

**GET**
- Visible from URL:
- Can be used easily in hyperlinks

**POST**
- Not visible in URL or link, embedded in request
- We can still alter these
Parameter Tampering
Overview

- Very basic attack on HTTP protocol
- Exploits server’s misguided trust in data from user
Example – Game High Scores

Give me a game

Here’s one

Web Server
Example – Game High Scores

Game (Local) → Score → Web Server
Here’s how I did…

Nice!

Game (Local)

Web Server
Attack – Game High Scores

Here’s how I SAY I did...

Nice!

Score

Game (Local)

Web Server
Example – PayPal

I want to buy this

Pay for it with PayPal
Example – PayPal

Here’s how much I owe you. Sounds good.
Example – PayPal

I paid

Thanks!

Tell them you paid
Here’s how much I say I owe you.

Sounds good.
Attack – PayPal

PayPal

Tell them you paid

I paid what you said

Thanks!

Tell them you paid

Merchant

PayPal
Tools and Demo

- Firefox
  - TamperData
  - Live HTTP Headers
- BurpSuite
Mitigation

- Never trust the integrity of data that a user can edit
- Web services can allow servers to talk and bypass the user
End of Week 1