Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Student

Name: Mahitha Kalaga

Email: kalagam1@udayton.edu

Repository Information

Repository's URL: https://github.com/kalagam1/waph-kalagam1.git

This is a private repository for Mahitha Kalaga to store all the code from the course. The organization of this repository is as follows.

Labs

Hands-on exercises in Lectures

- Lab 0: Development Environment Setup
- Lab 1: Foundations of the Web
- Lab 2: Front-end Web Development
- Lab 3: Secure Web Application Development in PHP/MySQL
- Lab 4: A Secure Login System with Session Authentication

Hackathons

- Hackathon 1: Cross-site Scripting Attacks and Defenses
- Hackathon 2: SQL Injection Attacks

Individual Projects

- Individual Project 1: Front-end Web Development with a Professional Profile Website and API Integration on github.io cloud service

Report

The lab's overview

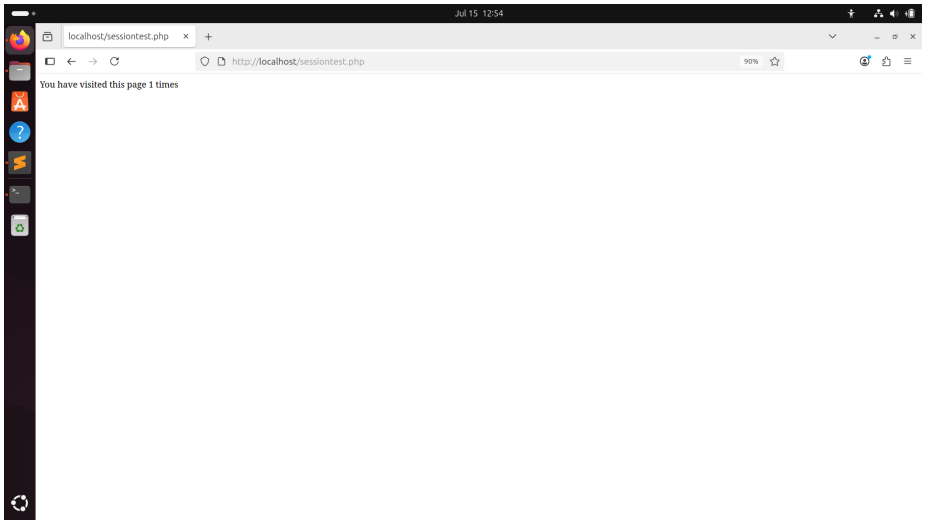
This lab focused on implementing secure session management in PHP-based web applications, highlighting risks like session hijacking and ways to defend against them. Tasks included deploying and testing session behavior, analyzing HTTP traffic with Wireshark, simulating session hijacking attacks, and building a basic login system using sessions. Security measures such as enabling HTTPS, setting HttpOnly and Secure cookie flags, and detecting session hijacking through browser checks were implemented. Through this lab, I gained practical experience in both exploiting insecure session handling and applying effective countermeasures to protect user authentication.

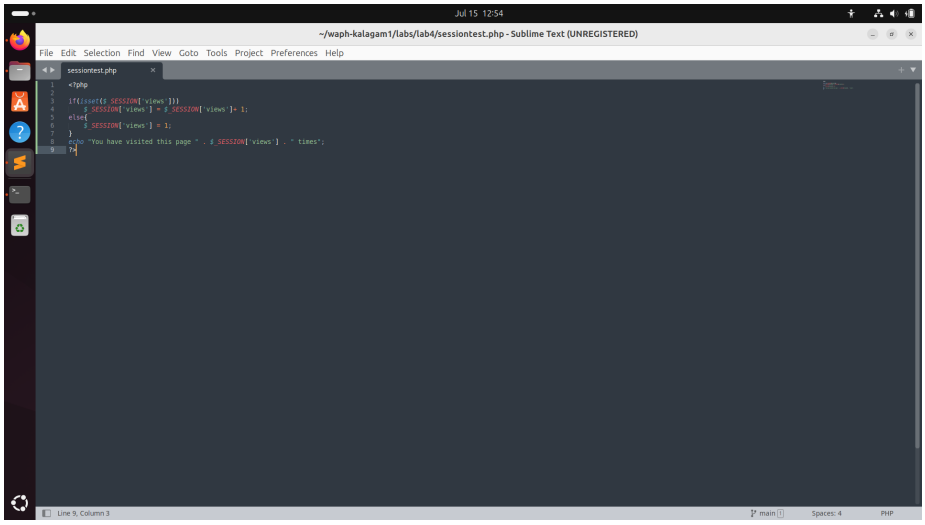
Lab's URL: Lab4

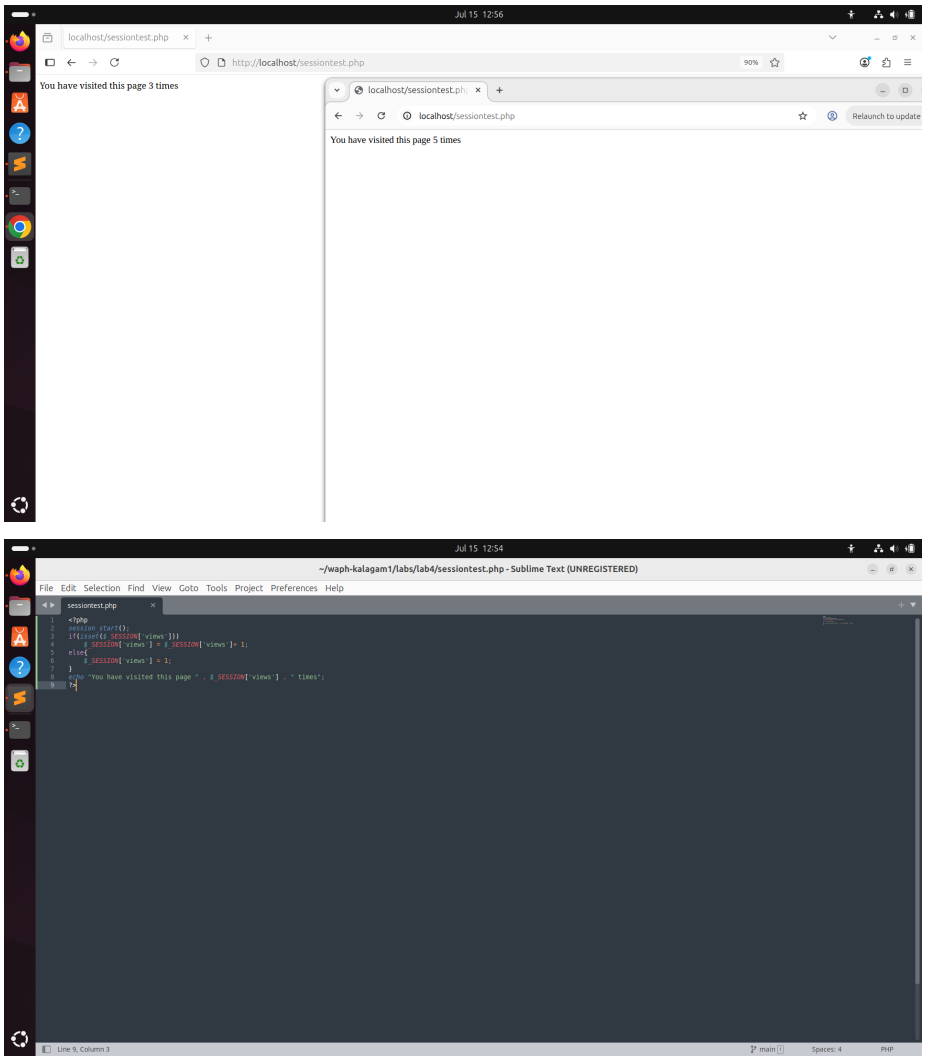
Part 1 - Understanding Session Management in a PHP Web Application

Task 1.A: Deploy and test sessiontest.php The script did not track the number of visits due to the absence of session management logic. To enable session tracking, I inserted the session_start(); function at the top of the PHP file. After making this change and deploying the file to the web directory using sudo cp, the page began displaying and updating the session visit count with each

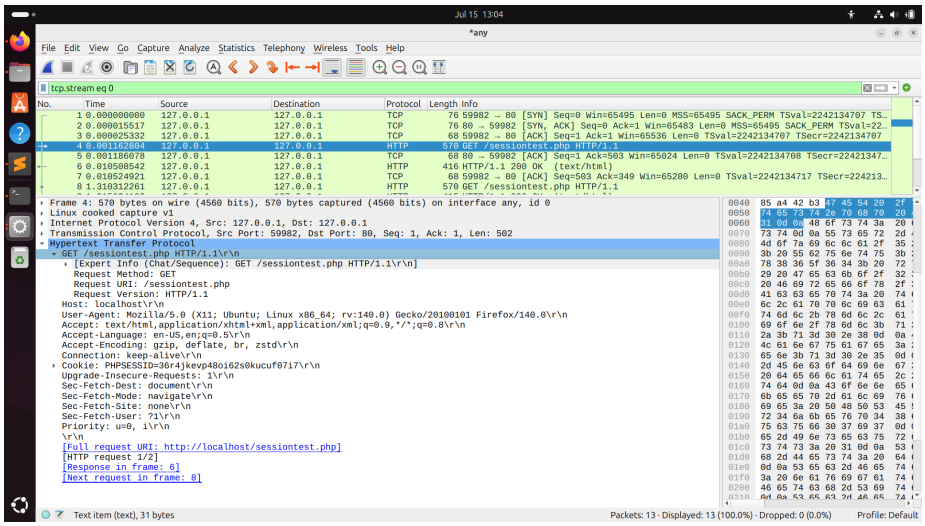
refresh. Testing in Chrome and Firefox showed different session IDs, verifying that PHP maintains sessions independently per browser.

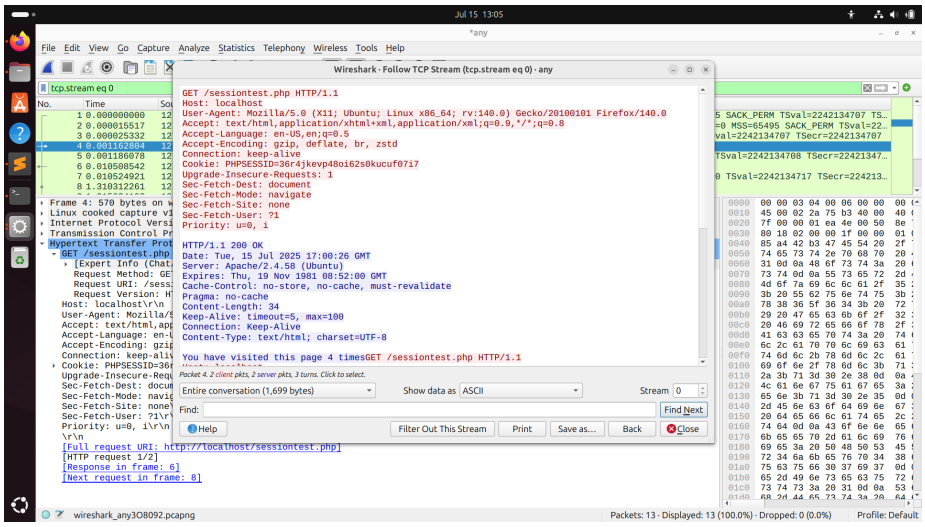


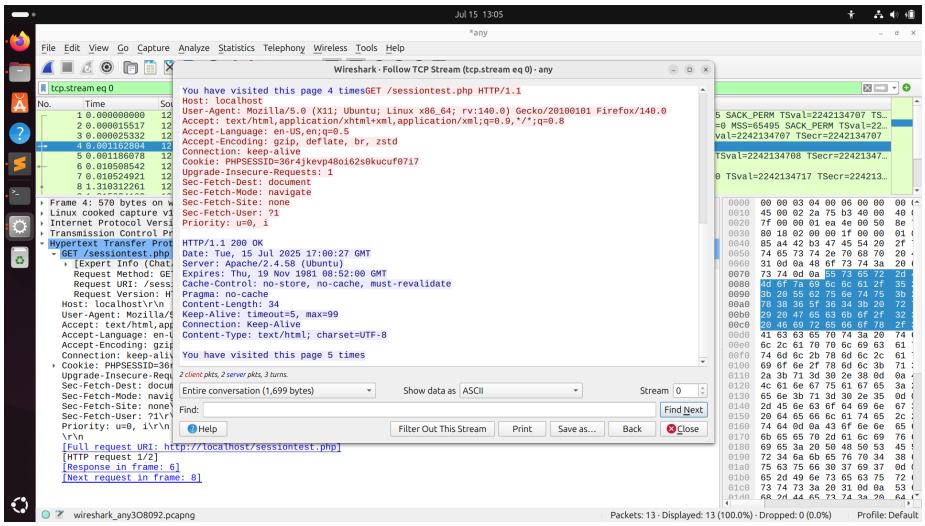




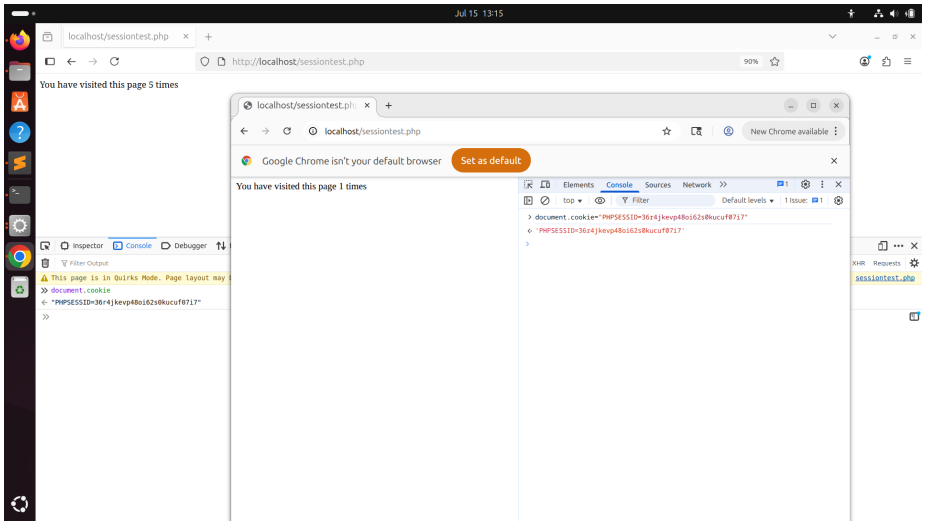
Task 1.B: Observing Session Handshaking with Wireshark After deploying session.php, I launched Wireshark to analyze the HTTP traffic during a page visit. On the first visit, there was no cookie sent by the client. However, the server responded with a Set-Cookie header containing a unique session ID. On frequent visits, this session ID was included in the client's request via the Cookie header, demonstrating the session persistence mechanism.

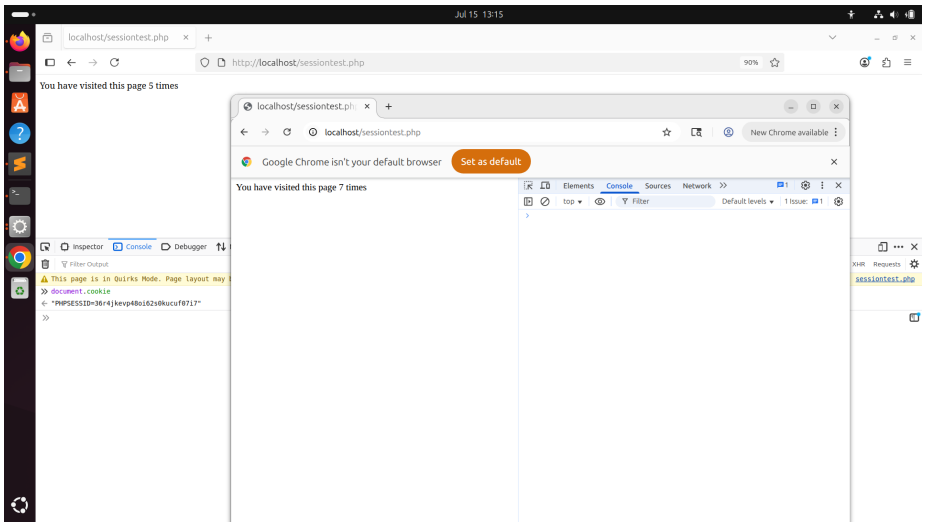

Jul 15 13:04
*any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.stream eq 0
No. Time Source Destination Protocol Length Info
1 0.000000000 127.0.0.1 127.0.0.1 TCP 76 59982 → 80 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=2242134707 TS…
2 0.000015517 127.0.0.1 127.0.0.1 TCP 76 80 → 59982 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=22…
3 0.000025332 127.0.0.1 127.0.0.1 TCP 68 59982 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2242134707 TSecr=2242134707
4 0.001162804 127.0.0.1 127.0.0.1 HTTP 570 GET /sessiontest.php HTTP/1.1
5 0.001186078 127.0.0.1 127.0.0.1 TCP 68 80 → 59982 [ACK] Seq=1 Ack=503 Win=65024 Len=0 TSval=2242134708 TSecr=22421347…
6 0.010508542 127.0.0.1 127.0.0.1 HTTP 416 HTTP/1.1 200 OK (text/html)
7 0.010524921 127.0.0.1 127.0.0.1 TCP 68 59982 → 80 [ACK] Seq=503 Ack=349 Win=65280 Len=0 TSval=2242134717 TSecr=224213…
8 1.310312261 127.0.0.1 127.0.0.1 HTTP 570 GET /sessiontest.php HTTP/1.1
Frame 4: 570 bytes on wire (4560 bits), 570 bytes captured (4560 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 59982, Dst Port: 80, Seq: 1, Ack: 1, Len: 502
Hypertext Transfer Protocol
GET /sessiontest.php HTTP/1.1\r\n
[Expert Info (Chat/Sequence): GET /sessiontest.php HTTP/1.1\r\n]
Request Method: GET
Request URI: /sessiontest.php
Request Version: HTTP/1.1
Host: localhost\r\n
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0\r\n
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
Accept-Language: en-US,en;q=0.5\r\n
Accept-Encoding: gzip, deflate, br, zstd\r\n
Connection: keep-alive\r\n
Cookie: PHPSESSID=36r4jkevp48oi62s0kucuf07i7\r\n
Upgrade-Insecure-Requests: 1\r\n
Sec-Fetch-Dest: document\r\n
Sec-Fetch-Mode: navigate\r\n
Sec-Fetch-Site: none\r\n
Sec-Fetch-User: ?1\r\n
Priority: u=0, i\r\n
\r\n
[Full request URI: http://localhost/sessiontest.php]
[HTTP request 1/2]
[Response in frame: 6]
[Next request in frame: 8]
Text item (text), 31 bytes
Packets: 13 · Displayed: 13 (100.0%) · Dropped: 0 (0.0%)
Profile: Default


Jul 15 13:05
*any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Wireshark · Follow TCP Stream (tcp.stream eq 0) · any
GET /sessiontest.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br, zstd
Connection: keep-alive
Cookie: PHPSESSID=36r4jkevp48oi62s0kucuf07i7
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Priority: u=0, i
HTTP/1.1 200 OK
Date: Tue, 15 Jul 2025 17:00:26 GMT
Server: Apache/2.4.58 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 34
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
You have visited this page 4 timesGET /sessiontest.php HTTP/1.1
Packet 4. 2 client pkts, 2 server pkts, 3 turns. Click to select.
Entire conversation (1,699 bytes)
Show data as ASCII
Stream 0
Find:
Find Next
Help
Filter Out This Stream
Print
Save as…
Back
Close
wireshark_any3O8092.pcapng
Packets: 13 · Displayed: 13 (100.0%) · Dropped: 0 (0.0%)
Profile: Default



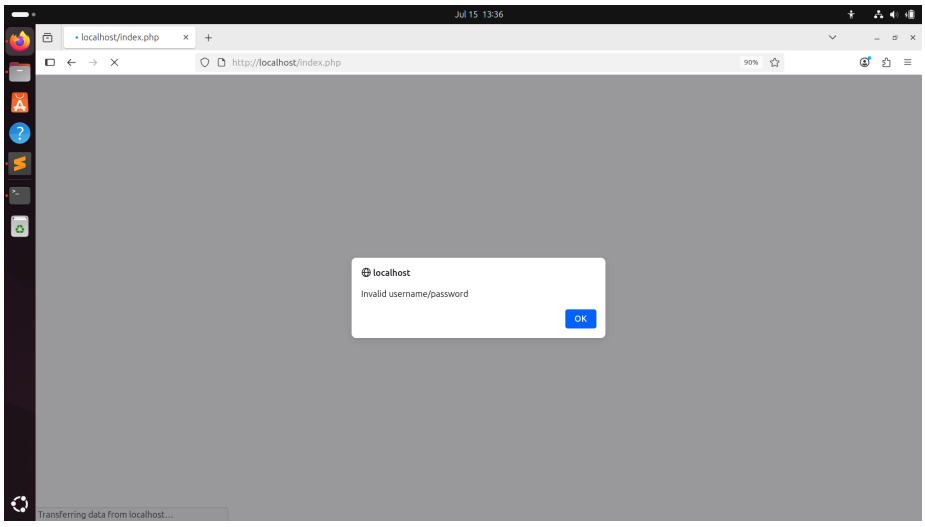
Task 1.C: Understanding Session Hijacking To demonstrate session hijacking, I loaded the session.php page in Chrome, used the document.cookie command to retrieve the session ID, and manually copied it into Firefox's developer console. After setting the cookie using document.cookie = "PHPSESSID=..." in Firefox, the page displayed the same session state as Chrome.

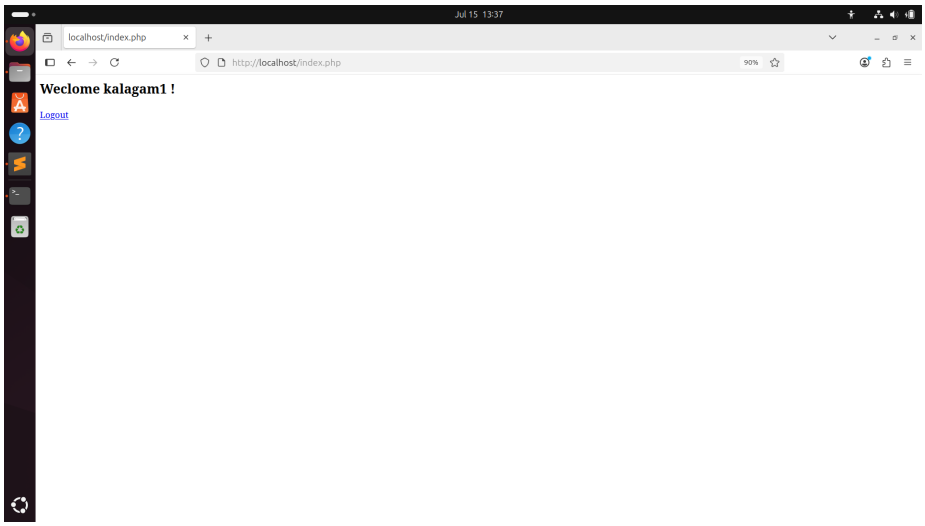


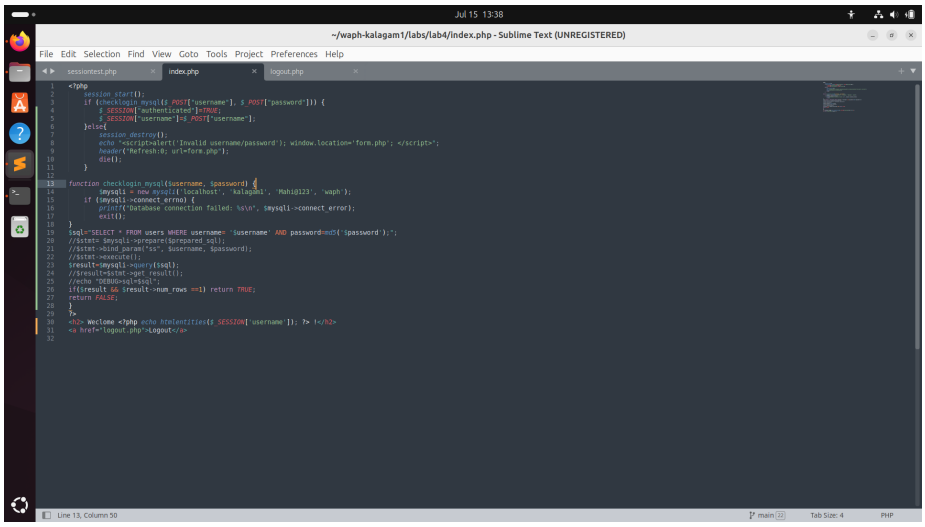


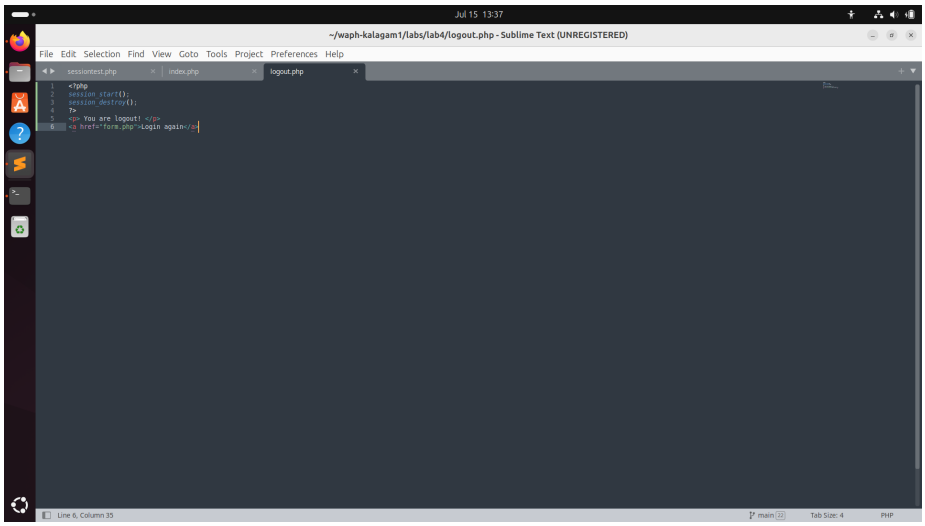
Task 2: Insecure Session Authentication

Task 2.A: Revised Login System with Session Management I took the index.php file from Lab 3 and modified it to include session-based access control. I copied the files (index.php, form.php, logout.php) to the Lab 4 directory and edited them. The logic was updated to verify authentication using session variables and restrict access to unauthorized users. A logout.php script was added to destroy the session. After deployment, the system correctly displayed login errors for invalid credentials and maintained authenticated sessions for valid users.

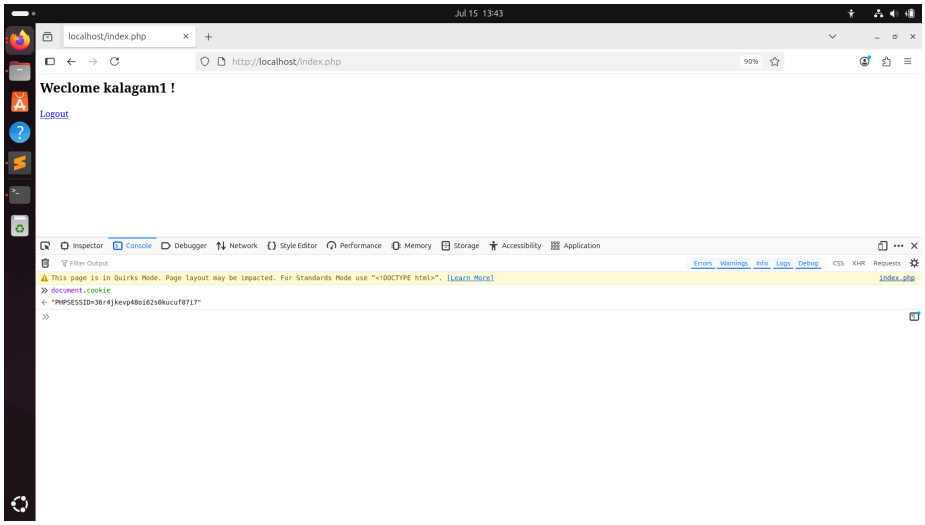


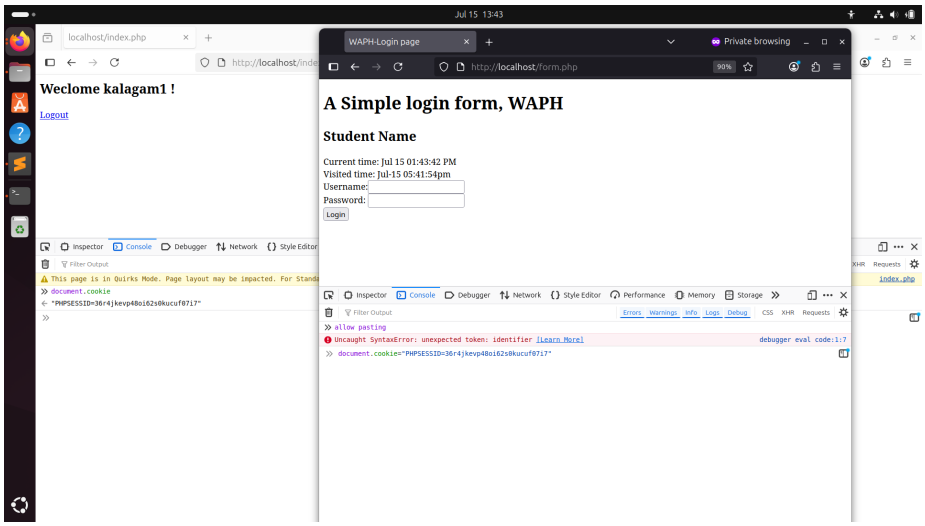

Jul 15 13:37
localhost/index.php
http://localhost/index.php
90%
Weclome kalagam1 !
Logout

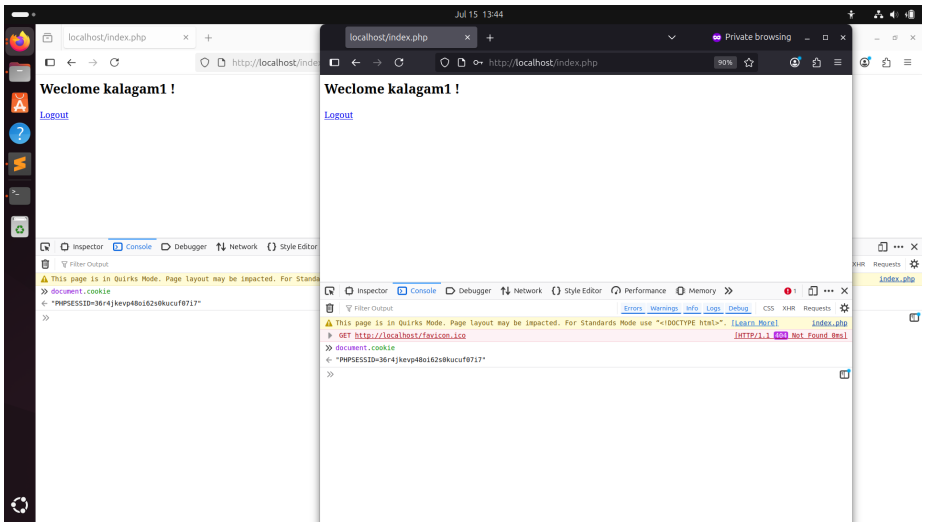

Jul 15 13:38
~/waph-kalagam1/labs/lab4/index.php - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
sessiontest.php
index.php
logout.php
<?php
session_start();
if (checklogin_mysqli($_POST['username'], $_POST['password'])) {
$_SESSION["authenticated"]=TRUE;
$_SESSION['username']=$_POST['username'];
}else{
session_destroy();
echo "<script>alert('Invalid username/password'); window.location='form.php';</script>";
header('Refresh:0; url=form.php');
die();
}
function checklogin_mysqli($username, $password) {
$mysqli = new mysqli('localhost', 'kalagam1', 'Mahi@123', 'waph');
if ($mysqli->connect_errno) {
printf("Database connection failed: %s\n", $mysqli->connect_error);
exit();
}
$sql="SELECT * FROM users WHERE username= '$username' AND password=md5('$password');";
//$stmt= $mysqli->prepare($prepared_sql);
//$stmt->bind_param("ss", $username, $password);
//$stmt->execute();
$result=$mysqli->query($sql);
//$result=$stmt->get_result();
//echo "DEBUG>sql=$sql";
if($result && $result->num_rows ==1) return TRUE;
return FALSE;
}
?>
<h2> Weclome <?php echo htmlentities($_SESSION['username']); ?> !</h2>
Logout
Line 13, Column 50
main
Tab Size: 4
PHP



Task 2.B: Session Hijacking Attacks To demonstrate vulnerability in the revised login system, I used document.cookie to extract the session ID from a logged-in user in one browser. I then injected this session ID into another browser. Upon refreshing the second browser, it was granted access to the authenticated session, proving that session hijacking is possible if the session ID is exposed.

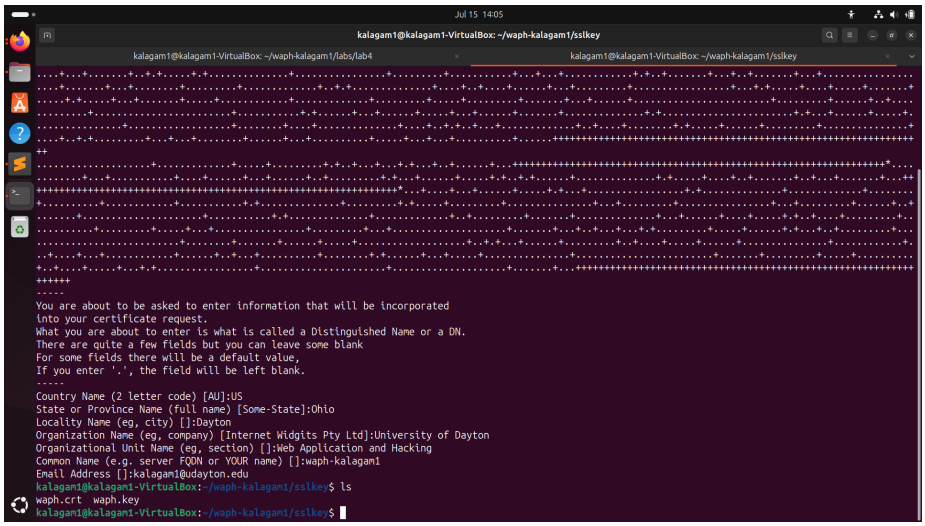






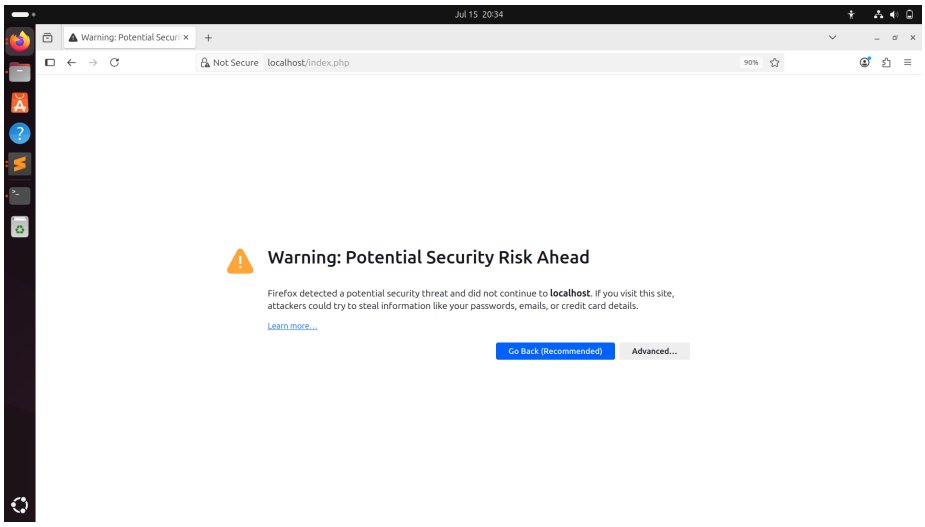
Task 3: Securing Session and Session Authentication

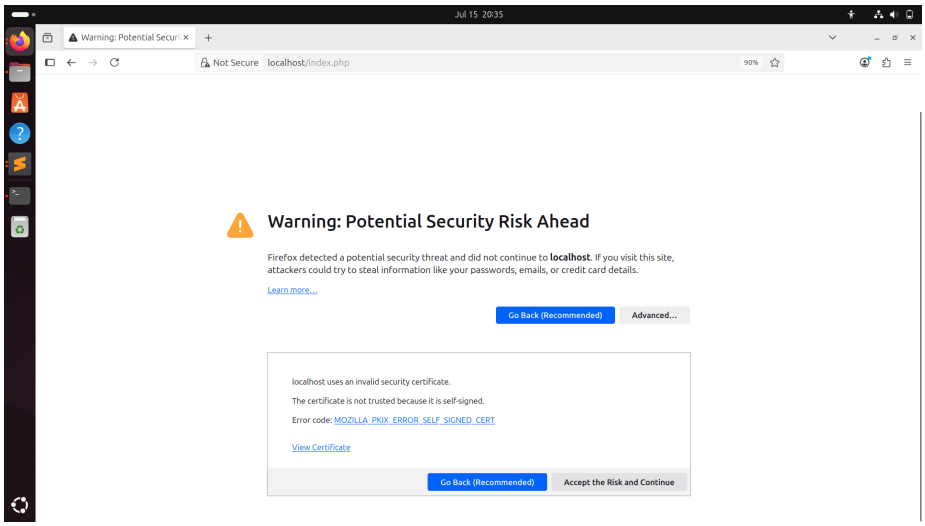
Task 3.A: Data Protection and HTTPS Setup To secure data in transit, I generated a self-signed SSL certificate using OpenSSL with a 4096-bit RSA key. The .crt and .key files were copied into the Apache SSL directories (/etc/ssl/certs and /etc/ssl/private). I then edited the Apache SSL configuration file (default-ssl.conf) to point to the certificate files. After enabling the SSL module and restarting Apache, the application became accessible via HTTPS.

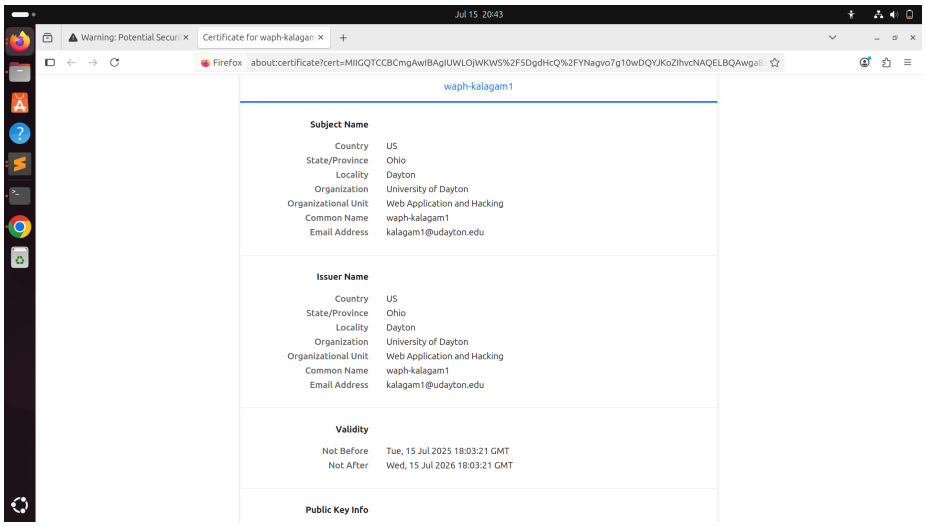

Jul 15 14:05
kalagam1@kalagam1-VirtualBox: ~/waph-kalagam1/sslkey
kalagam1@kalagam1-VirtualBox: ~/waph-kalagam1/labs/lab4
kalagam1@kalagam1-VirtualBox: ~/waph-kalagam1/sslkey

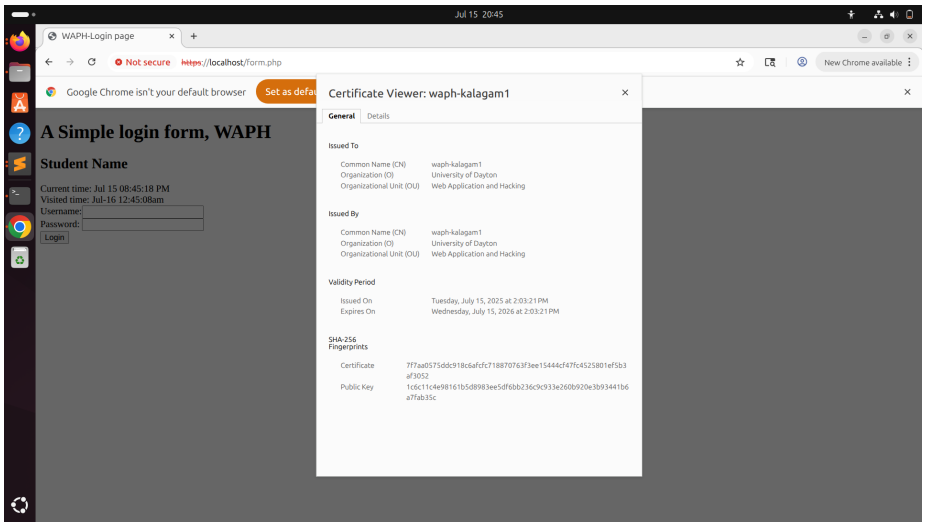
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

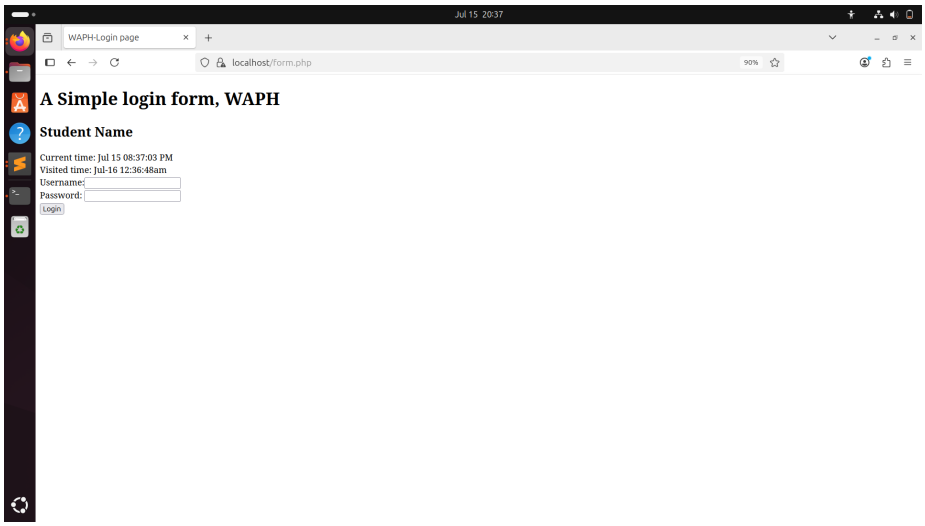
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:Ohio
Locality Name (eg, city) []:Dayton
Organization Name (eg, company) [Internet Widgits Pty Ltd]:University of Dayton
Organizational Unit Name (eg, section) []:Web Application and Hacking
Common Name (e.g. server FQDN or YOUR name) []:waph-kalagam1
Email Address []:kalagam1@udayton.edu
kalagam1@kalagam1-VirtualBox:~/waph-kalagam1/sslkey$ ls
waph.crt waph.key
kalagam1@kalagam1-VirtualBox:~/waph-kalagam1/sslkey$

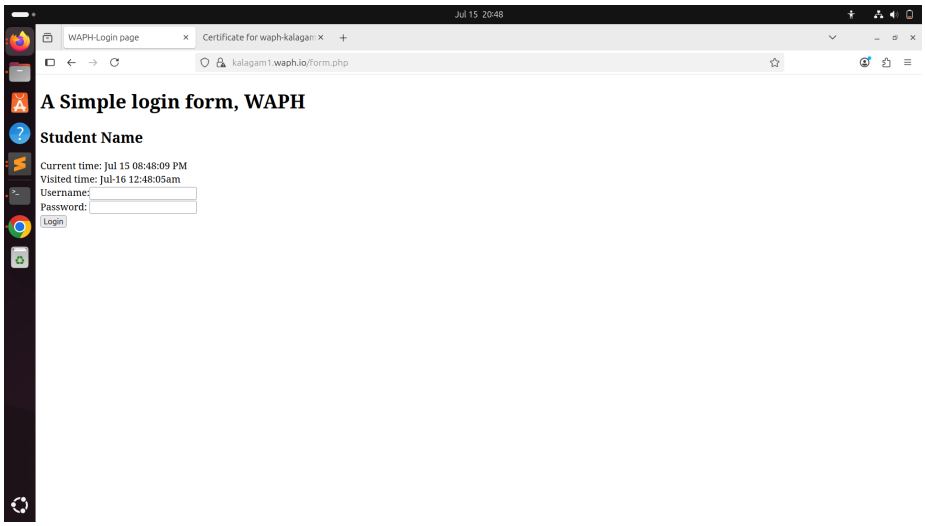

Jul 15 20:34
Warning: Potential Secur
Not Secure
localhost/index.php
90%
Warning: Potential Security Risk Ahead
Firefox detected a potential security threat and did not continue to localhost. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.
Learn more...
Go Back (Recommended)
Advanced...


Jul 15 20:35
Warning: Potential Secur
Not Secure localhost/index.php
90%
Warning: Potential Security Risk Ahead
Firefox detected a potential security threat and did not continue to localhost. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.
Learn more...
Go Back (Recommended)
Advanced...
localhost uses an invalid security certificate.
The certificate is not trusted because it is self-signed.
Error code: MOZILLA_PKIX_ERROR_SELF_SIGNED_CERT
View Certificate
Go Back (Recommended)
Accept the Risk and Continue

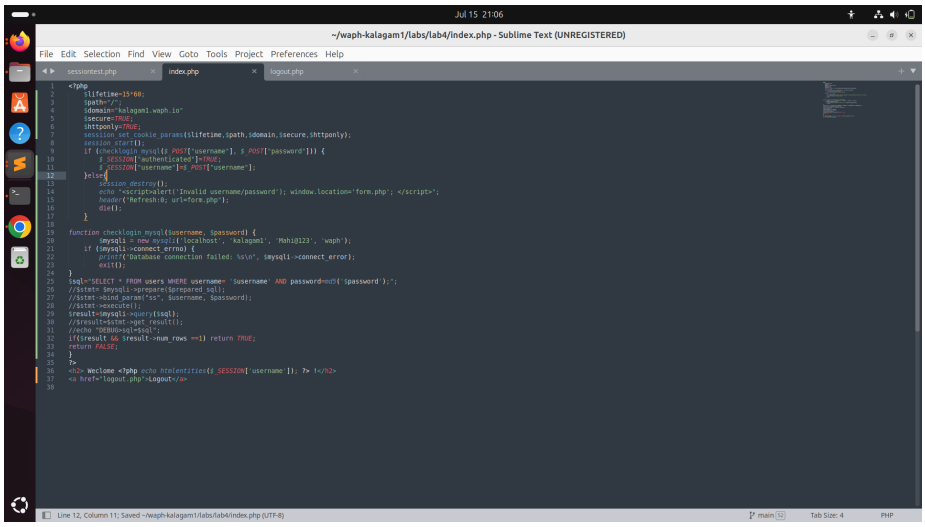

Jul 15 20:43
Warning: Potential Secur
Certificate for waph-kalagam
Firefox about:certificate?cert=MIIGQTCCBCmgAwIBAgIUWLOjWKWS%2F5DgdHcQ%2FYNagvo7g10wDQYJKoZIhvcNAQELBQAwgaE
waph-kalagam1
Subject Name
Country US
State/Province Ohio
Locality Dayton
Organization University of Dayton
Organizational Unit Web Application and Hacking
Common Name waph-kalagam1
Email Address kalagam1@udayton.edu
Issuer Name
Country US
State/Province Ohio
Locality Dayton
Organization University of Dayton
Organizational Unit Web Application and Hacking
Common Name waph-kalagam1
Email Address kalagam1@udayton.edu
Validity
Not Before Tue, 15 Jul 2025 18:03:21 GMT
Not After Wed, 15 Jul 2026 18:03:21 GMT
Public Key Info

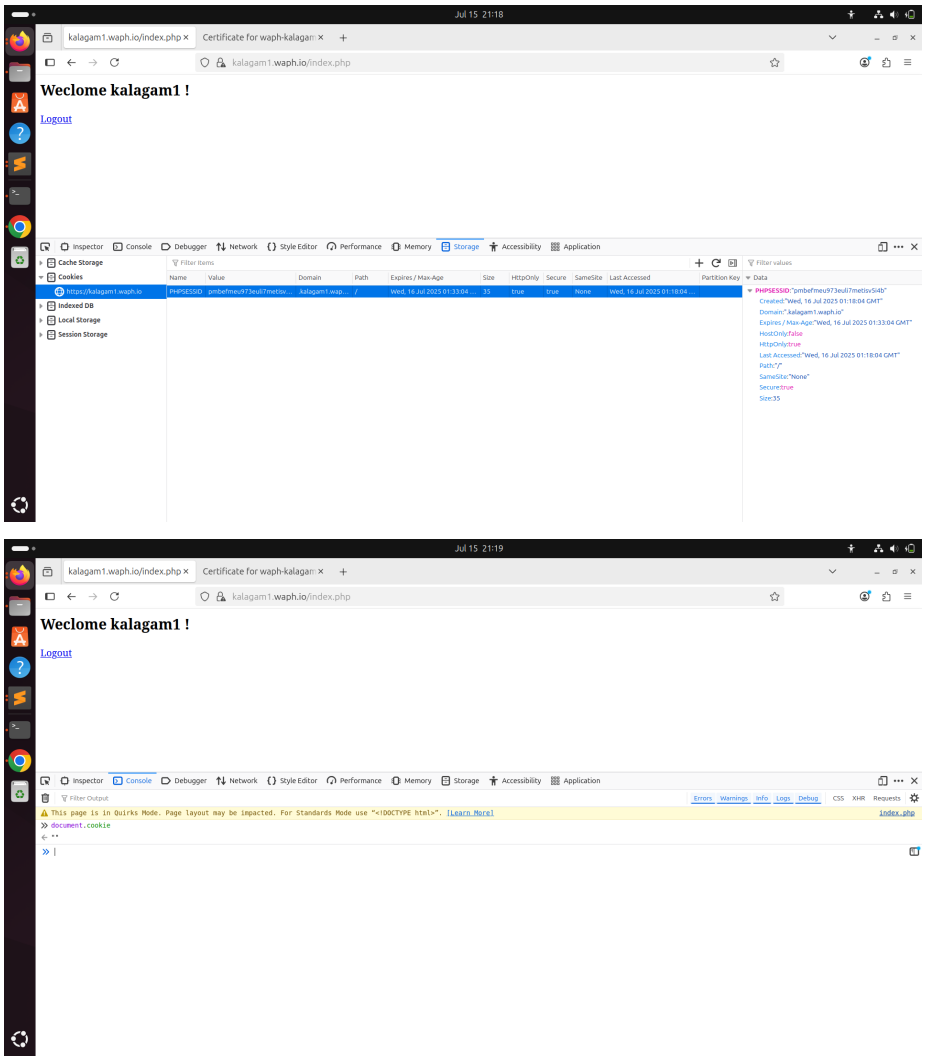

Jul 15 20:45
WAPH-Login page
Not secure
https://localhost/form.php
New Chrome available
Google Chrome isn't your default browser
Set as defau
A Simple login form, WAPH
Student Name
Current time: Jul 15 08:45:18 PM
Visited time: Jul-16 12:45:08am
Username:
Password:
Login
Certificate Viewer: waph-kalagam1
General
Details
Issued To
Common Name (CN) waph-kalagam1
Organization (O) University of Dayton
Organizational Unit (OU) Web Application and Hacking
Issued By
Common Name (CN) waph-kalagam1
Organization (O) University of Dayton
Organizational Unit (OU) Web Application and Hacking
Validity Period
Issued On Tuesday, July 15, 2025 at 2:03:21 PM
Expires On Wednesday, July 15, 2026 at 2:03:21 PM
SHA-256 Fingerprints
Certificate 7f7aa0575ddc918c6afcfc71887076 3f3ee15444cf47fc452580 1ef5b3af3052
Public Key 1c6c11c4e98161b5d8983ee5df6bb236c9c933e260b920e3b93441b6a7fab35c


Jul 15 20:37
WAPH-Login page
localhost/form.php
90%
A Simple login form, WAPH
Student Name
Current time: Jul 15 08:37:03 PM
Visited time: Jul-16 12:36:48am
Username:
Password:
Login

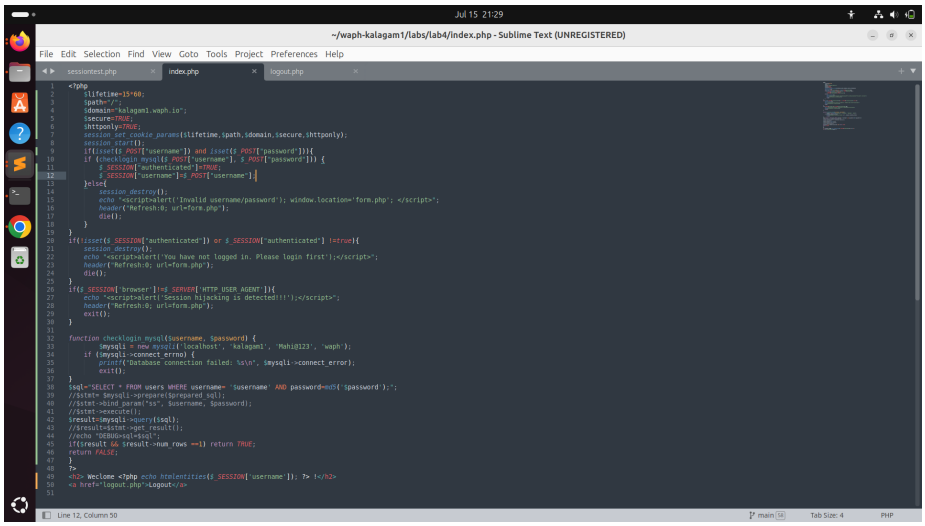


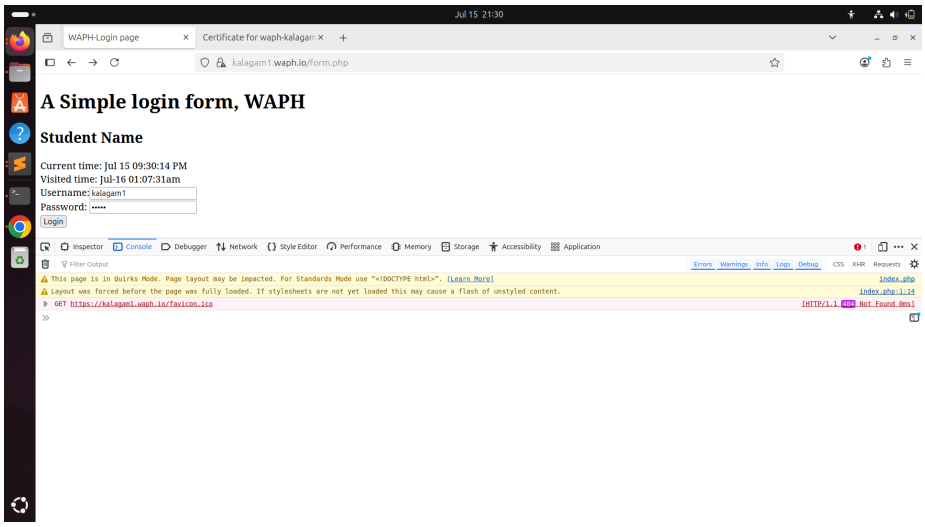
Task 3.B: Securing Sessions with HttpOnly and Secure Flags To strengthen session protection, I modified the PHP configuration to set the HttpOnly and Secure flags for cookies. These settings were defined using session_set_cookie_params() before invoking session_start(). The HttpOnly flag ensures cookies cannot be accessed via JavaScript, mitigating XSS risks, while the Secure flag ensures cookies are only sent over HTTPS. These changes were tested by attempting to access the session cookie from the browser console.

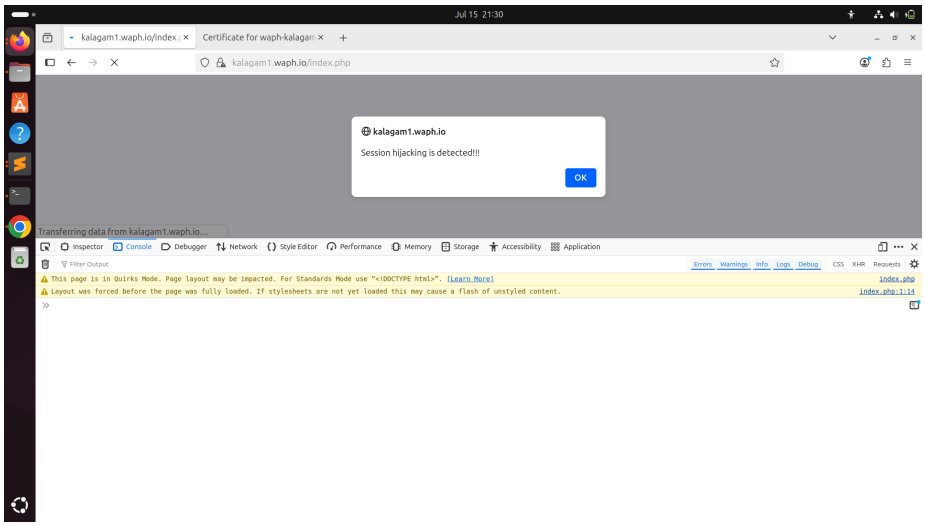




Task 3.C: Defense In-Depth: Detecting Session Hijacking I implemented session hijack detection by checking for mismatches in client information (e.g., user agent or IP address) between requests. If a mismatch was detected, the application triggered a JavaScript alert stating "Session hijacking is detected" and redirected the user to the login page.


Jul 15 21:29
~/waph-kalagam1/labs/lab4/index.php - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
sessiontest.php
index.php
logout.php
<?php
$lifetime=15*60;
$path="/";
$domain="kalagam1.waph.io";
$secure=TRUE;
$httponly=TRUE;
session_set_cookie_params($lifetime,$path,$domain,$secure,$httponly);
session_start();
if(isset($_POST['username']) and isset($_POST['password'])){
if (checklogin_mysqli($_POST['username'], $_POST['password'])) {
$_SESSION["authenticated"]=TRUE;
$_SESSION['username']=$_POST['username'];
}else{
session_destroy();
echo "<script>alert('Invalid username/password'); window.location='form.php'; </script>";
header('Refresh:0; url=form.php');
die();
}
}
if(!isset($_SESSION['authenticated']) or $_SESSION['authenticated'] !=true){
session_destroy();
echo "<script>alert('You have not logged in. Please login first');</script>";
header('Refresh:0; url=form.php');
die();
}
if($_SESSION['browser']!=$_SERVER['HTTP_USER_AGENT']){
echo "<script>alert('Session hijacking is detected!!!');</script>";
header('Refresh:0; url=form.php');
exit();
}
function checklogin_mysqli($username, $password) {
$mysqli = new mysqli('localhost', 'kalagam1', 'Mahi@123', 'waph');
if (mysqli_connect_errno) {
printf("Database connection failed: %s\n", $mysqli->connect_error);
exit();
}
$sql="SELECT * FROM users WHERE username= '$username' AND password=md5('$password');";
//$stmt= $mysqli->prepare($prepared_sql);
//$stmt->bind_param("ss", $username, $password);
//$stmt->execute();
$result=$mysqli->query($sql);
//$result=$stmt->get_result();
//echo "DEBUG=$sql=$sql";
if($result && $result->num_rows ==1) return TRUE;
return FALSE;
}
?>
<h2> Welcome <?php echo htmlentities($_SESSION['username']); ?> !</h2>
Logout
Line 12, Column 50
main
Tab Size: 4
PHP


Jul 15 21:30
WAPH-Login page
Certificate for waph-kalagam
kalagam1.waph.io/form.php
A Simple login form, WAPH
Student Name
Current time: Jul 15 09:30:14 PM
Visited time: Jul-16 01:07:31am
Username: kalagam1
Password: •••••
Login
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Filter Output
Errors
Warnings
Info
Logs
Debug
CSS
XHR
Requests
This page is in Quirks Mode. Page layout may be impacted. For Standards Mode use "<!DOCTYPE html>". [Learn More]
index.php
Layout was forced before the page was fully loaded. If stylesheets are not yet loaded this may cause a flash of unstyled content.
index.php:1:14
GET https://kalagam1.waph.io/favicon.ico
[HTTP/1.1 404 Not Found 0ms]


Jul 15 21:30
kalagam1.waph.io/index.
Certificate for waph-kalagam
kalagam1.waph.io/index.php
kalagam1.waph.io
Session hijacking is detected!!!
OK
Transferring data from kalagam1.waph.io...
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Filter Output
Errors
Warnings
Info
Logs
Debug
CSS
XHR
Requests
This page is in Quirks Mode. Page layout may be impacted. For Standards Mode use "<!DOCTYPE html>". [Learn More]
index.php
Layout was forced before the page was fully loaded. If stylesheets are not yet loaded this may cause a flash of unstyled content.
index.php:1:14