WAPH-Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Student

Name: Amit Gaddi

Email: gaddiat@mail.uc.edu

Short-bio: Amit has keen interests in IT.

Repository Information

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

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

Lab 4 -A Secure Login Systwm With Session Authentication

Lab4 Link

Overview

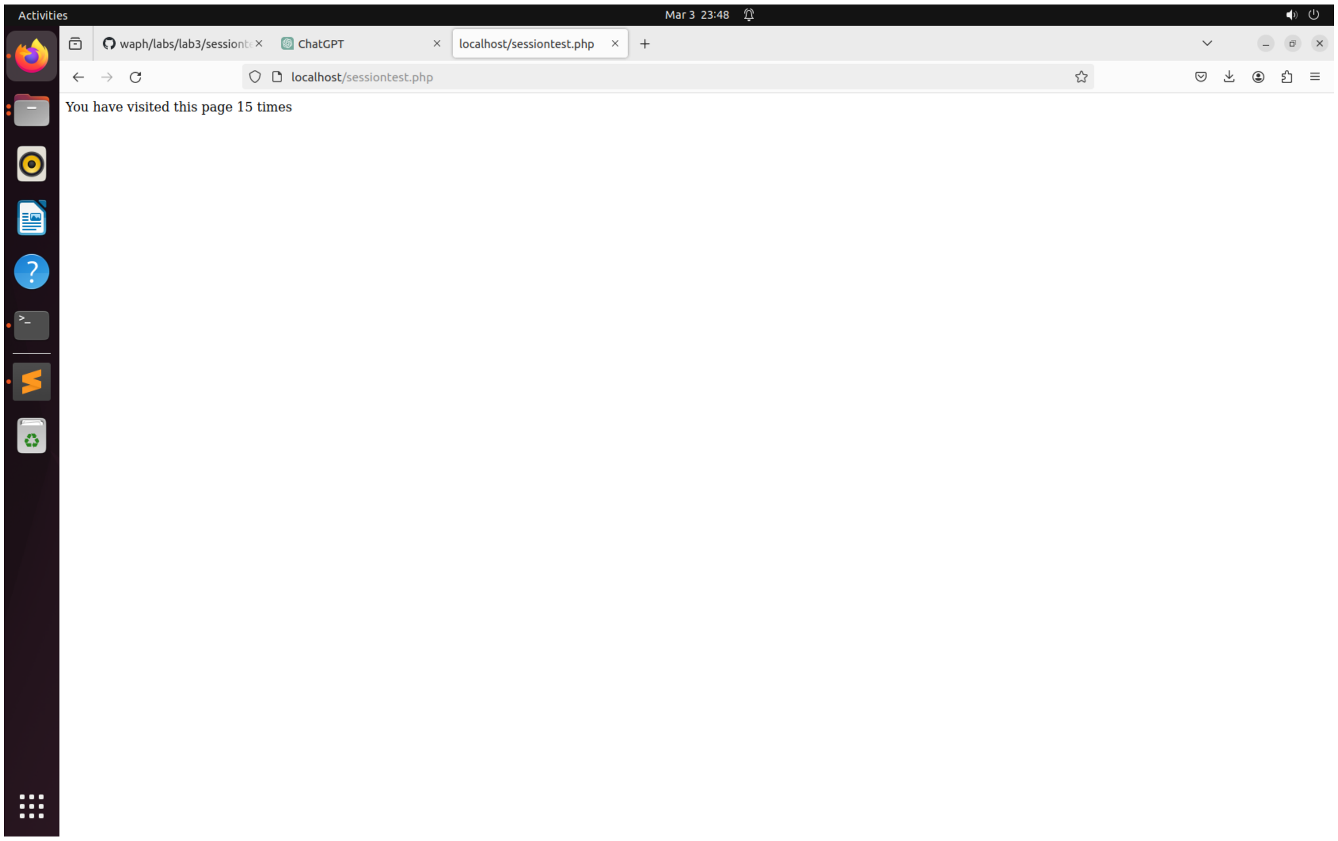
This lab offers a comprehensive investigation into PHP web application session management and security, along with practical exercises. Through the implementation, monitoring, and preservation of session-based authentication protocols, I acquired pragmatic understandings of crucial security protocols. I learned about the inner workings of session handshaking procedures by first examining the fundamentals of sessions using tools such as Wireshark. The exercises that follow focus on locating and fixing problems related to insecure session authentication. In order to prevent possible session hijacking attempts, these actions culminate in the deployment of strong security measures, such as the adoption of HTTPS and the application of certain session cookie settings, including the HttpOnly and Secure flags.

Hands-on exercise that compares browser information saved in session data in order to identify session hijacking. I got an understanding of the intricacies of session management through these assignments, as well as the value of using safe authentication procedures and putting in place strong security countermeasures to protect web apps from frequent attack vectors.

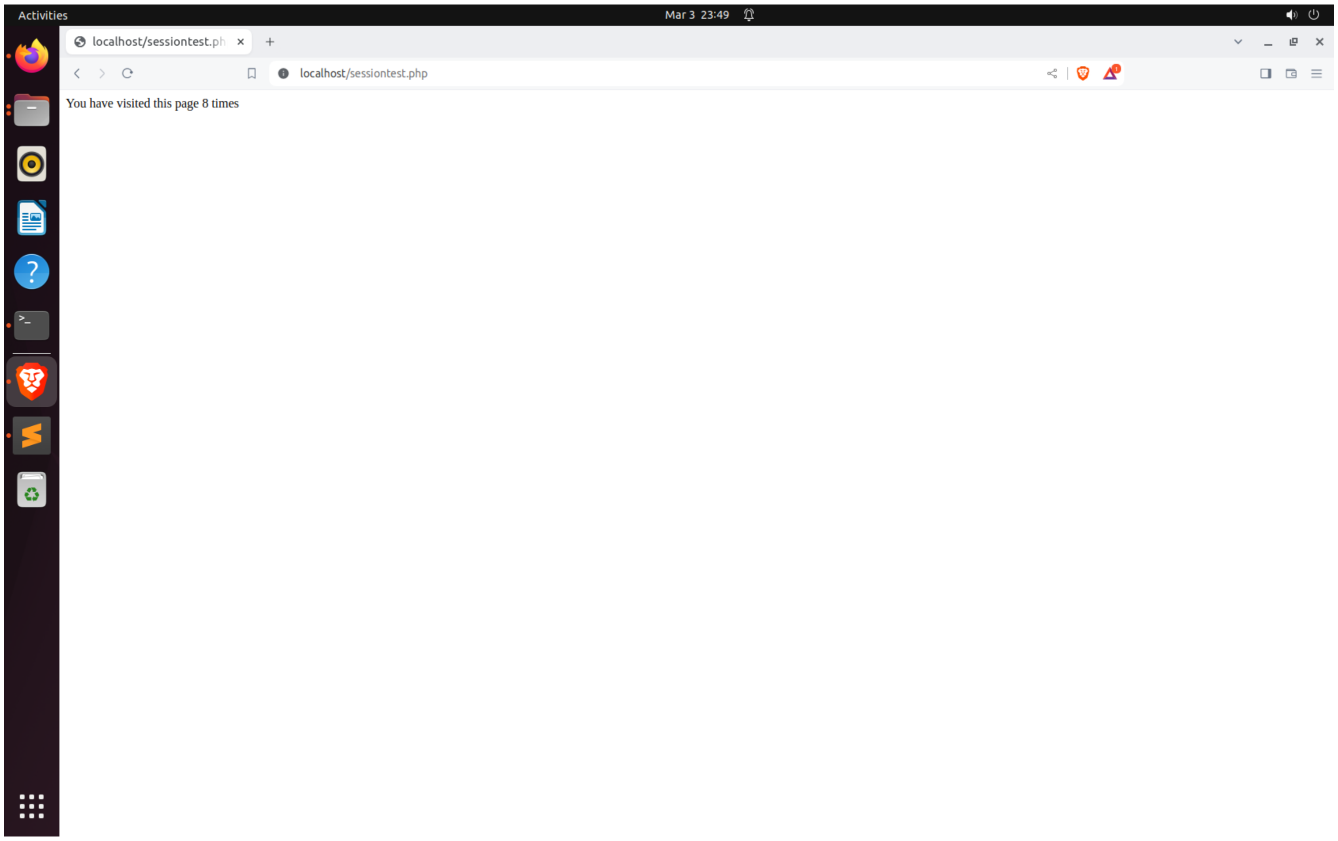
Task 1: Understanding Session Management in a PHP Web Application

1.a. Deploy and test `sessiontest.php`

Here I created a new php page which caputured the session of the user and displayed then the numbers of times that they have visited the page, I used two different browsers and accessed the same page.



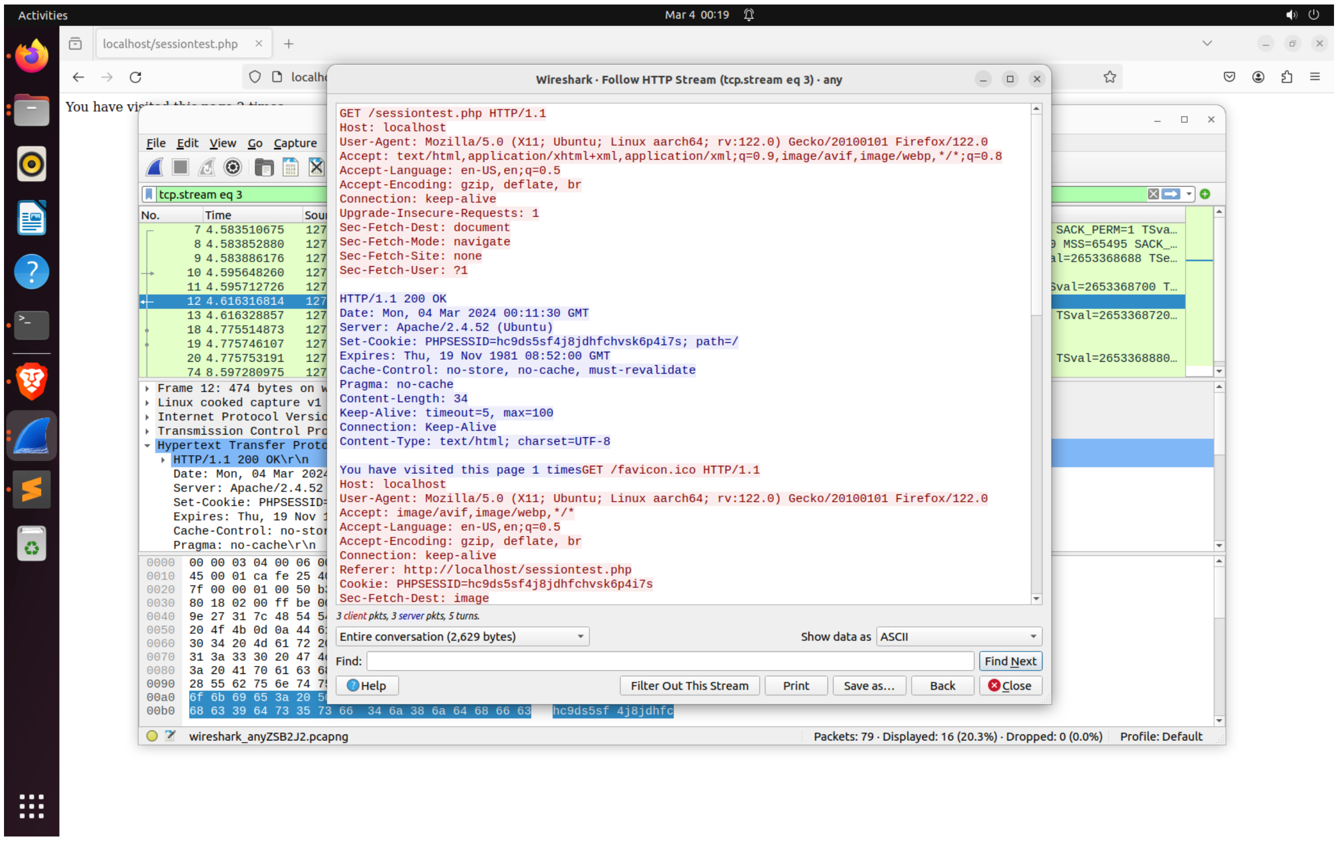
You have visited this page 15 times



You have visited this page 8 times

1.b. Observe the Session-Handshaking using Wireshark

Using wireshark I captured the packets of the session handshaking, I first ran the wireshark and then opened the session page in browser and then reloaded the page, then I stopped the wireshark and saw the captured http packets and saw the request send and the response recived one where the cookie is set and another where the cookie is dispayed with the response showed to the user.

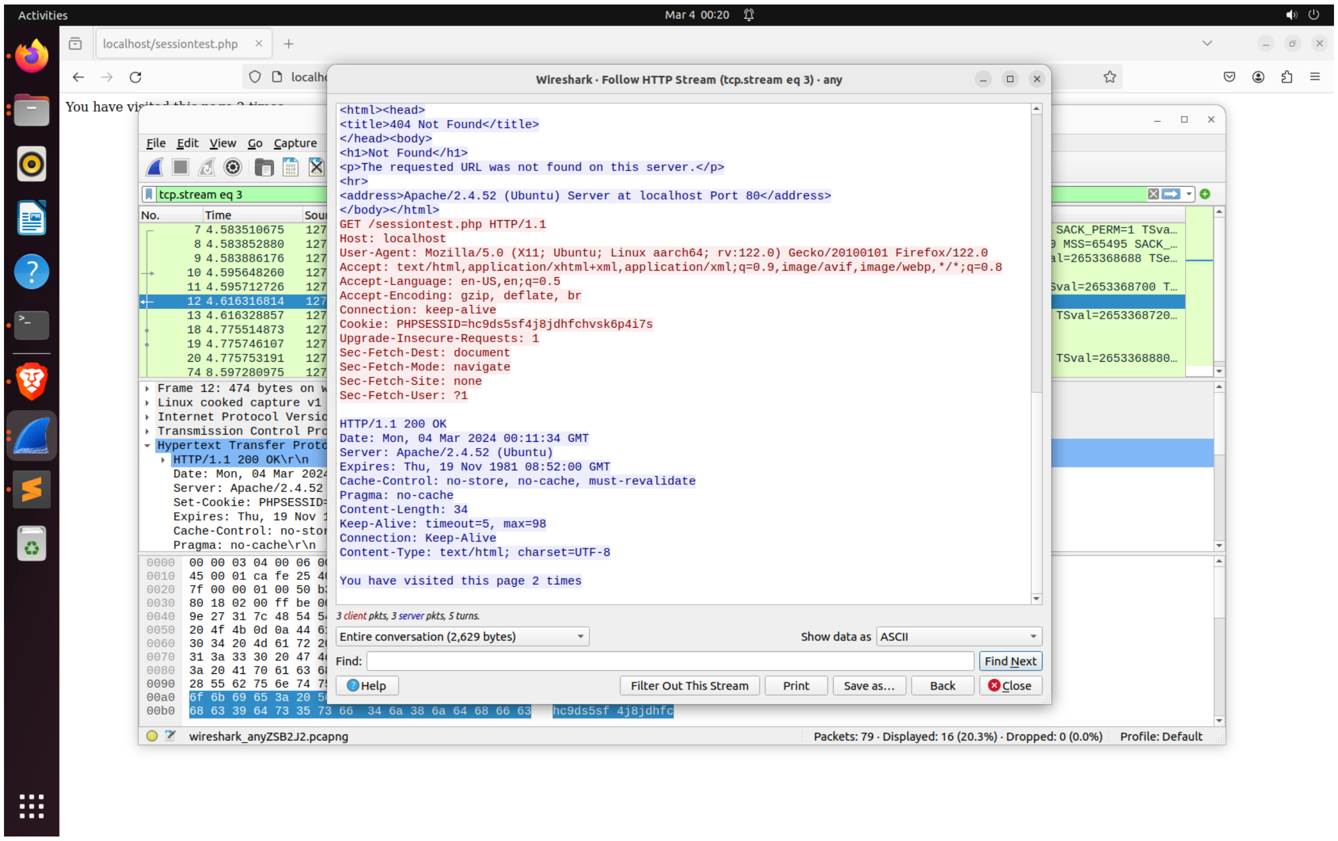

Activities
Mar 4 00:19
localhost/sessiontest.php
Wireshark · Follow HTTP Stream (tcp.stream eq 3) · any
File Edit View Go Capture
tcp.stream eq 3
3 client pkts, 3 server pkts, 5 turns.
Entire conversation (2,629 bytes)
Show data as ASCII
Find:
Find Next
Help
Filter Out This Stream
Print
Save as…
Back
Close
wireshark_anyZSB2J2.pcapng
Packets: 79 · Displayed: 16 (20.3%) · Dropped: 0 (0.0%)
Profile: Default

```
GET /sessiontest.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux aarch64; rv:122.0) Gecko/20100101 Firefox/122.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1

HTTP/1.1 200 OK
Date: Mon, 04 Mar 2024 00:11:30 GMT
Server: Apache/2.4.52 (Ubuntu)
Set-Cookie: PHPSESSID=hc9ds5sf4j8jdhfchvsk6p4i7s; path=/
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 1 timesGET /favicon.ico HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux aarch64; rv:122.0) Gecko/20100101 Firefox/122.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://localhost/sessiontest.php
Cookie: PHPSESSID=hc9ds5sf4j8jdhfchvsk6p4i7s
Sec-Fetch-Dest: image
```

```
No.  Time
   7 4.583510675  127
   8 4.583852880  127
   9 4.583886176  127
  10 4.595648260  127
  11 4.595712726  127
  12 4.616316814  127
  13 4.616328857  127
  18 4.775514873  127
  19 4.775746107  127
  20 4.775753191  127
  74 8.597280975  127
```

```
Frame 12: 474 bytes on w
Linux cooked capture v1
Internet Protocol Versio
Transmission Control Pro
Hypertext Transfer Proto
  HTTP/1.1 200 OK\r\n
  Date: Mon, 04 Mar 2024
  Server: Apache/2.4.52
  Set-Cookie: PHPSESSID=
  Expires: Thu, 19 Nov 1
  Cache-Control: no-stor
  Pragma: no-cache\r\n
```

```
0000  00 00 03 04 00 06 0
0010  45 00 01 ca fe 25 4
0020  7f 00 00 01 00 50 b
0030  80 18 02 00 ff be 0
0040  9e 27 31 7c 48 54 5
0050  20 4f 4b 0d 0a 44 6
0060  30 34 20 4d 61 72 2
0070  31 3a 33 30 20 47 4
0080  3a 20 41 70 61 63 6
0090  28 55 62 75 6e 74 7
00a0  6f 6b 69 65 3a 20 5
00b0  68 63 39 64 73 35 73 66  34 6a 38 6a 64 68 66 63   hc9ds5sf 4j8jdhfc
```



Wireshark · Follow HTTP Stream (tcp.stream eq 3) · any

```
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL was not found on this server.</p>
<hr>
<address>Apache/2.4.52 (Ubuntu) Server at localhost Port 80</address>
</body></html>
GET /sessiontest.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux aarch64; rv:122.0) Gecko/20100101 Firefox/122.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Cookie: PHPSESSID=hc9ds5sf4j8jdhfchvsk6p4i7s
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1

HTTP/1.1 200 OK
Date: Mon, 04 Mar 2024 00:11:34 GMT
Server: Apache/2.4.52 (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=98
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

You have visited this page 2 times
```

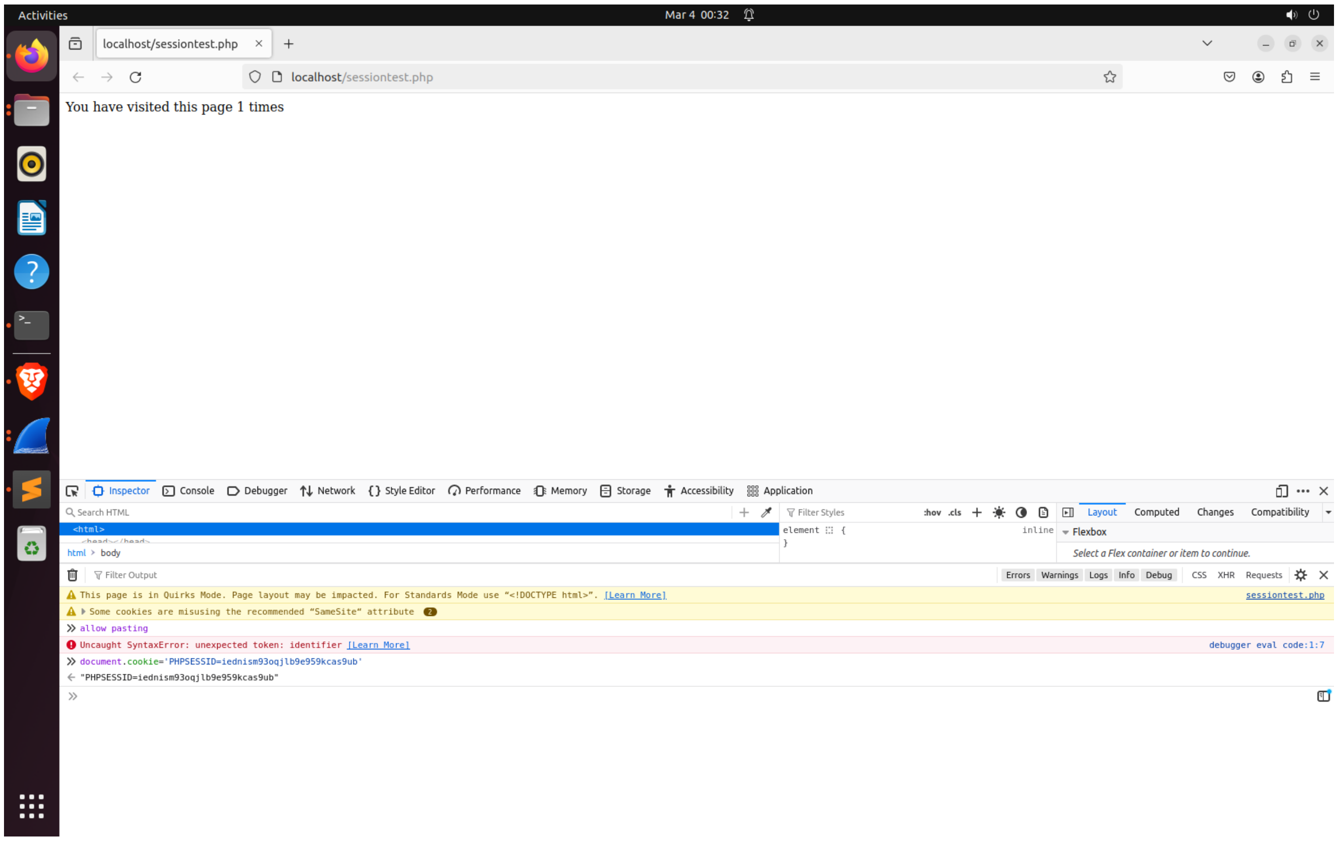
3 client pkts, 3 server pkts, 5 turns.

Entire conversation (2,629 bytes) Show data as ASCII

wireshark_anyZSB2J2.pcapng Packets: 79 · Displayed: 16 (20.3%) · Dropped: 0 (0.0%) Profile: Default

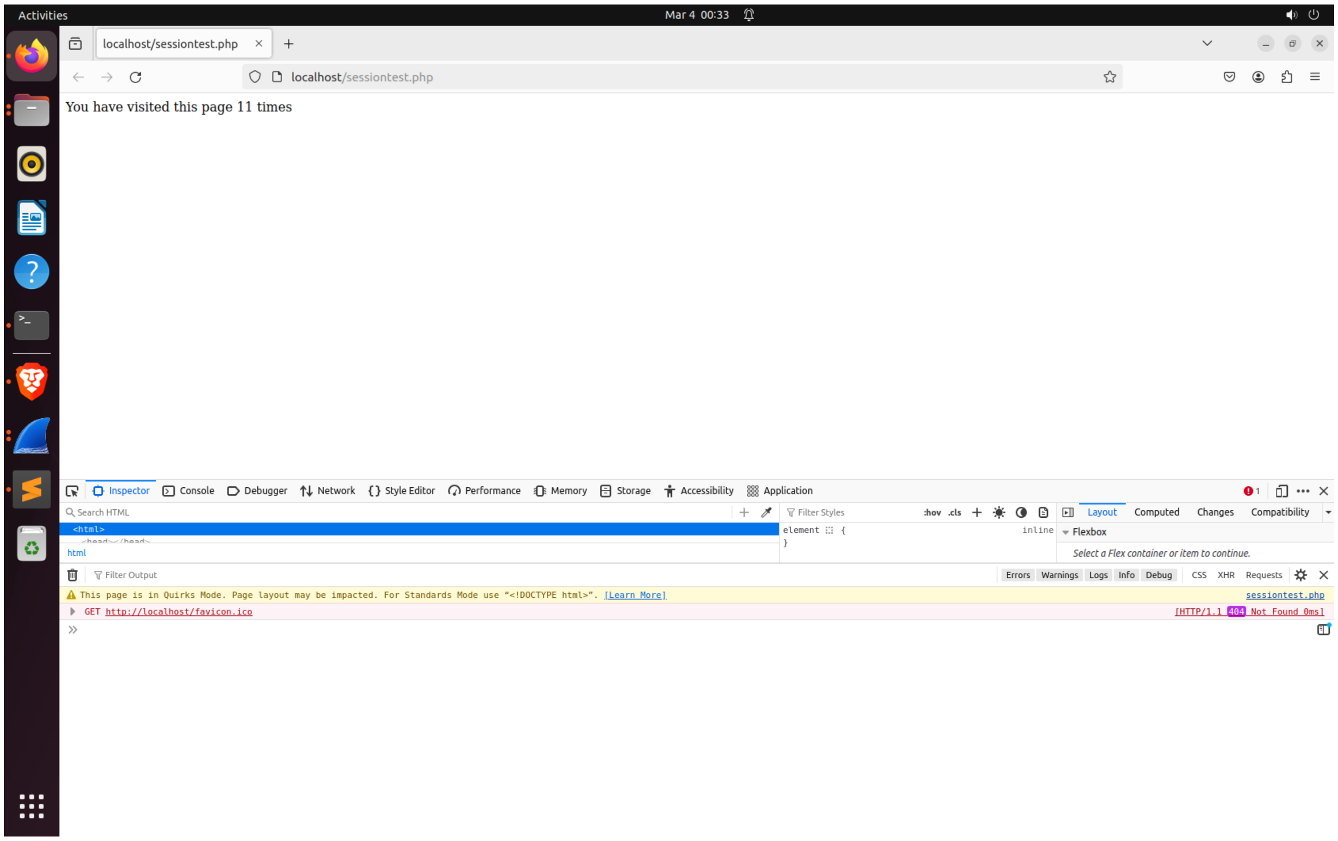
1.c. Understanding Session Hijacking

We can session hijack using the cookies where you can capture /copy the cookie from one browser and the use the same cookie and set the cookie value same and you can now access the page hijack the session which was displayed in the another browser.



```
You have visited this page 1 times
```

```
⚠ This page is in Quirks Mode. Page layout may be impacted. For Standards Mode use "<!DOCTYPE html>". [Learn More]    sessiontest.php
⚠ ▸ Some cookies are misusing the recommended "SameSite" attribute 2
>> allow pasting
❗ Uncaught SyntaxError: unexpected token: identifier [Learn More]    debugger eval code:1:7
>> document.cookie='PHPSESSID=iednism93oqjlb9e959kcas9ub'
← "PHPSESSID=iednism93oqjlb9e959kcas9ub"
```



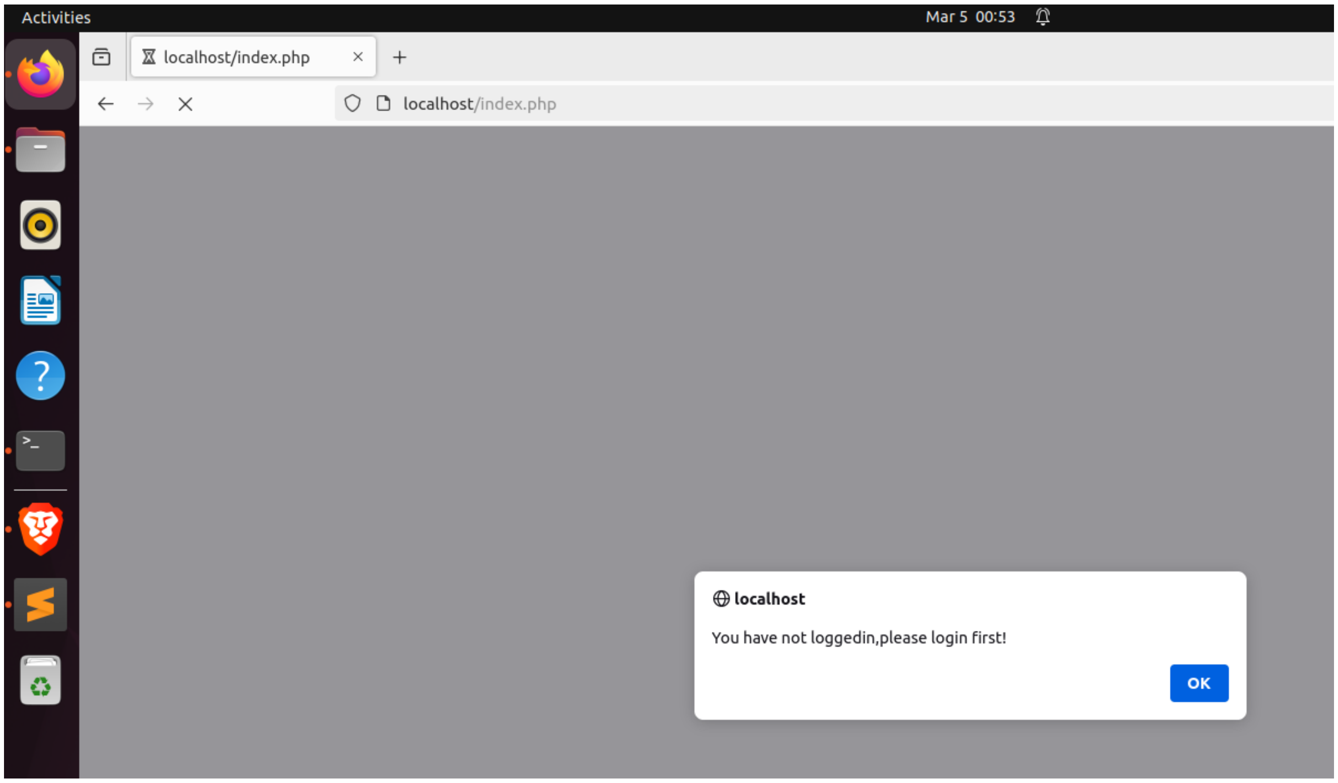
```
You have visited this page 11 times
```

```
⚠ This page is in Quirks Mode. Page layout may be impacted. For Standards Mode use "<!DOCTYPE html>". [Learn More]    sessiontest.php
▸ GET http://localhost/favicon.ico    [HTTP/1.1 404 Not Found 0ms]
```

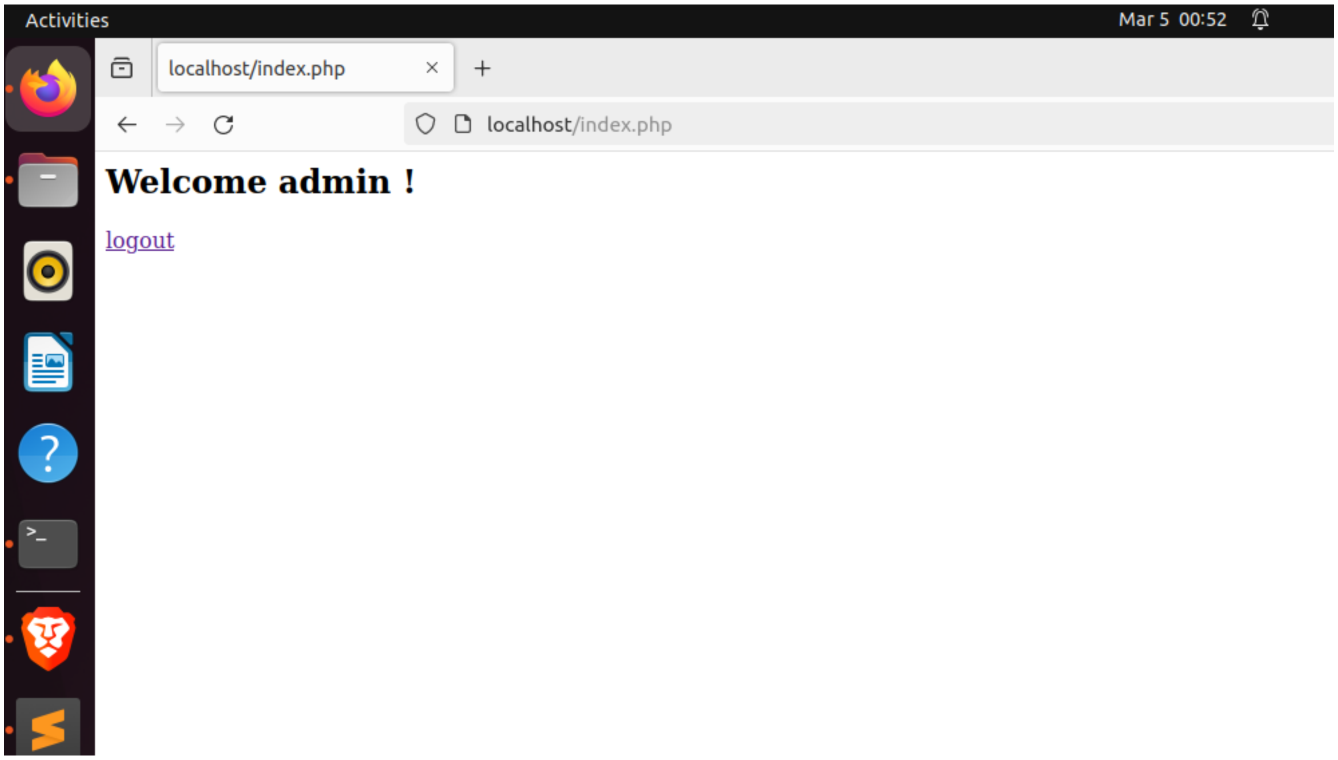
Task 2: Insecure Session Authentication

2.a. Revised Login System with Session Management

Here I modified the `index.php` page and impletemtn the session managment where we created session and also a new link which leads to the `logout.php` page, in the session managment I created in such a way that an unauthenticated user when trys to access `index.php` page gets an alert



You have not loggedin,please login first!

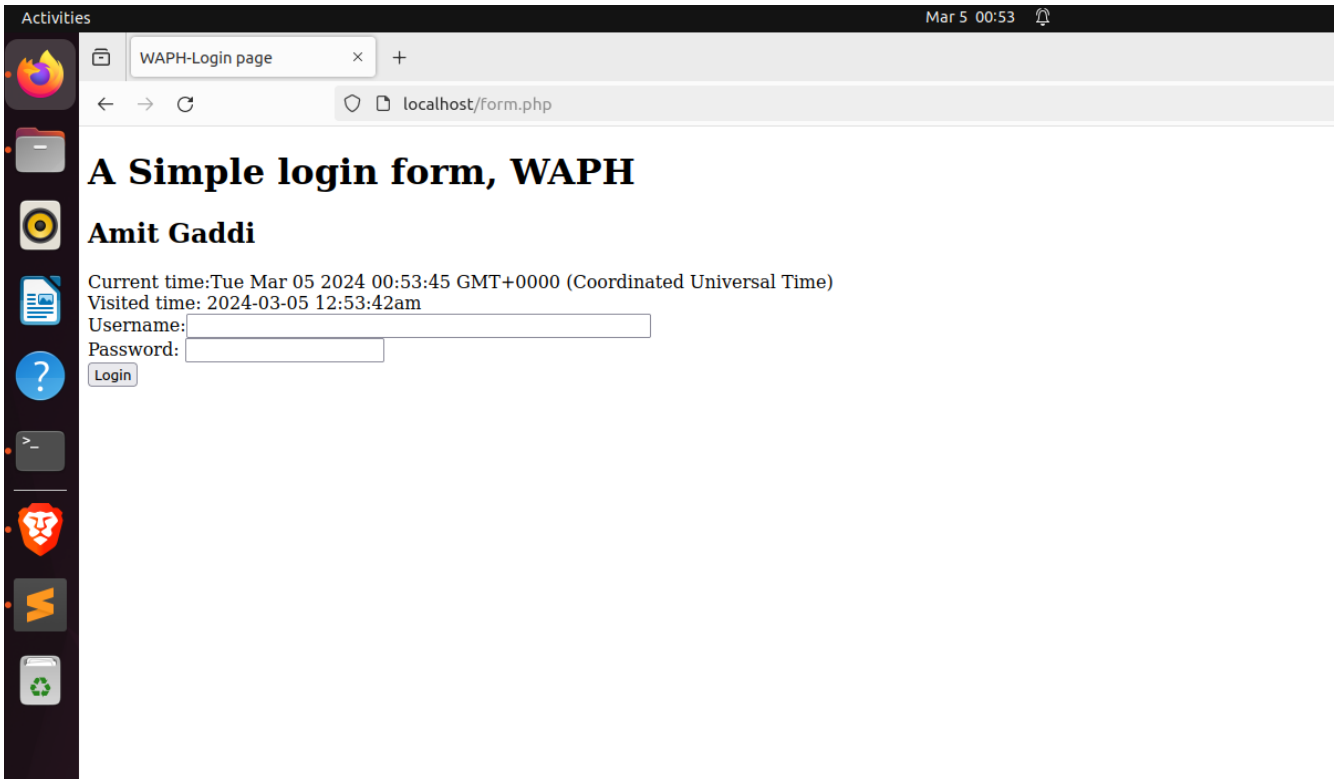


Welcome admin !

logout

2.b. Session Hijacking Attacks

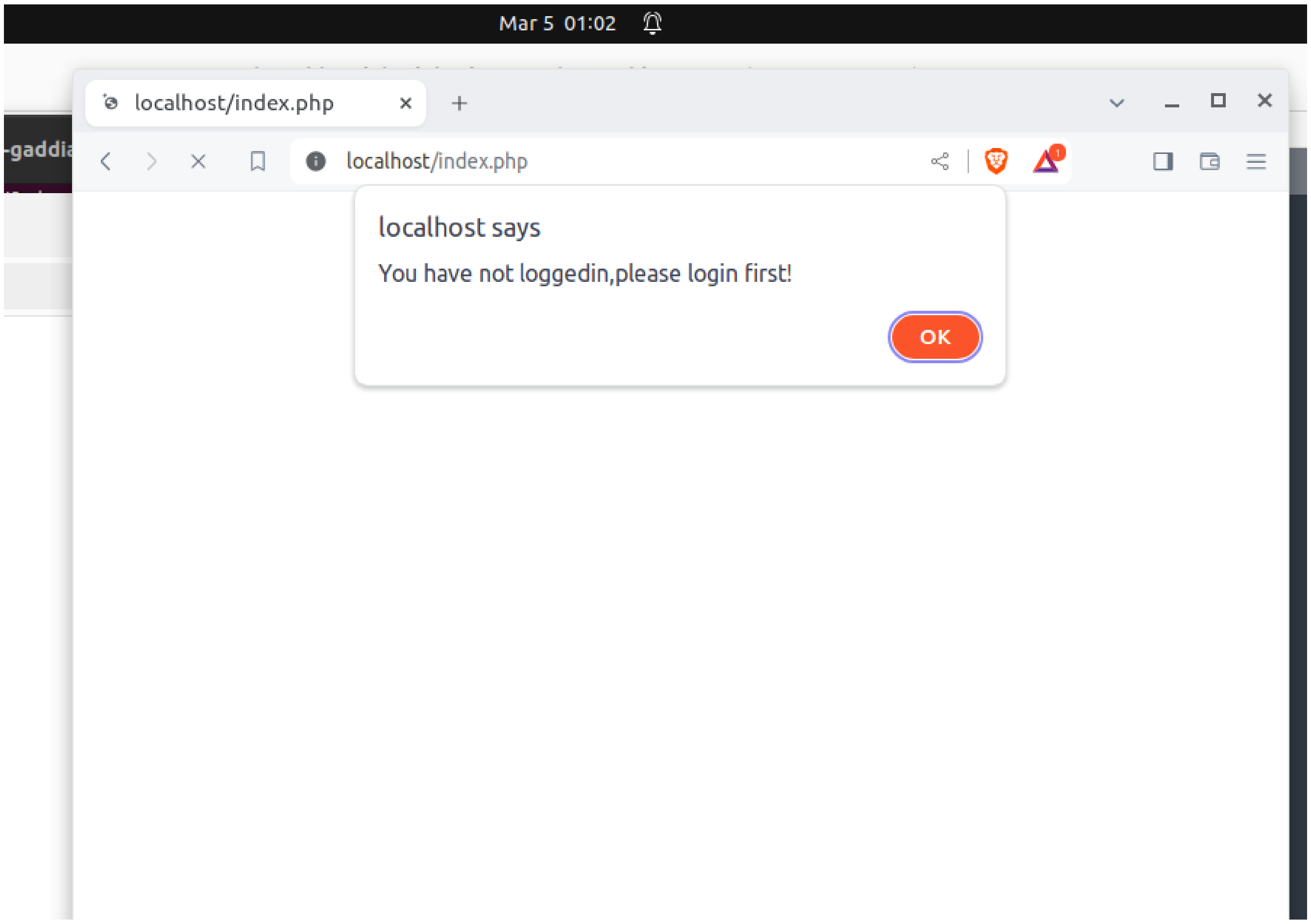
Here we used the cookie to hijack the session from one browsert to another, as seen in the screenshots, I captured the cookie from one browser and set the same cookie value in another browser to got access to the page without entering the credentials.



A Simple login form, WAPH

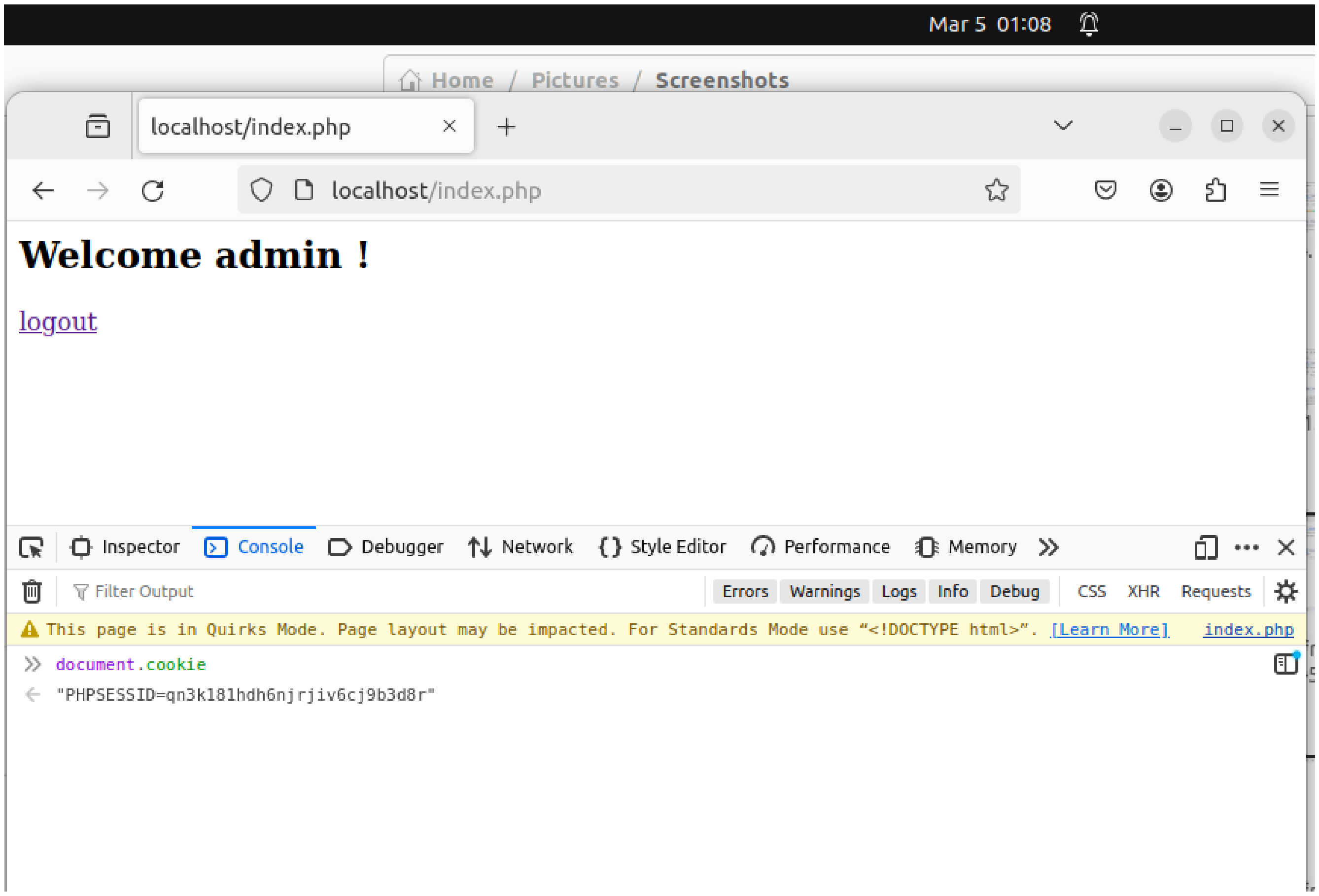
Amit Gaddi

Current time:Tue Mar 05 2024 00:53:45 GMT+0000 (Coordinated Universal Time)
Visited time: 2024-03-05 12:53:42am
Username:
Password:



localhost says

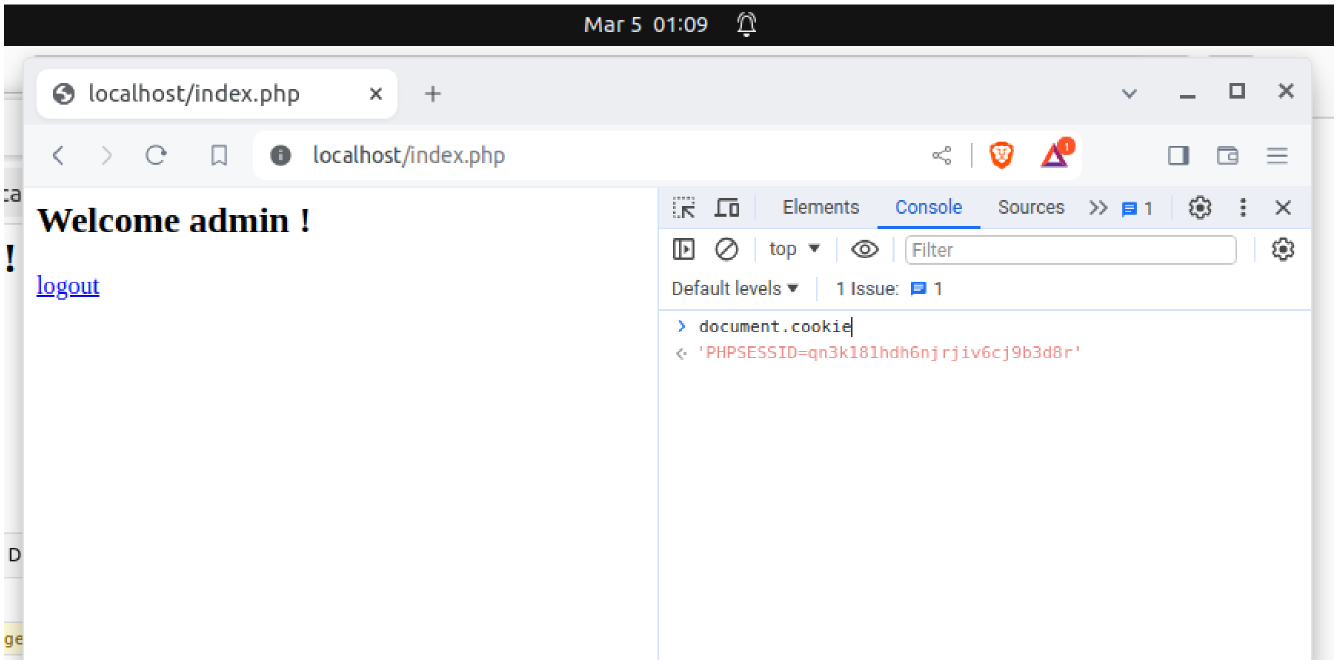
You have not loggedin,please login first!



Welcome admin !

logout

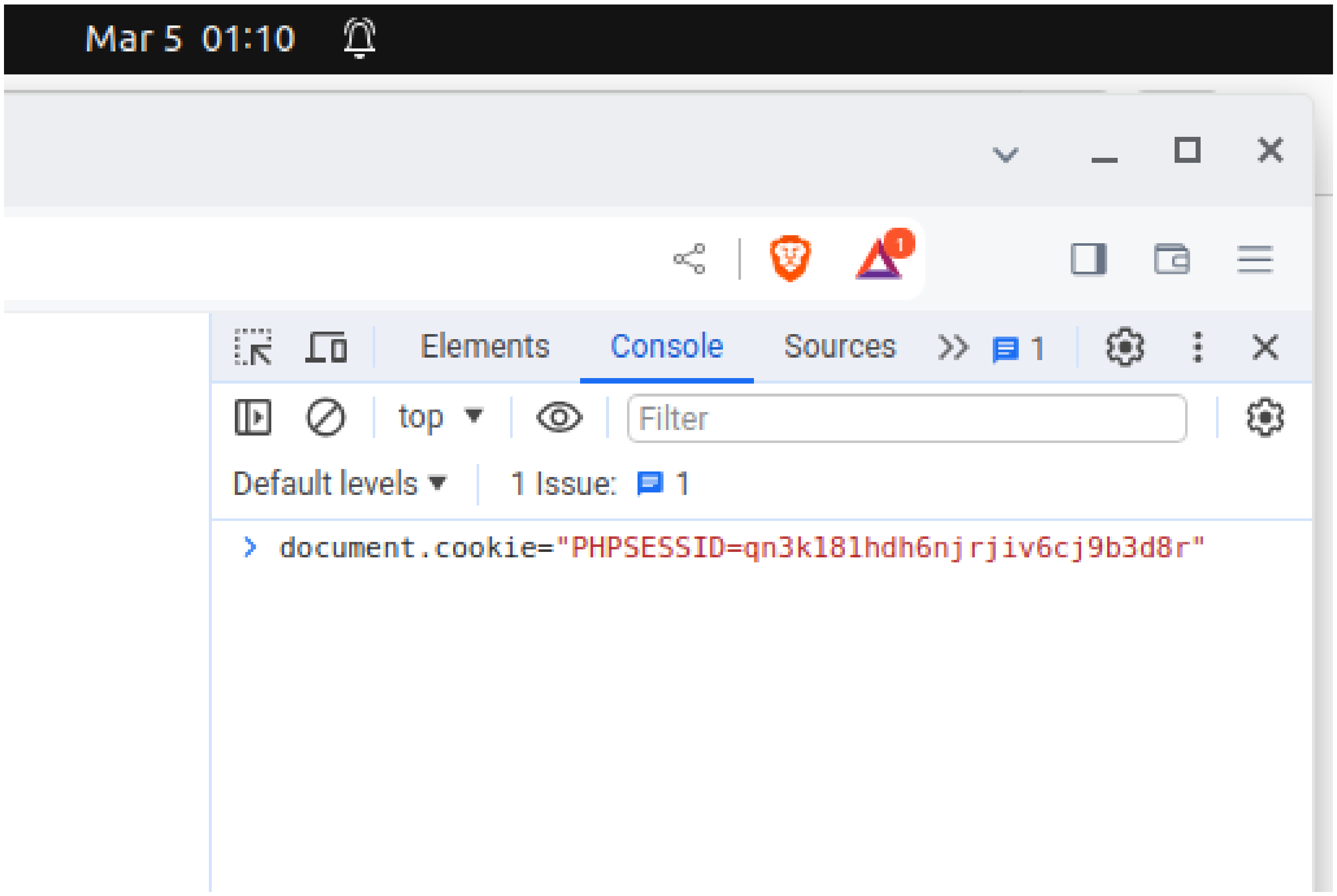
```
⚠ This page is in Quirks Mode. Page layout may be impacted. For Standards Mode use "<!DOCTYPE html>". [Learn More]   index.php
>> document.cookie
<- "PHPSESSID=qn3k181hdh6njrjiv6cj9b3d8r"
```



Welcome admin !

logout

```
> document.cookie
<- 'PHPSESSID=qn3k181hdh6njrjiv6cj9b3d8r'
```

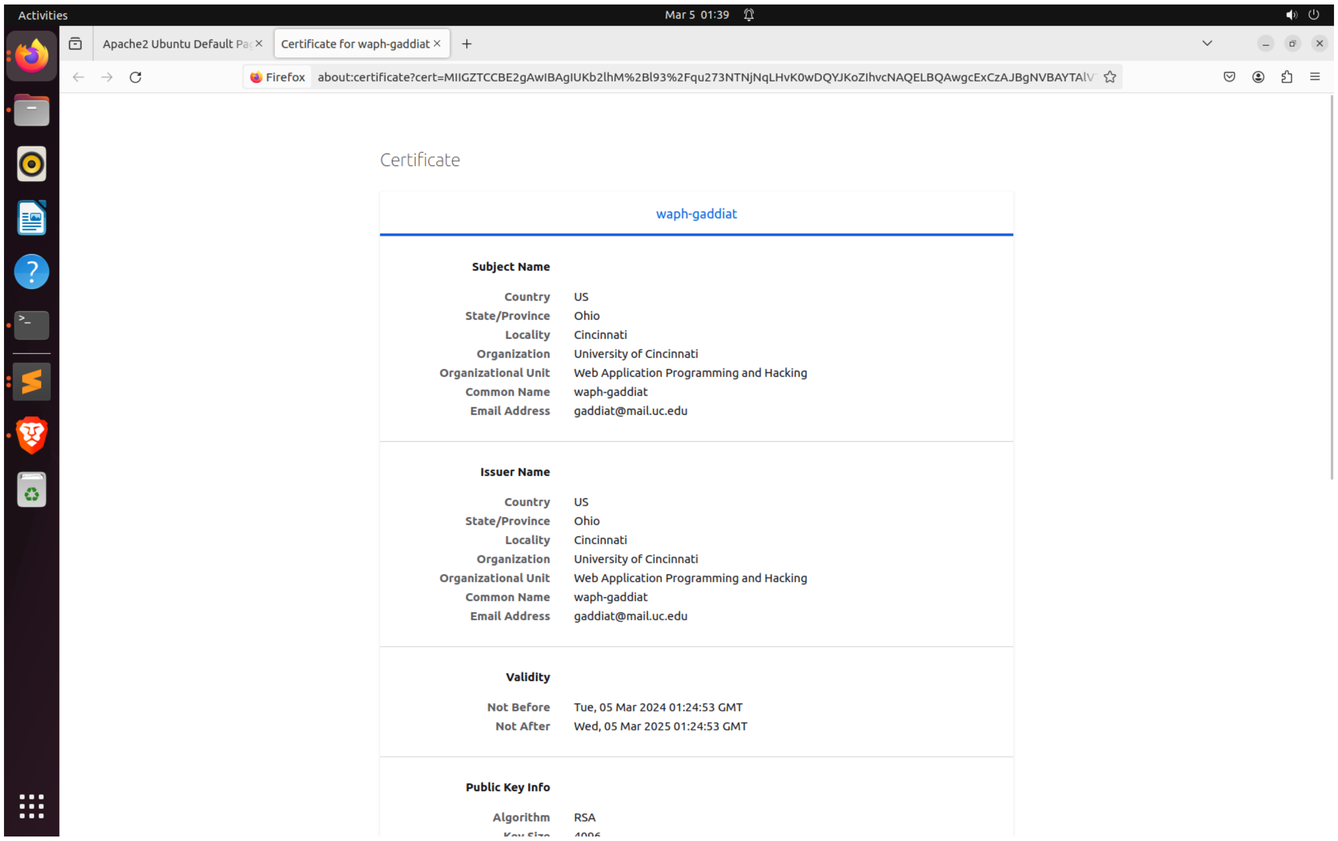


```
> document.cookie="PHPSESSID=qn3k181hdh6njrjiv6cj9b3d8r"
```

Task 3: Securing Session and Session Authentication

3.a. Data Protection and HTTPS Setup

Here we generated a key and stored in apache and configured the ssl file and then accessed the web page with HTTPS and as the certificate is not valid, I have to add an exception and below screenshot is the certificate that is showed in browser, moreover we can access our page with https too.


Activities
Mar 5 01:39
Apache2 Ubuntu Default Pa
Certificate for waph-gaddiat
Firefox
about:certificate?cert=MIIGZTCCBE2gAwIBAgIUKb2lhM%2Bl93%2Fqu273NTNjNqLHvK0wDQYJKoZIhvcNAQELBQAwgcExCzAJBgNVBAYTAlV

Certificate

waph-gaddiat

Subject Name

Country	US
State/Province	Ohio
Locality	Cincinnati
Organization	University of Cincinnati
Organizational Unit	Web Application Programming and Hacking
Common Name	waph-gaddiat
Email Address	gaddiat@mail.uc.edu

Issuer Name

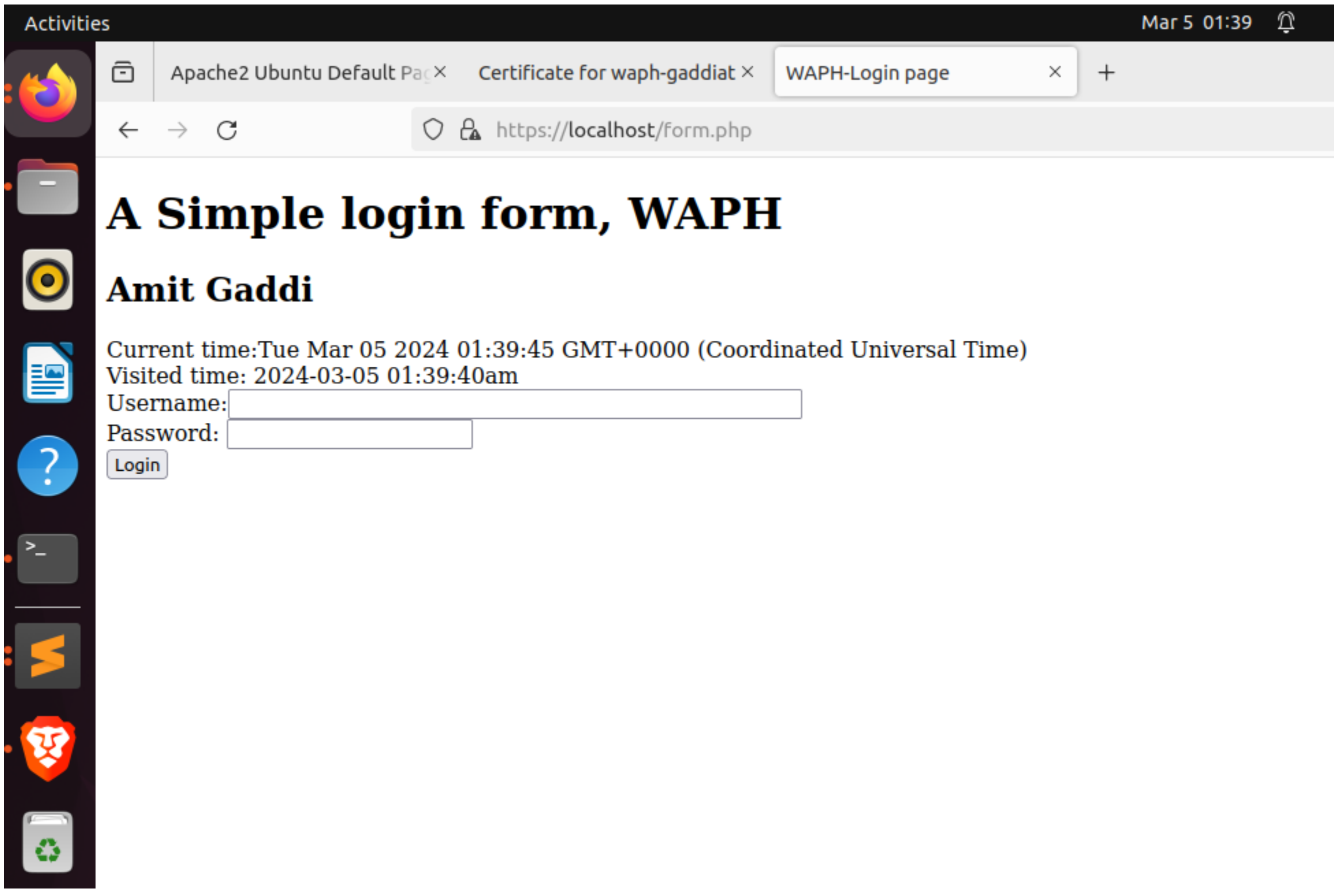
Country	US
State/Province	Ohio
Locality	Cincinnati
Organization	University of Cincinnati
Organizational Unit	Web Application Programming and Hacking
Common Name	waph-gaddiat
Email Address	gaddiat@mail.uc.edu

Validity

Not Before	Tue, 05 Mar 2024 01:24:53 GMT
Not After	Wed, 05 Mar 2025 01:24:53 GMT

Public Key Info

Algorithm	RSA


Activities
Mar 5 01:39
Apache2 Ubuntu Default Pa
Certificate for waph-gaddiat
WAPH-Login page
https://localhost/form.php

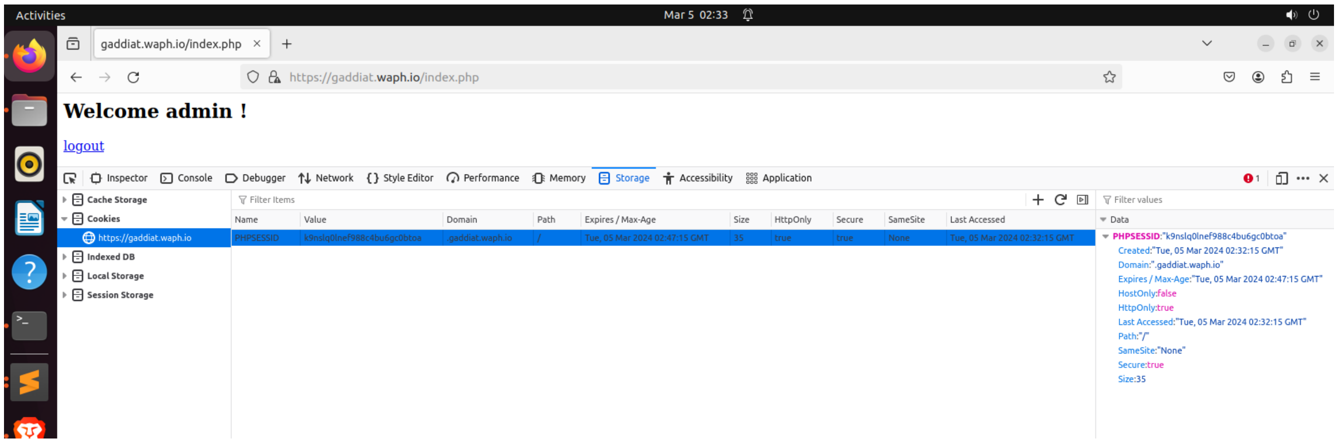
A Simple login form, WAPH

Amit Gaddi

Current time:Tue Mar 05 2024 01:39:45 GMT+0000 (Coordinated Universal Time)
Visited time: 2024-03-05 01:39:40am
Username:
Password:

3.b. Securing Session Against Session Hijacking Attacks - setting HttpOnly and Secure flags for cookies

Here in the `index.php` I added a code to set the parameters of the cookie before starting the sessuin, here the values set were - time, path, domain, httponly and secure. after deployin the code you can now see as seen in the below screenshot the value is set to ture and now the cookie id cannot be accessed.

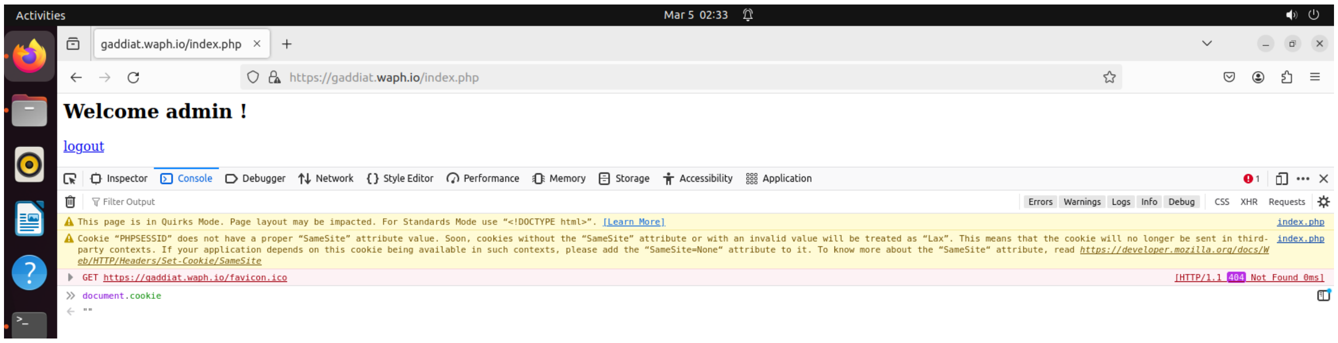


Welcome admin !

logout

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
PHPSESSID	k9nslq0lnef988c4bu6gc0btoa	.gaddiat.waph.io	/	Tue, 05 Mar 2024 02:47:15 GMT	35	true	true	None	Tue, 05 Mar 2024 02:32:15 GMT

```
PHPSESSID:"k9nslq0lnef988c4bu6gc0btoa"
  Created:"Tue, 05 Mar 2024 02:32:15 GMT"
  Domain:".gaddiat.waph.io"
  Expires / Max-Age:"Tue, 05 Mar 2024 02:47:15 GMT"
  HostOnly:false
  HttpOnly:true
  Last Accessed:"Tue, 05 Mar 2024 02:32:15 GMT"
  Path:"/"
  SameSite:"None"
  Secure:true
  Size:35
```



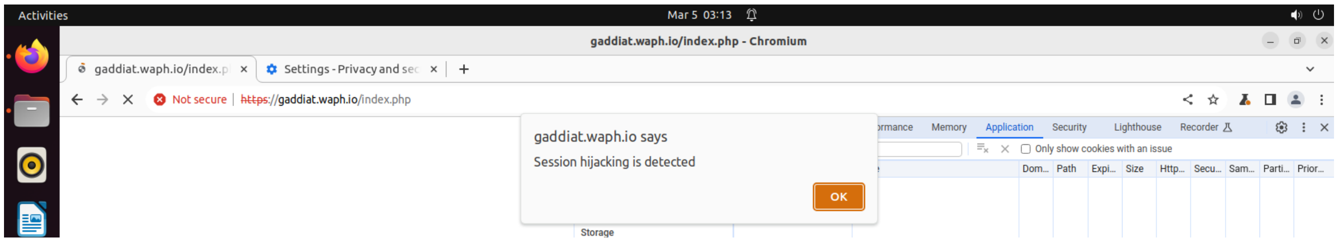
Welcome admin !

logout

```
This page is in Quirks Mode. Page layout may be impacted. For Standards Mode use "<!DOCTYPE html>". [Learn More]   index.php
Cookie "PHPSESSID" does not have a proper "SameSite" attribute value. Soon, cookies without the "SameSite" attribute or with an invalid value will be treated as "Lax". This means that the cookie will no longer be sent in third-party contexts. If your application depends on this cookie being available in such contexts, please add the "SameSite=None" attribute to it. To know more about the "SameSite" attribute, read https://developer.mozilla.org/docs/Web/HTTP/Headers/Set-Cookie/SameSite   index.php
GET https://gaddiat.waph.io/favicon.ico   [HTTP/1.1 404 Not Found 0ms]
>> document.cookie
<- ""
```

3.c. Securing Session Against Session Hijacking Attacks - Defense In-Depth

After modifying the code to add the new protectiion layer, now even if we try to access the page with the another browser we would not be able to access the page even if we have the cookie id as here we are using browser identification.



gaddiat.waph.io says

Session hijacking is detected

OK