Lifestyle Store - Project Web Application

Detailed Developer Report

Security Status – Extremely Vulnerable

- Hacker can steal all records in Internshala databases (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 Internshala (XSS)
- Hacker can execute any command to extract information from website and deface it.
- Hacker can easily view default page and debug the pages, can easily guess the default password and can exploit vulnerability.

Vulnerability Statistics

Critical	Severe	Moderate
13	11	4

Low
6

Vulnerabilities:

No	Severity	Vulnerability	Count
1	Critical	SQL Injection	3
2	Sever	Reflected and Stored Cross Site Scripting	2
3	Sever	Insecure Direct Object Reference	3
4	Critical	Rate Limit Issues	1
5	Critical	Insecure File Update	1
6	Moderate	Client Side Filter bypass	1
7	Critical	Component with known Vulnerability	3
8	Critical	Default Admin Password	1
9	Low	Descriptive Error Messages	1
10	Low	Default File and Folder	5

1. SQL Injection

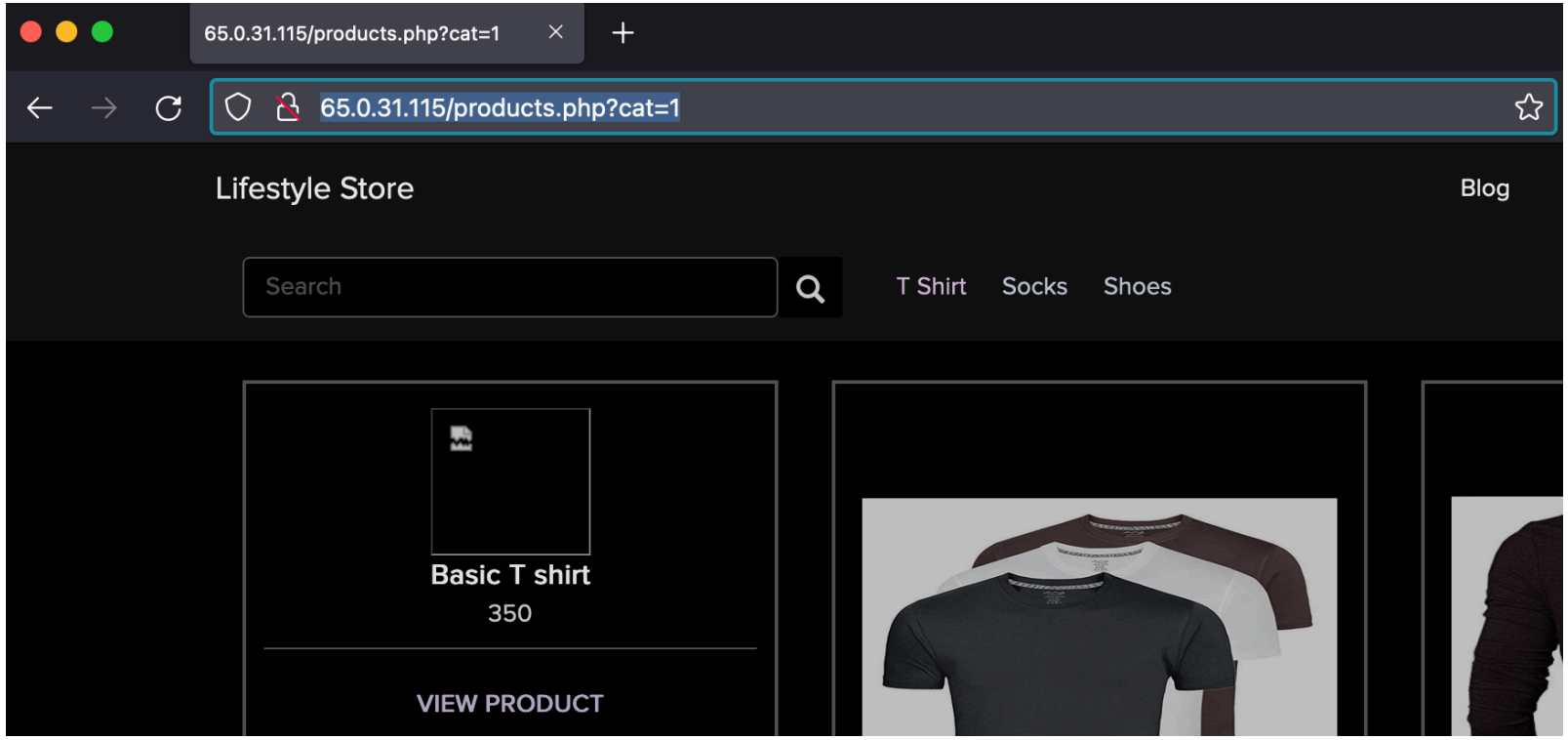
SQL Injection (Critical)	Below mentioned URL in the E-Commerce Website is vulnerable to SQL injection attack **Affected URL :** • http://65.0.31.115/products.php?cat=1 **Affected Parameters :** • cat (GET parameter) **Payload:** • cat=1’

1. SQL Injection

SQL Injection (Critical)	Here are other similar SQLi in the application **Affected URL :** • http://65.0.31.115/products.php?cat=2 • http://65.0.31.115/products.php?cat=3

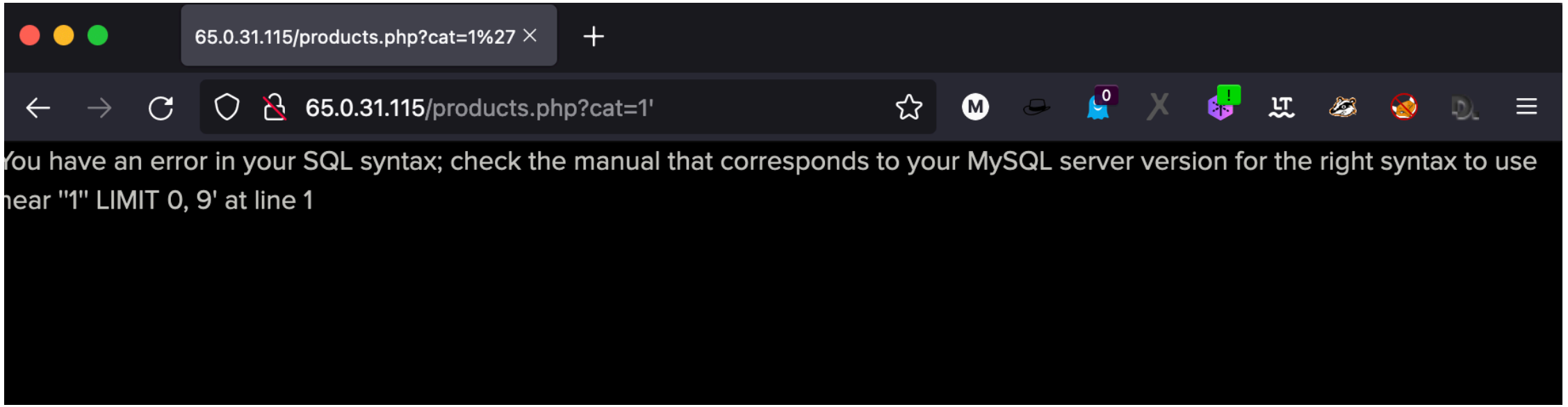
Observation

- Navigate to Home Page of the Website. There you will see category options. Click on “T-Shirt" When you get into this URL, you will see products as per the category you have chosen. Notice the GET parameter **house** in the URL:



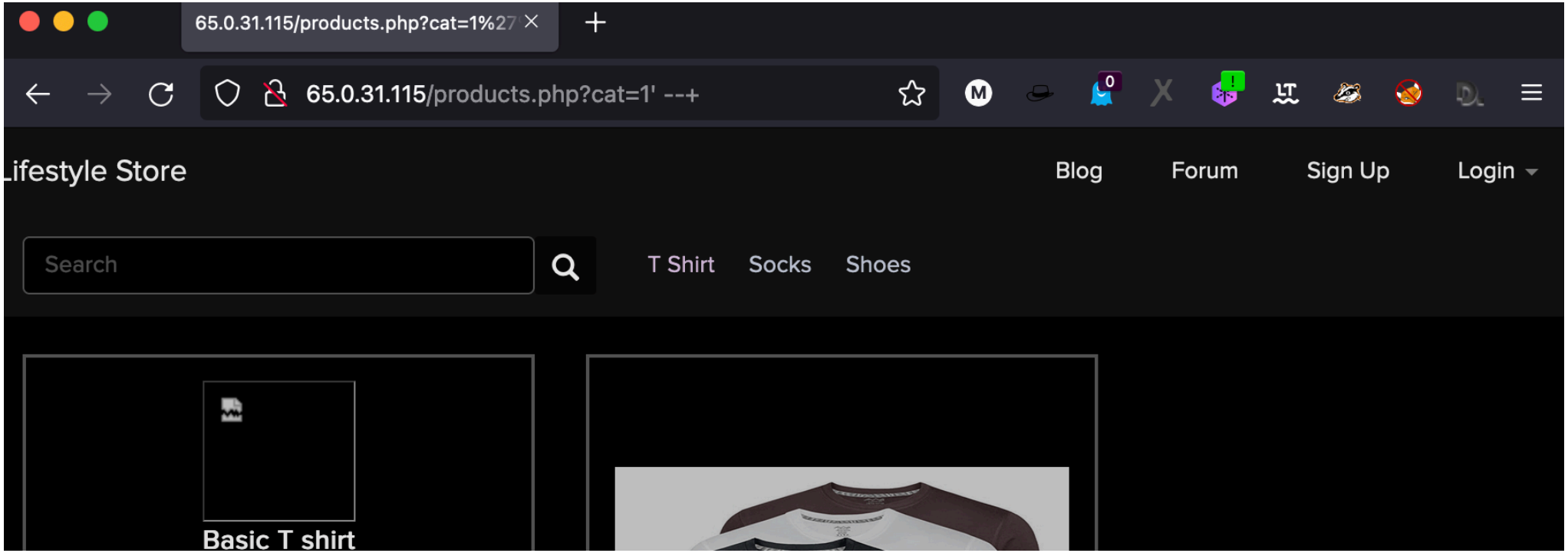
Observation

- Now, we apply a single quote in the category parameter (i.e., GET parameter): **65.0.31.115/products.php?cat=1’**, and we get a complete MySQL error.



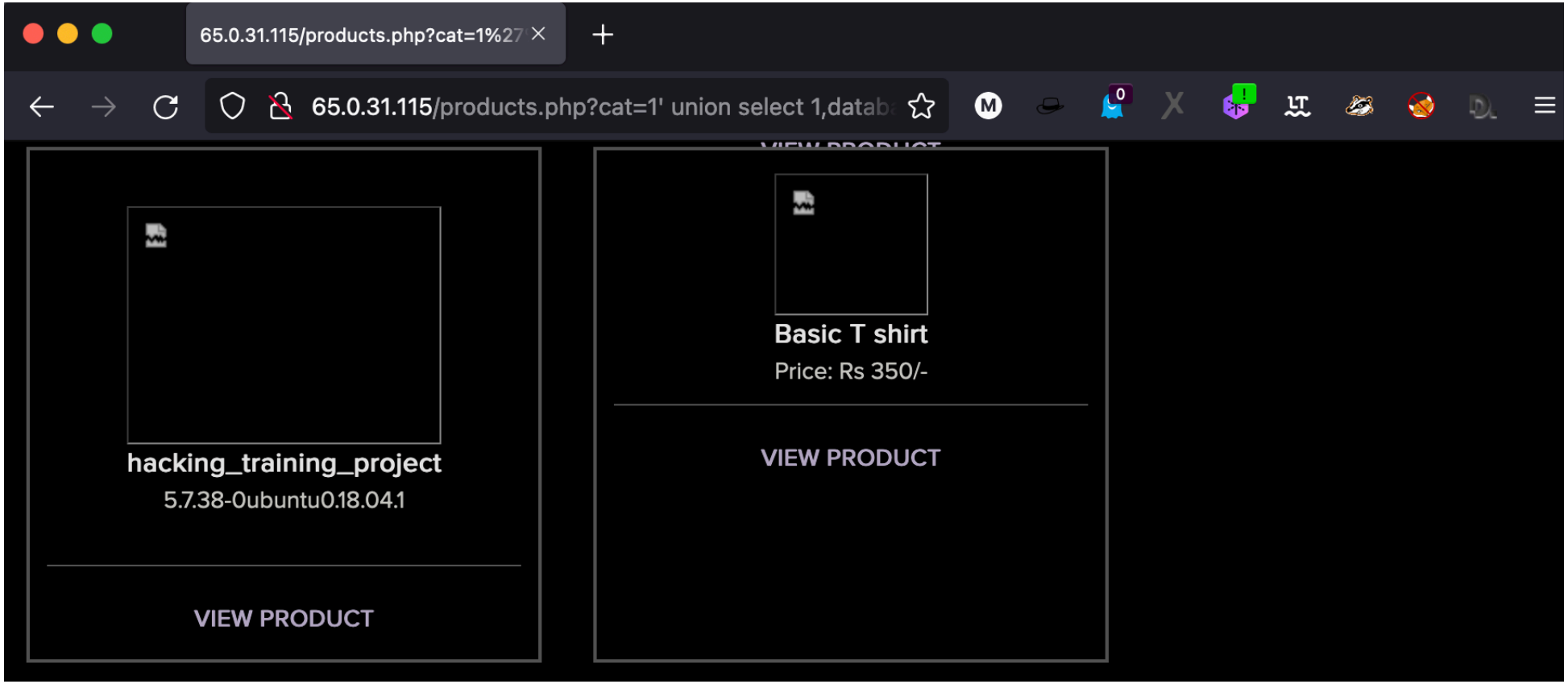
Observation

- We then put --+ : **http://65.0.31.115/products.php?cat=1'--+ and we 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://65.0.31.115/products.php?cat=1’ union select 1,database(),version()3,4,5,—+



PoC – Attacker can dump arbitrary data

- No of databases: 2
  - hacking_training_project
  - Information_schema

```
available 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

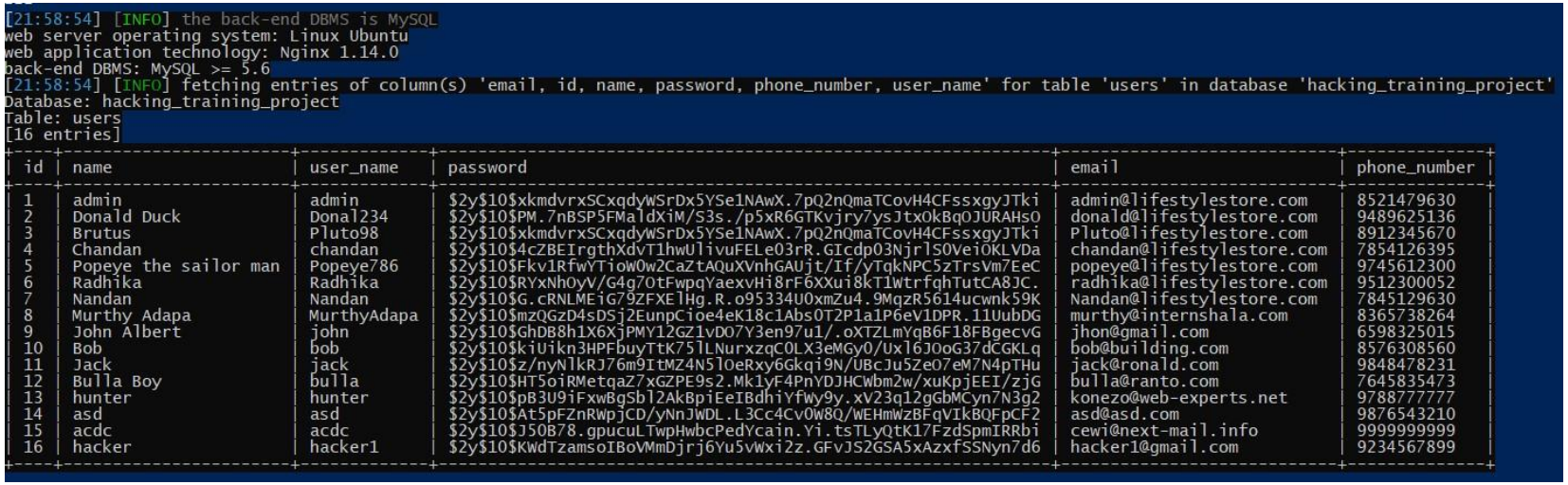
```
Database: hacking_training_project
[10 tables]
+-------------------------------------+
| brands                              |
| cart_items                          |
| categories                          |
| customers                           |
| order_items                         |
| orders 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

Business Impact – Extremely High

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

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

An attacker can use this information to login to admin panels and gain complete admin level access to the website, which could lead to a 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$10$xkmdvrxSCxqdyWSrDx5YSe1NAwX.7pQ2nQmaTCovH4CFssxgyJTki | admin@lifestylestore.com  | 8521479630   |
| 2  | Donald Duck          | Donal234   | $2y$10$PM.7nBSP5FMaldXiM/S3s./p5xR6GTKvjry7ysJtxOkBqOJURAHsO | donald@lifestylestore.com | 9489625136   |
| 3  | Brutus               | Pluto98    | $2y$10$xkmdvrxSCxqdyWSrDx5YSe1NAwX.7pQ2nQmaTCovH4CFssxgyJTki | Pluto@lifestylestore.com  | 8912345670   |
| 4  | Chandan              | chandan    | $2y$10$4cZBEIrgthXdvT1hwUlivuFELe03rR.GIcdpO3NjrlSOVeiOKLVDa | chandan@lifestylestore.com| 7854126395   |
| 5  | Popeye the sailor man| Popeye786  | $2y$10$Fkv1RfwYTioWOw2CaZtAQuXVnhGAUjt/If/yTqkNPC5zTrsVm7EeC | popeye@lifestylestore.com | 9745612300   |
| 6  | Radhika              | Radhika    | $2y$10$RYxNhOyV/G4g7OtFwpqYaexvHi8rF6XXui8kT1WtrfqhTutCA8JC. | radhika@lifestylestore.com| 9512300052   |
| 7  | Nandan               | Nandan     | $2y$10$G.cRNLMEiG79ZFXElHg.R.o95334UOxmZu4.9MqzR5614ucwnk59K | Nandan@lifestylestore.com | 7845129630   |
| 8  | Murthy Adapa         | MurthyAdapa| $2y$10$mzQGzD4sDSj2EunpCioe4eK18c1Abs0T2P1a1P6eV1DPR.11UubDG | murthy@internshala.com    | 8365738264   |
| 9  | John Albert          | john       | $2y$10$GhDB8h1X6XjPMY12GZ1vDO7Y3en97u1/.oXTZLmYqB6F18FBgecvG | jhon@gmail.com            | 6598325015   |
| 10 | Bob                  | bob        | $2y$10$kiUikn3HPFbuyTtK75lLNurxzqCOLX3eMGyO/Ux16JOoG37dCGKLq | bob@building.com          | 8576308560   |
| 11 | Jack                 | jack       | $2y$10$z/nyNlkRJ76m9ItMZ4N5lOeRxy6Gkqi9N/UBcJu5ZeO7eM7N4pTHu | jack@ronald.com           | 9848478231   |
| 12 | Bulla Boy            | bulla      | $2y$10$HT5oiRMetqaZ7xGZPE9s2.Mk1yF4PnYDJHCWbm2w/xuKpjEEI/zjG | bulla@ranto.com           | 7645835473   |
| 13 | hunter               | hunter     | $2y$10$pB3U9iFxwBgSbl2AkBpiEeIBdhiYfWy9y.xV23q12gGbMCyn7N3g2 | konezo@web-experts.net    | 9788777777   |
| 14 | asd                  | asd        | $2y$10$At5pFZnRWpjCD/yNnJWDL.L3Cc4CvOW8Q/WEHmWzBFqVIkBQFpCF2 | asd@asd.com               | 9876543210   |
| 15 | acdc                 | acdc       | $2y$10$J50B78.gpucuLTwpHwbcPedYcain.Yi.tsTLyQtK17FzdSpmIRRbi | cewi@next-mail.info       | 9999999999   |
| 16 | hacker               | hacker1    | $2y$10$KWdTzamsoIBoVMmDjrj6Yu5vWxi2z.GFvJS2GSA5xAzxfSSNyn7d6 | hacker1@gmail.com         | 9234567899   |
+----+----------------------+------------+--------------------------------------------------------------+---------------------------+--------------+
```

Recommendation

Take the following precautions to avoid exploitation of SQL injections:

- Whitelist User Input: Whitelist all user input for expected data only. For example if you are expecting a flower name, limit it to alphabets only upto 20 characters in length. If you are expecting some ID, restrict it to numbers only
- 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 has HTML encoding, URL encoding etc
- Do not store passwords in plain text. Convert them to hashes using SHA1 SHA256 Blowfish 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

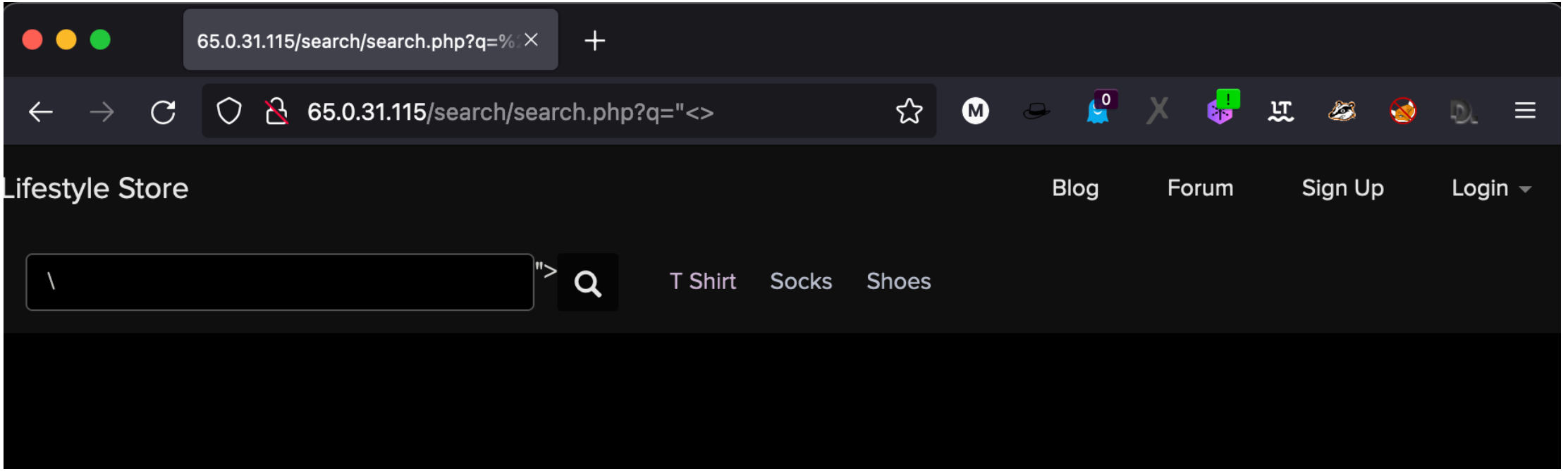
- *https://www.owasp.org/index.php/SQL_Injection*
- *https://en.wikipedia.org/wiki/SQL_injection*

2. Reflected Cross Site Scripting (XSS)

Cross Site Scripting - XSS (Sever)	Below mentioned parameters are vulnerable to Reflected Cross Site Scripting. **Affected URL :** • http://65.0.31.115/search/search.php?q=(here) **Affected Parameters :** • q **Payload:** • '><script>alert(1)</script>

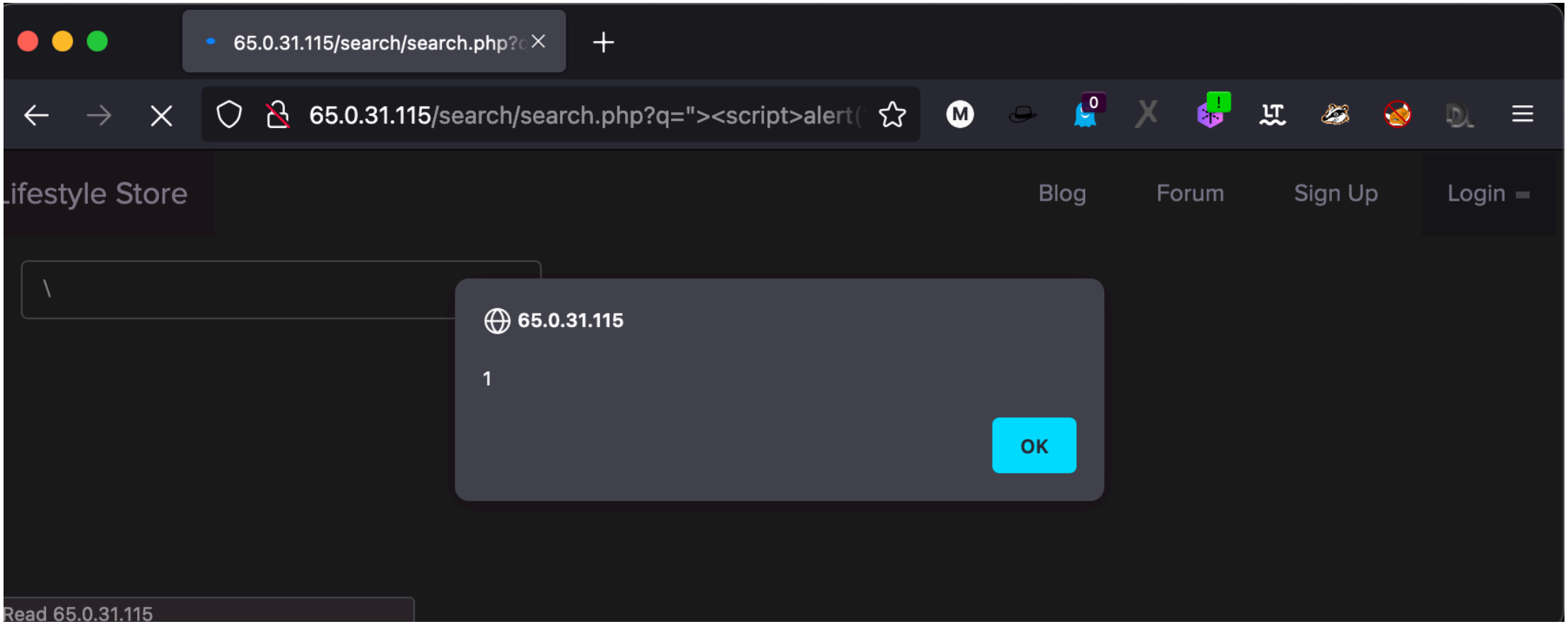
Observation

- Login to your account .
- Next, go to My Cart, click on SHOP NOW button, and enter ">" in the Search Box.
- You will notice that the code is reflected on the website.



PoC - Custom Script executions

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

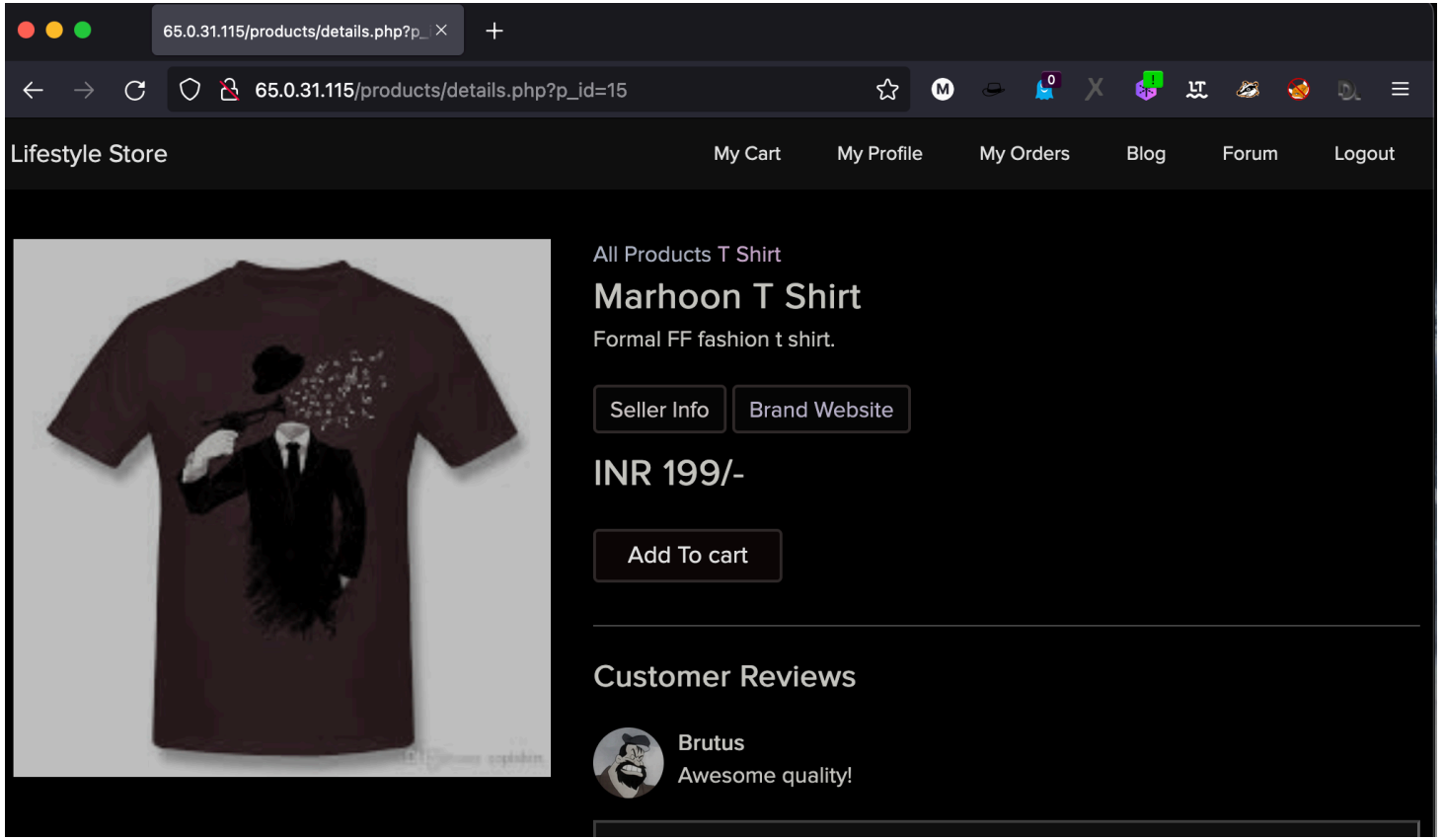


2. Cross Site Scripting (XSS)

Cross Site Scripting - XSS (Sever)	Below mentioned parameters are vulnerable to Cross Site Scripting. **Affected URL :** • http://65.0.31.115/products/details.php?p_id=("all id's") **Affected Parameters :** • Custom review text field **Payload:** • '><script>alert(1)</script>

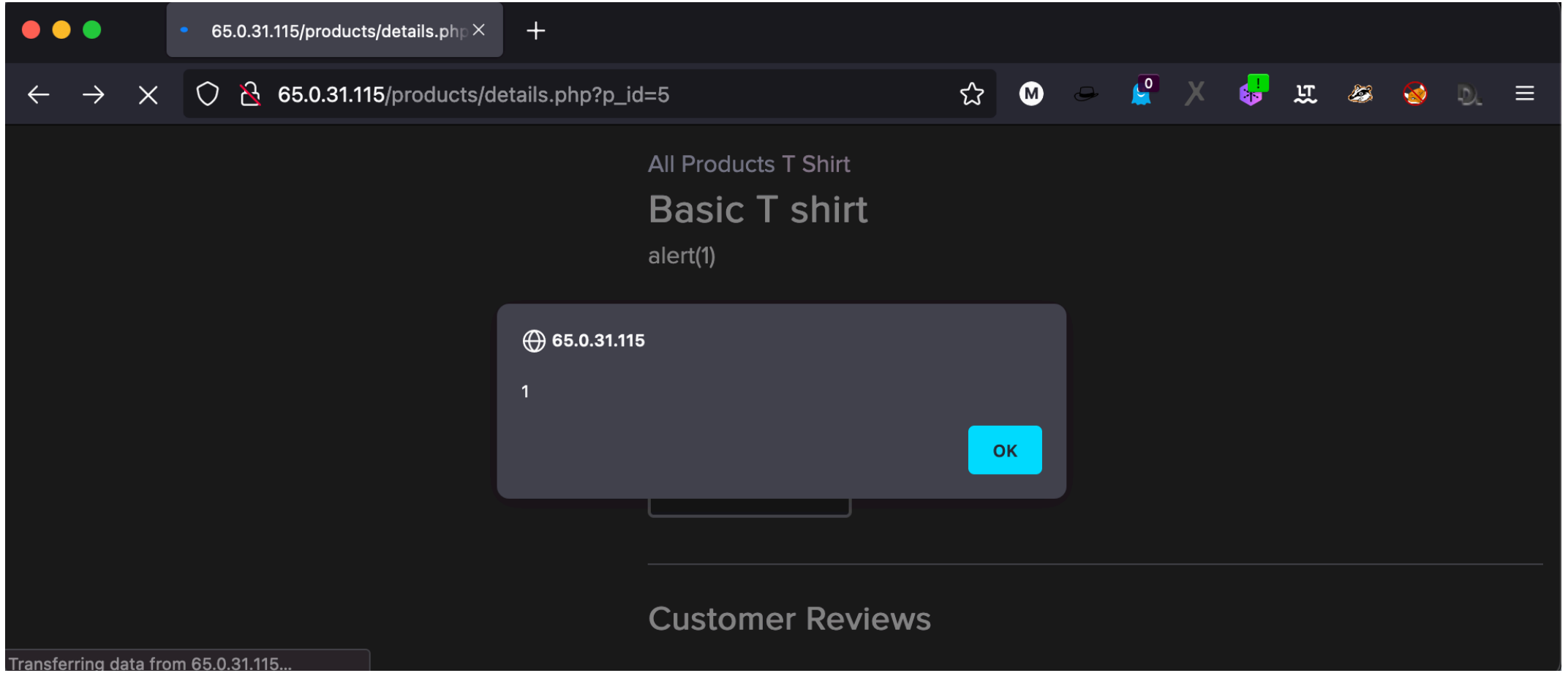
Observation

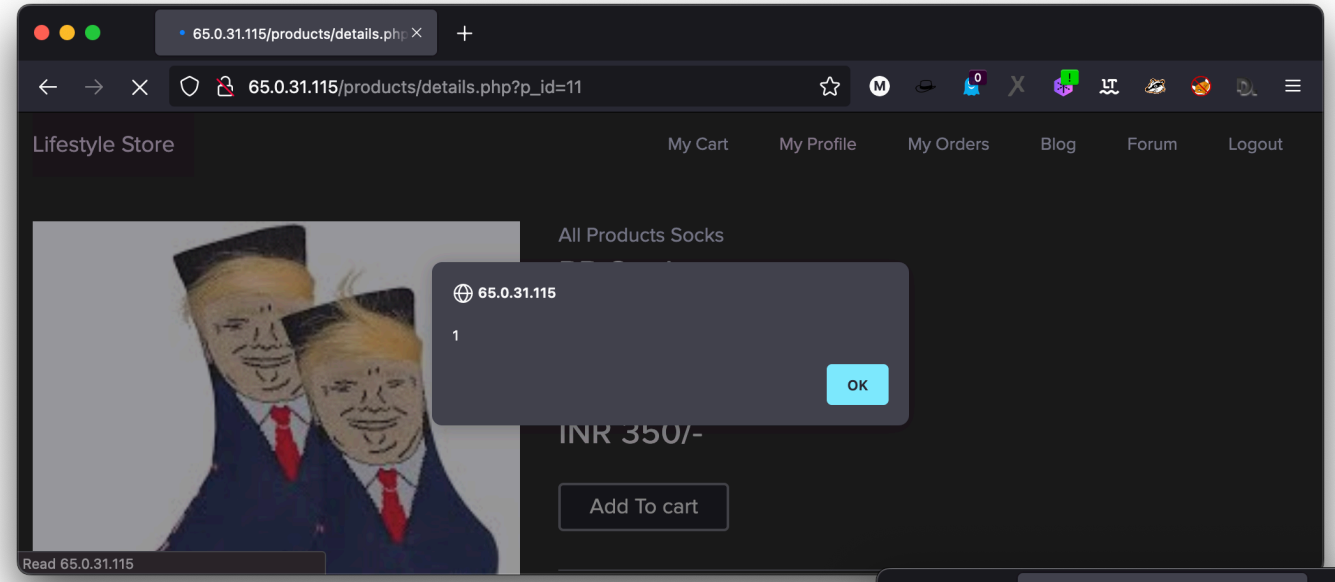
- Log in to your account. Then go to My Cart and then click on the SHOP NOW button and select any product,
- Or Navigate to **http://65.0.31.115/products/details.php?p_id=15** (I selected product number 15).

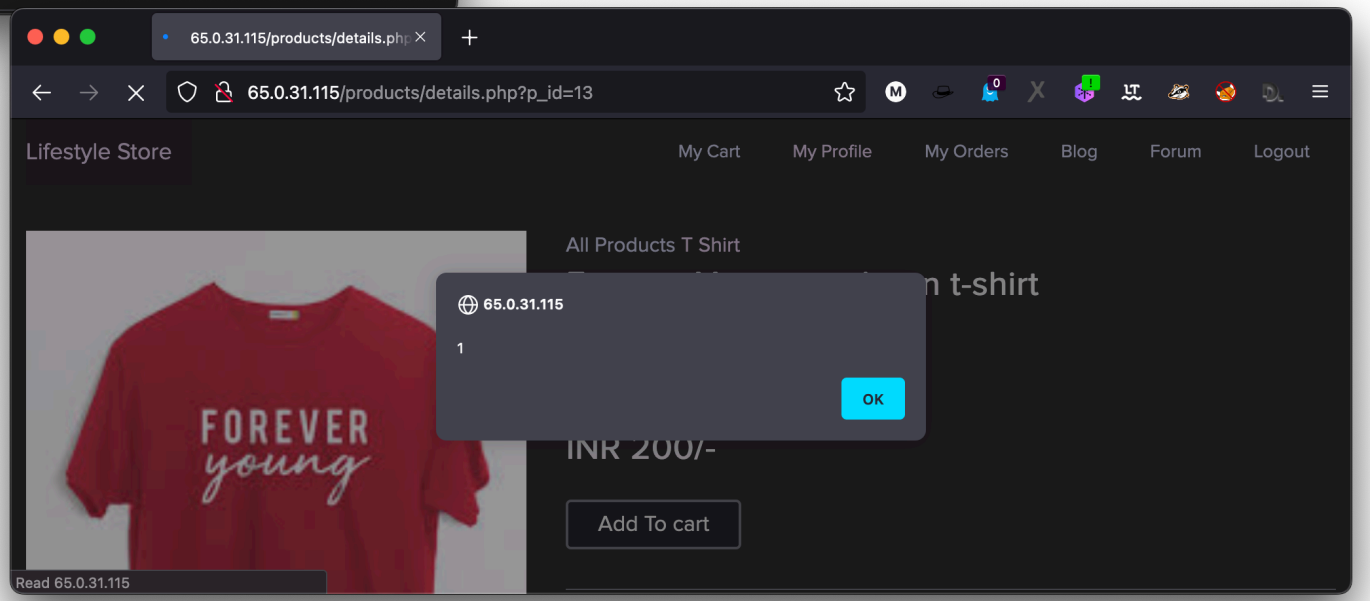


PoC - Custom Script executions

- Now, 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.




65.0.31.115/products/details.php
65.0.31.115/products/details.php?p_id=11
Lifestyle Store
My Cart
My Profile
My Orders
Blog
Forum
Logout
All Products Socks
65.0.31.115
1
OK
INR 350/-
Add To cart
Read 65.0.31.115


65.0.31.115/products/details.php
65.0.31.115/products/details.php?p_id=13
Lifestyle Store
My Cart
My Profile
My Orders
Blog
Forum
Logout
All Products T Shirt
t-shirt
65.0.31.115
1
OK
INR 200/-
Add To cart
FOREVER
young
Read 65.0.31.115

Business Impact – High

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

The attacker only needs to type the malicious script into the review field, and anyone who clicks on the link will be attacked by the hacker, and the victim will see hacker-controlled content on the website.As long as the user trusts the website, he/she will trust the content too.

As for PoC, a short screen recording has been attached along with in-screen rec/stored cross-site scripting.

```
                                    <div class="review_item">
    <div class="profile_picture">
        <img src="/static/images/uploads/customers/default.png">
    </div>
    <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>
</div>
```

Recommendation

Take the following precautions:

- Do sanitise all user input and block characters you do not want.
- Before printing special HTML characters like " > into HTML entities" (%22), convert them.

References:

https://owasp.org/www-community/attacks/xss/
https://en.wikipedia.org/wiki/Cross-site_scripting
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 suffers from an Insecure Direct Object Reference (IDOR) that allows attackers to get access to other customers' order details along with shipping details and payment modes. **Affected URL :** • http://65.0.31.115/orders/orders.php?customer=("all custoomer id") **Affected Parameters :** • Customer (GET parameters)

3. Insecure Direct Object Reference

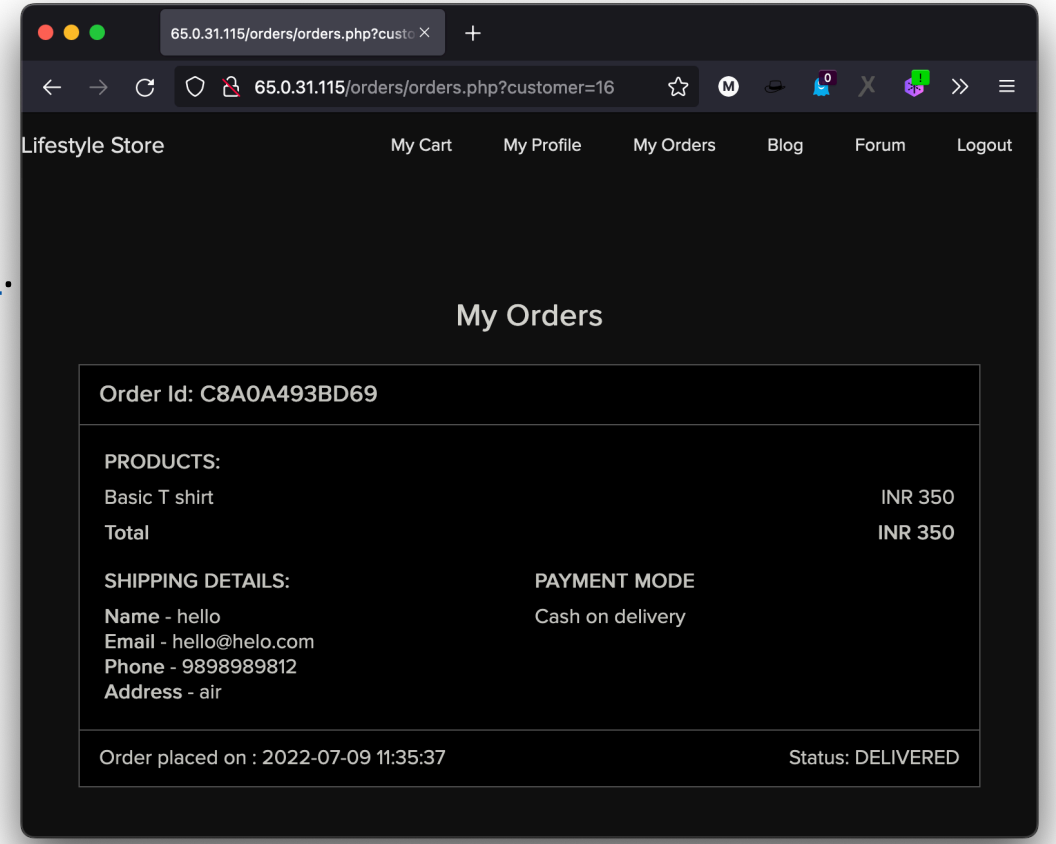
Insecure Direct Object Reference (Critical)	Similar issue is observed on the below mentioned login pages too **Affected URL :** • http://65.0.31.115/products/details.php?p_id=5 • http://65.0.31.115/forum/index.php?u=/user/profile/1 **Affected Parameters :** • p_id(GET Parameter) • u=/user/profile/(any id)

Observation

- Login to your account and go to the My orders section.

- Your My Orders section will be displayed. Notice the URL: http://65.0.31.115/orders/orders.php?customer=16.

- It contains the user's customer id, and we get the order details, shipping details, and payment mode for our user.



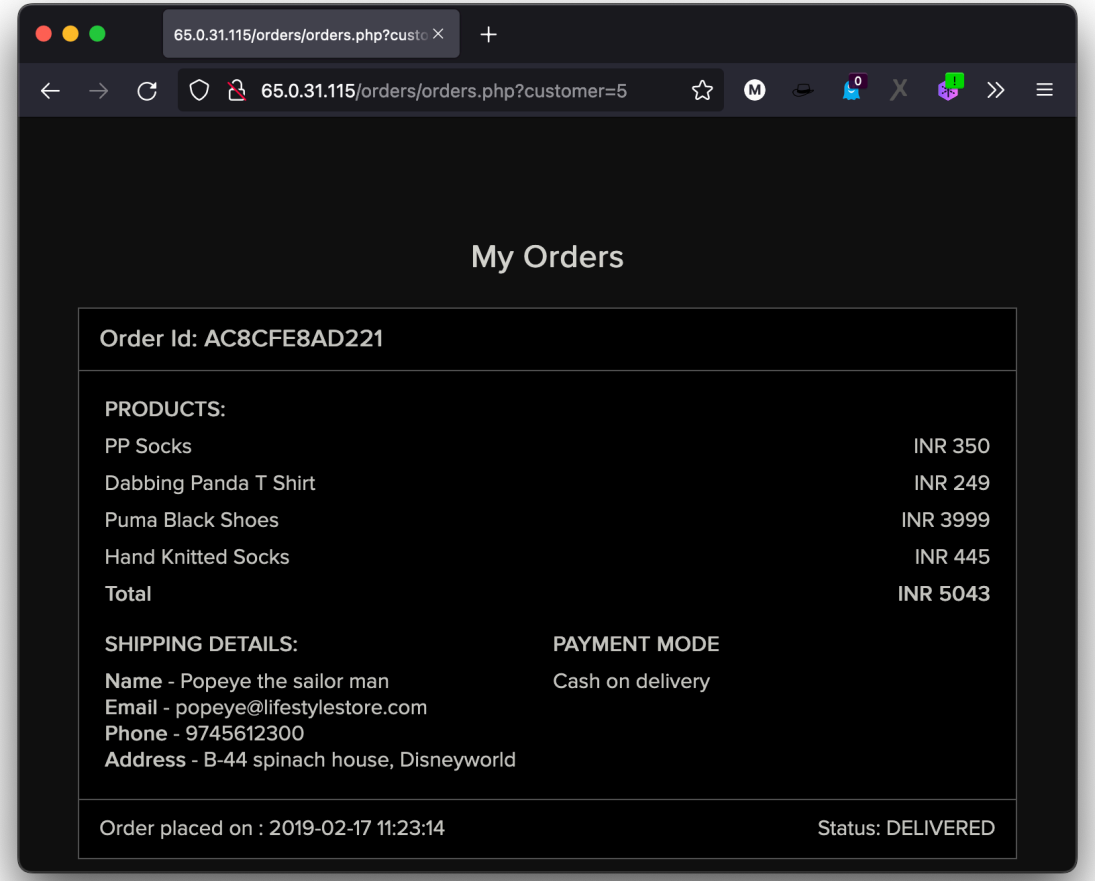
Observation

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

Request ▲	Payload	Status	Error	Timeout	Length	Comment
1	1	302	☐	☐	505	
2	2	200	☐	☐	6419	
3	3	200	☐	☐	6430	
4	4	302	☐	☐	505	
5	5	200	☐	☐	7080	
6	6	302	☐	☐	505	
7	7	302	☐	☐	505	
8	8	200	☐	☐	9718	
9	9	200	☐	☐	3019	
10	10	200	☐	☐	3019	
11	11	200	☐	☐	3019	
12	12	200	☐	☐	3019	
13	13	200	☐	☐	15383	
14	14	200	☐	☐	6056	
15	15	200	☐	☐	3019	
16	16	200	☐	☐	6072	
17	17	302	☐	☐	505	
18	18	302	☐	☐	505	
19	19	302	☐	☐	505	
20	20	302	☐	☐	505	

Observation

- 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).



Business Impact – Extremely High

A malicious hacker can read the order information of any user just by knowing the customer ID. This discloses critical order information to users, including:

- Name
- Order Id
- The Bill Mobile Number
- Email Address
- Physical Address
- 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 or the black market.

Furthermore, because there are no rate limiting checks, the attacker can brute force the customer id for all possible values and obtain bill information for every organisation user, resulting in massive information leakage.
As a PoC, order details of a few users are dumped in the folder named "customer order details".

Recommendation

Take the following precautions:

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

References:

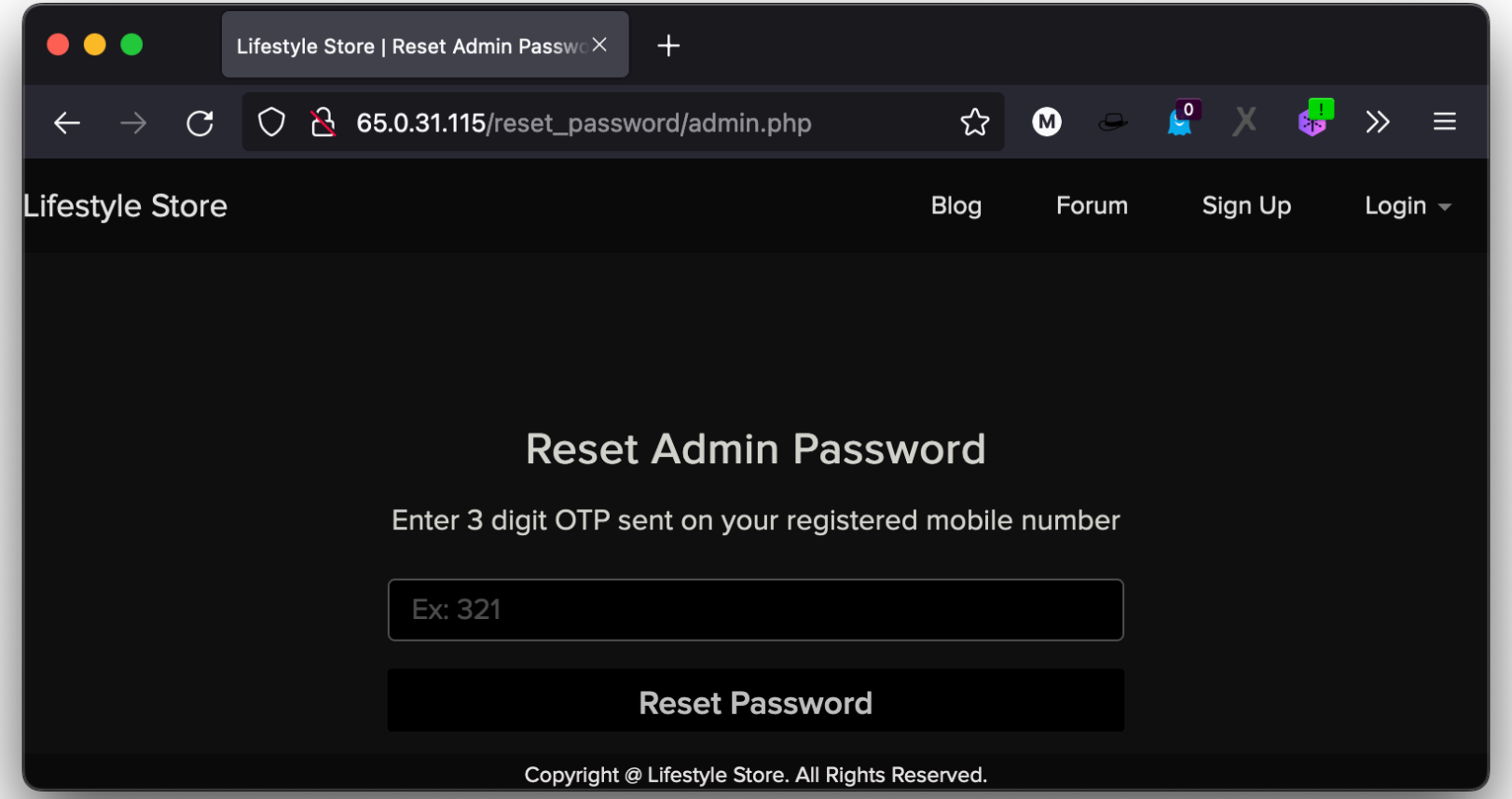
http://owasp.org/index.php/Insecure_configuration_Management
http://owasp.org/index.php/Top_10_2013-A4-Insecure_Direct_Object_Referrences

4. Rate Limit Issues

Account Takeover Using OPT Bypass (Critical)	The below mentioned login page allows login via OTP, which can be brute forced. **Affected URL :** • http://65.0.31.115/login/admin.php **Affected Parameters :** • OTP (POST parameters)

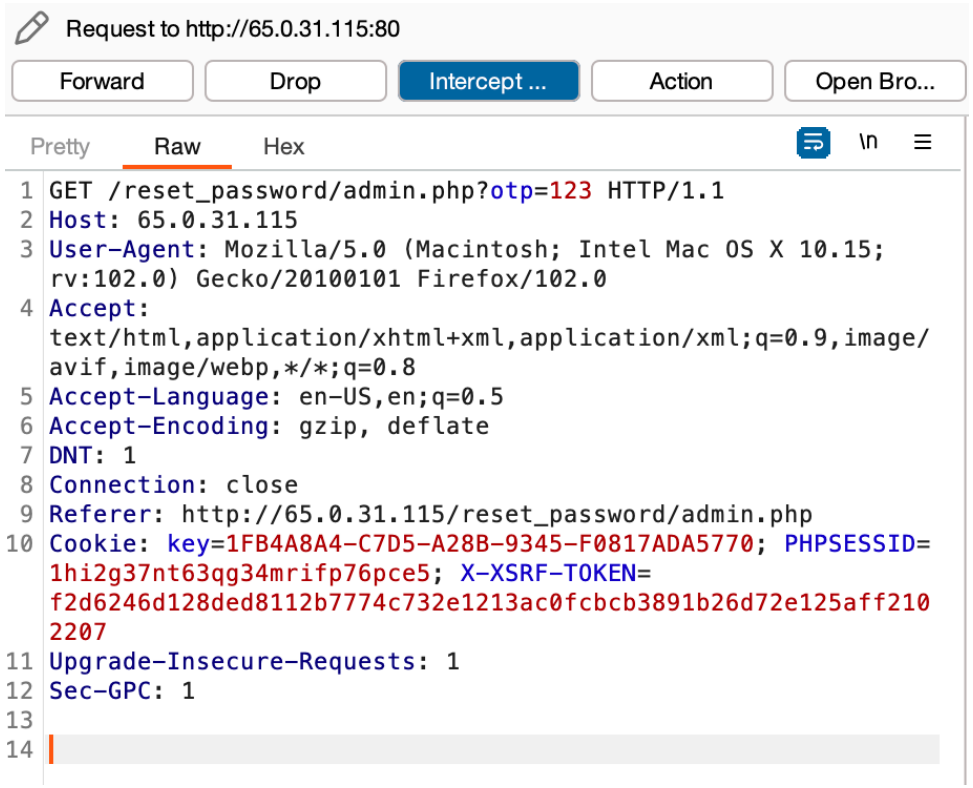
Observation

- Navigate to http://65.0.31.115/login/admin.php. You will see a "Forgot your password?"hyperlink which asks for OTP, which is sent to the admin’s phone number. Write any 3-digit number (i.e.,any number from 100-999) and intercept the request with Burp Suite.



Observation

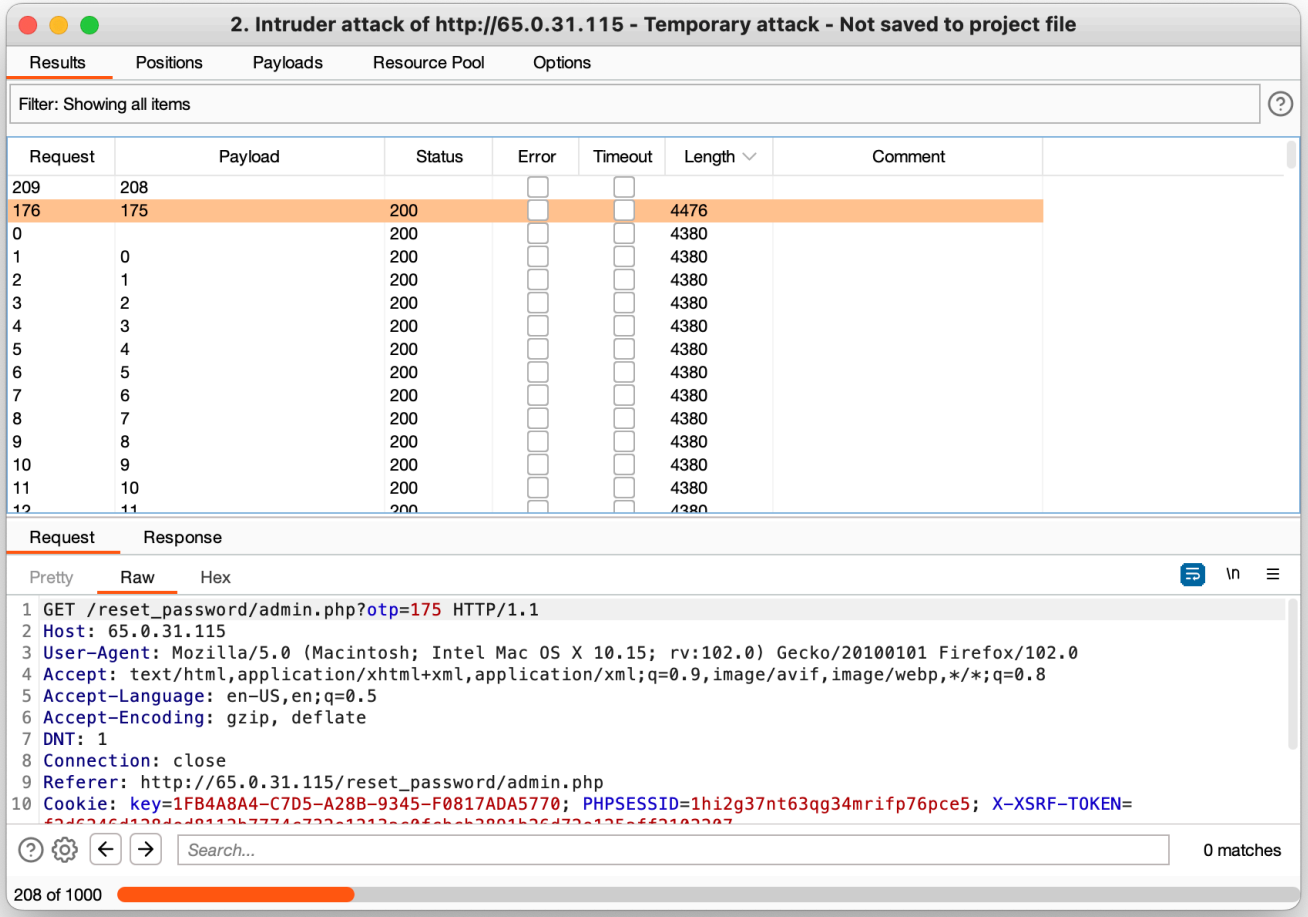
- The following request will be generated containing the OTP parameter (GET)



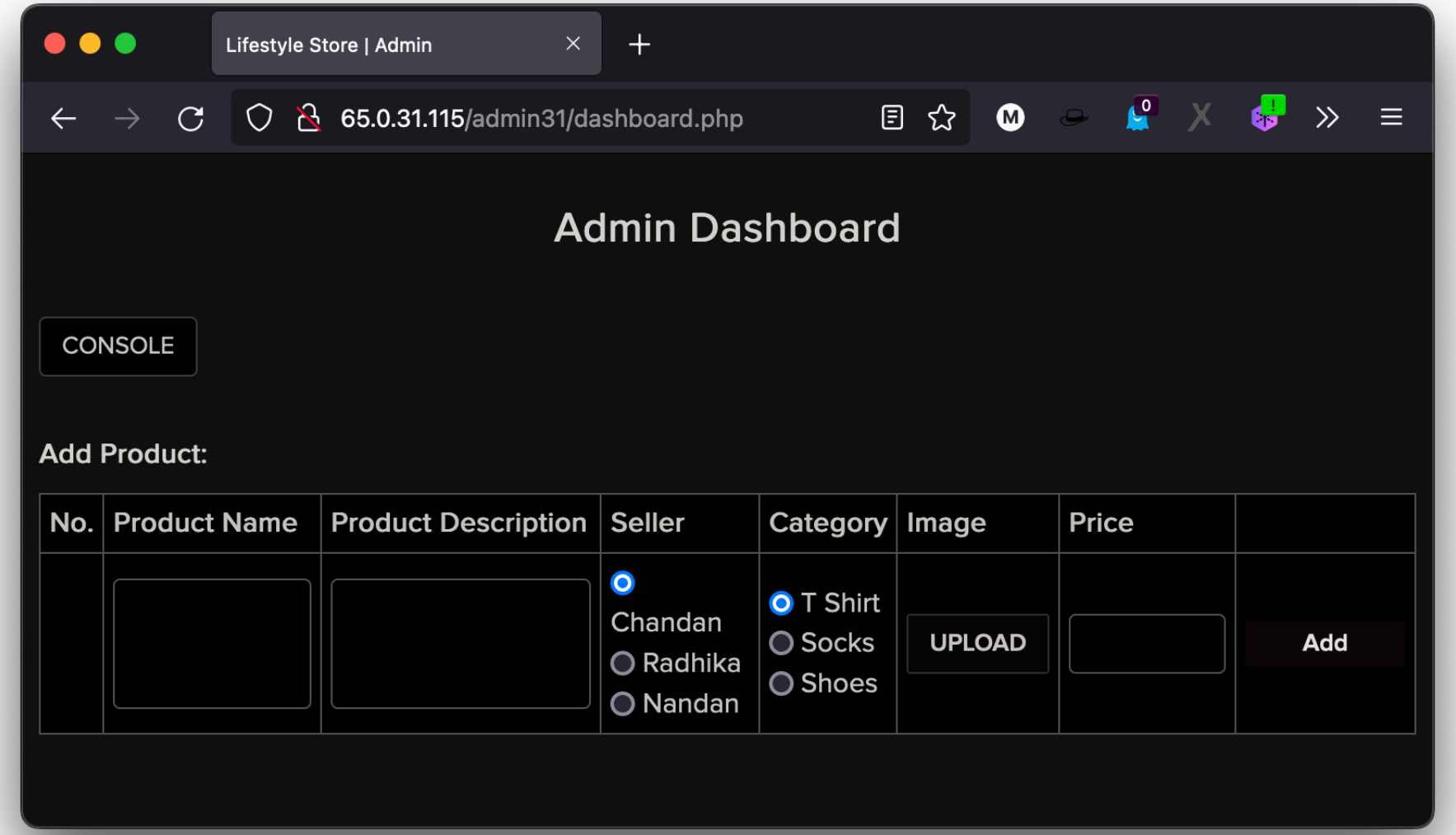
Observation

- 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 the admin password and then use the new admin password to login as administrator.

- The OTP for this session was “175”.



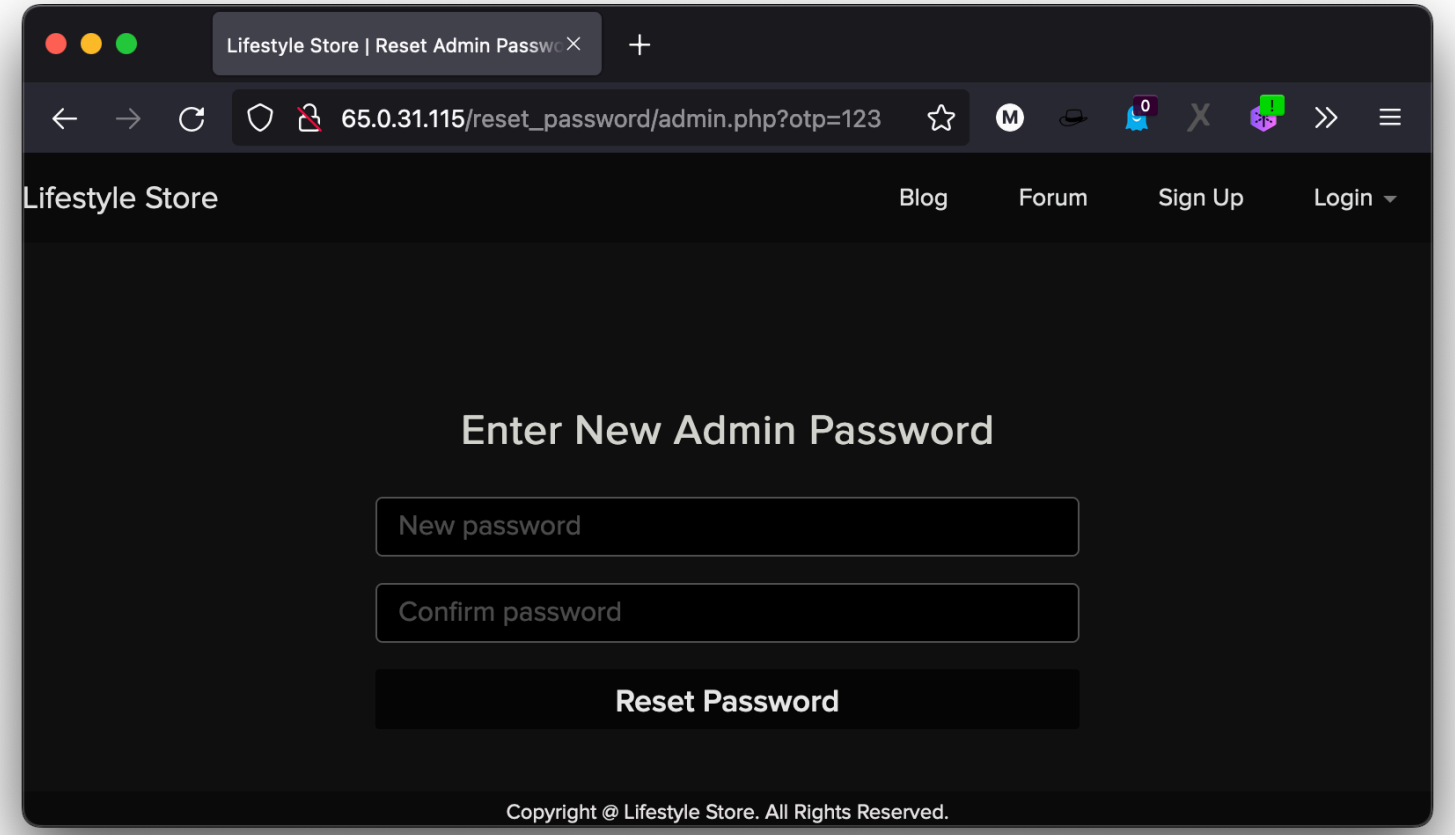
PoC - Access to Admin Dashboard



Business Impact – Extremely High

A malicious hacker can gain complete access to an admin account simply by brute-forcing, and because there is no limit to the number of attempts, a hacker can attempt as many times as he wants. This leads to the complete compromise of the personal user data of every customer

Once the attacker logs in as admin, he can perform actions on behalf of the victim (admin) that could result in significant financial loss for him/her, such as changing the name, image, and even the price of the products.



Recommendation

Take the following precautions:

- Use proper rate-limiting checks on the number of OTP checks and generation requests.
- After multiple failed attempts, implement anti-bot measures such as ReCAPTCHA.
- The OTP should expire after a certain amount of time, such as 2-5 minutes.
- OTP should be at least 6 digits and alphanumeric for more.

References:

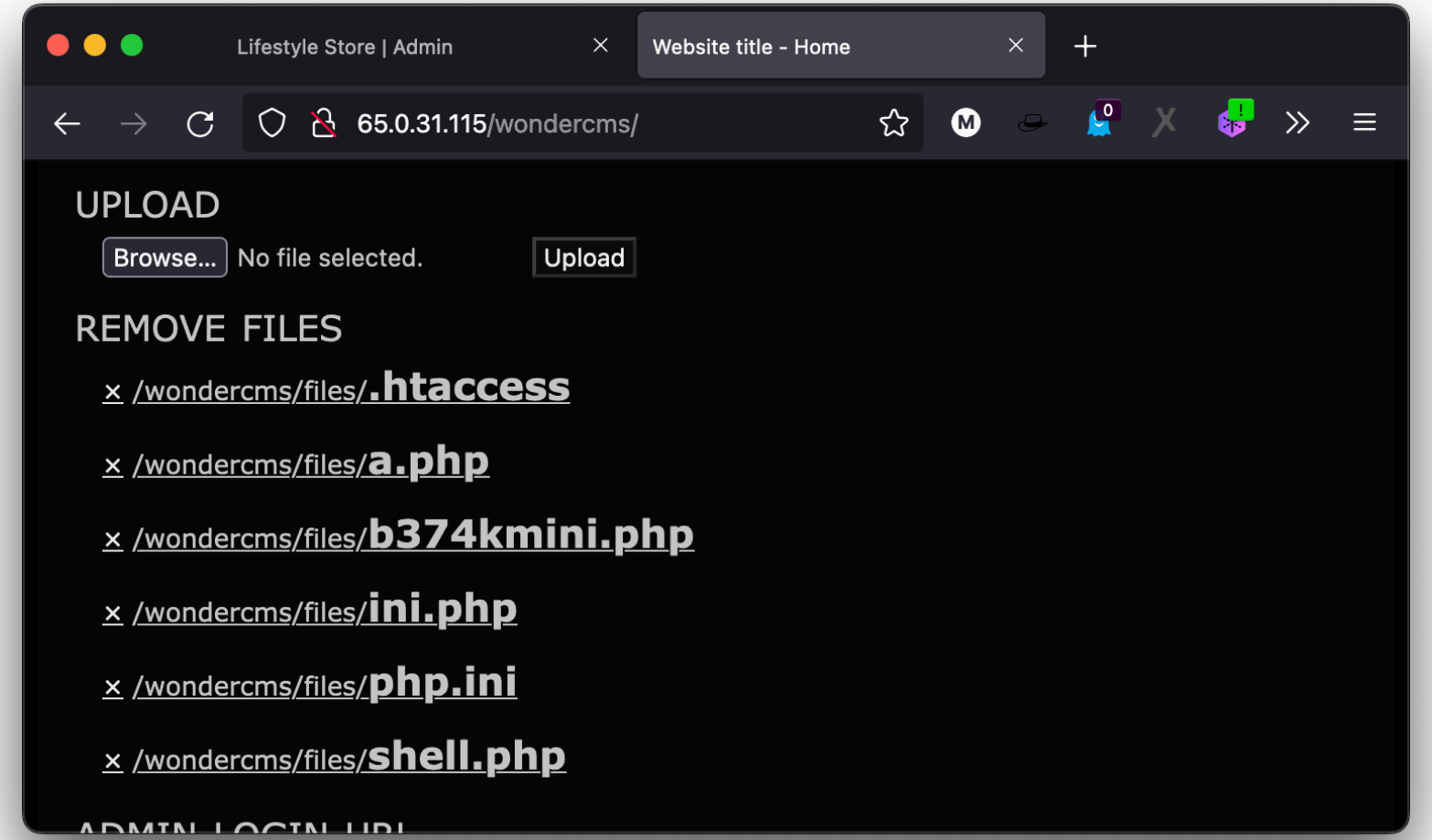
https://www.owasp.org/index.php/Testing_Multiple_Factors_Authentication_(OWASP-AT-009)
https://www.owasp.org/index.php/Blocking_Brute_Force_Attacks

5. Insecure File Uploads

Insecure File Uploads (Critical)	The below mentioned URL is vulnerable to Insecure File Uploads. **Affected URL :** • http://65.0.31.115/wondercms/ **Affected Parameters :** • Backdoor shell (anonymous.php)

Observation

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

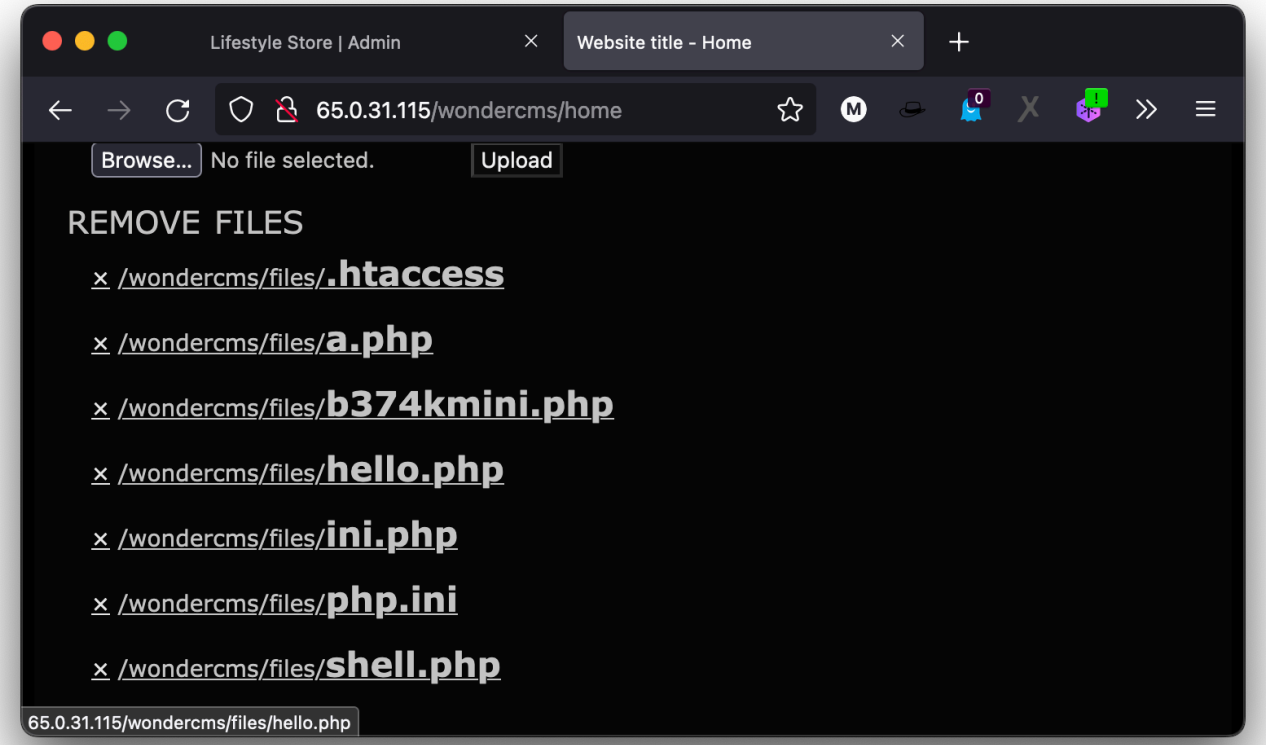


Observation

It looks like we can upload files here. Let's try uploading a file named anonymous.php.

File uploaded.

And it's successfully uploaded.



PoC

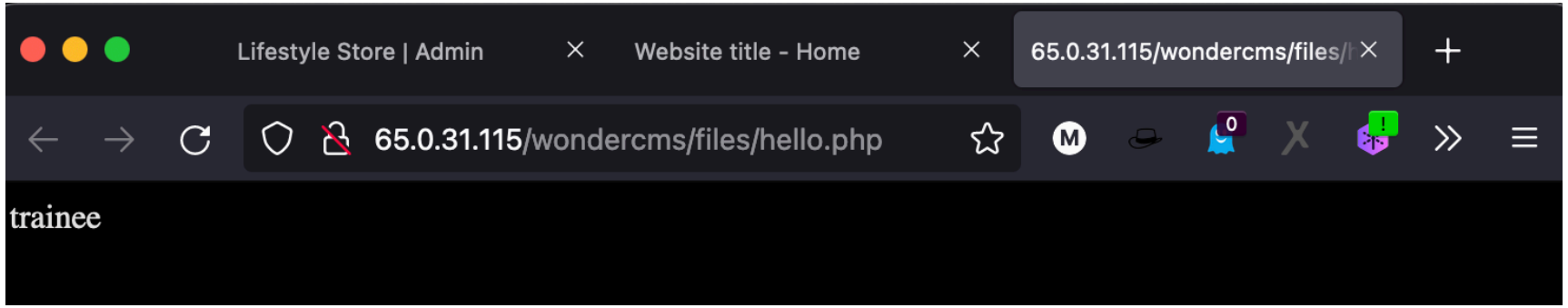
- Shell - hello.php

hello.php

Sanket Nighot Internshala Ethical Hack

```
1  <?php
2  echo whoami;
3
4  >;
```

- The uploaded shell was executed successfully.



Business Impact – 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 without first running a whitelist filter.
- All the control characters, Unicode, and special characters should be discarded.

References:

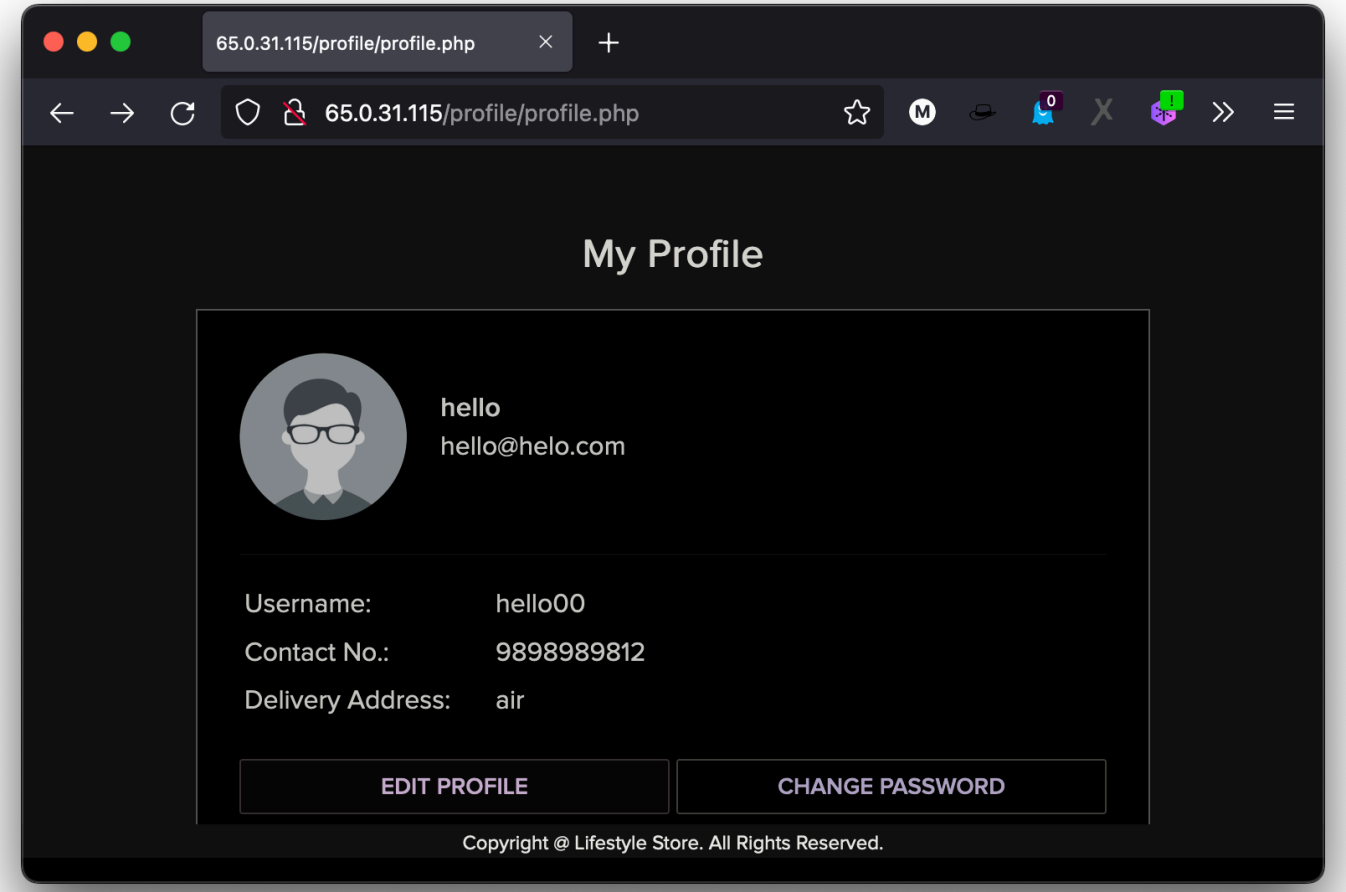
https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload
https://www.hackingarticles.in/comprehensive-guide-on-unrestricted-file-upload/

6. Client Side Filter Bypass

Client Side Filter Bypass (Moderate)	Below mentioned parameters are vulnerable to Client Side Filter Bypass **Affected URL :** • http://65.0.31.115/profile/16/edit/

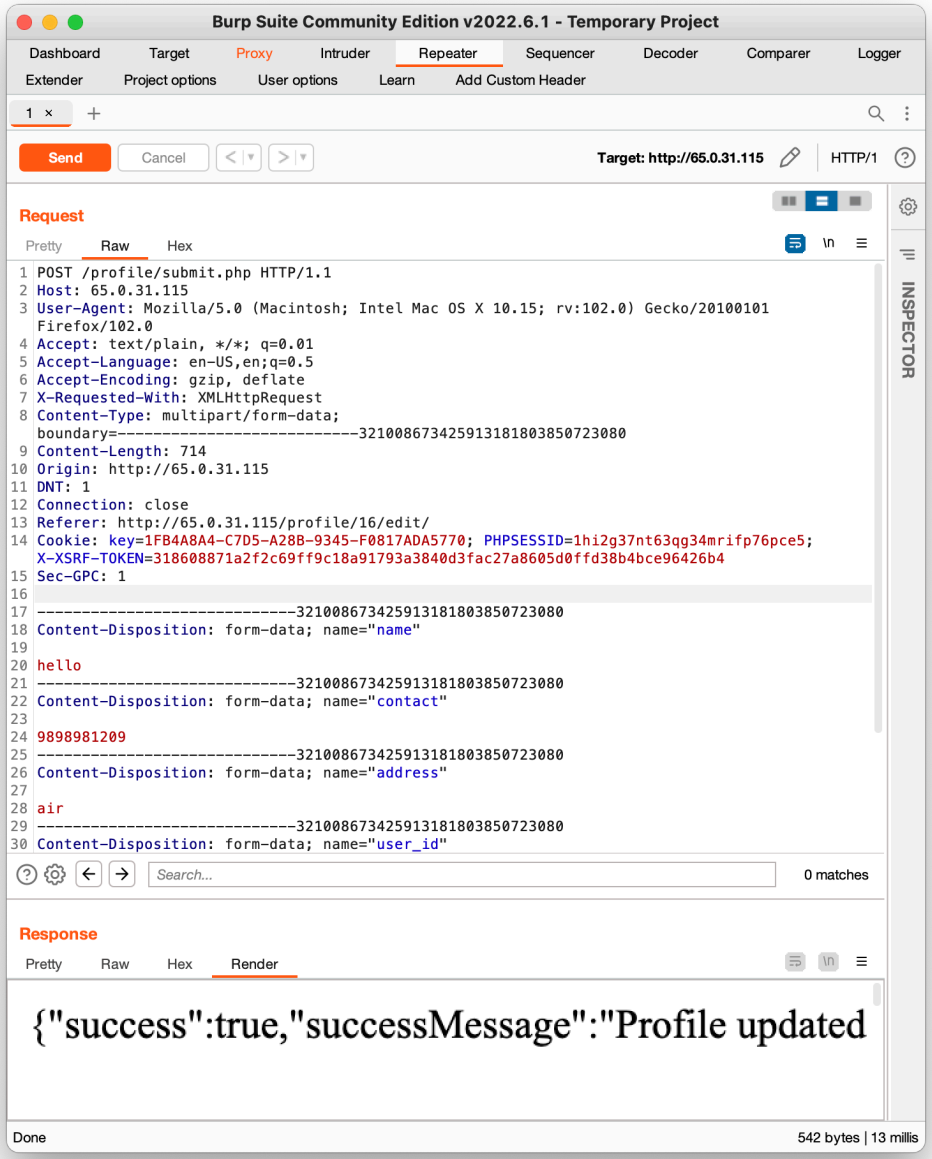
Observation

- Login to your account and go to the My Profile section.
- Now, click the edit profile button and update any of your details, including your phone number.
- I updated my phone number from 9898989898 to 9898989812 .
- Click the UPDATE button once more to intercept the request with Burp Suite.



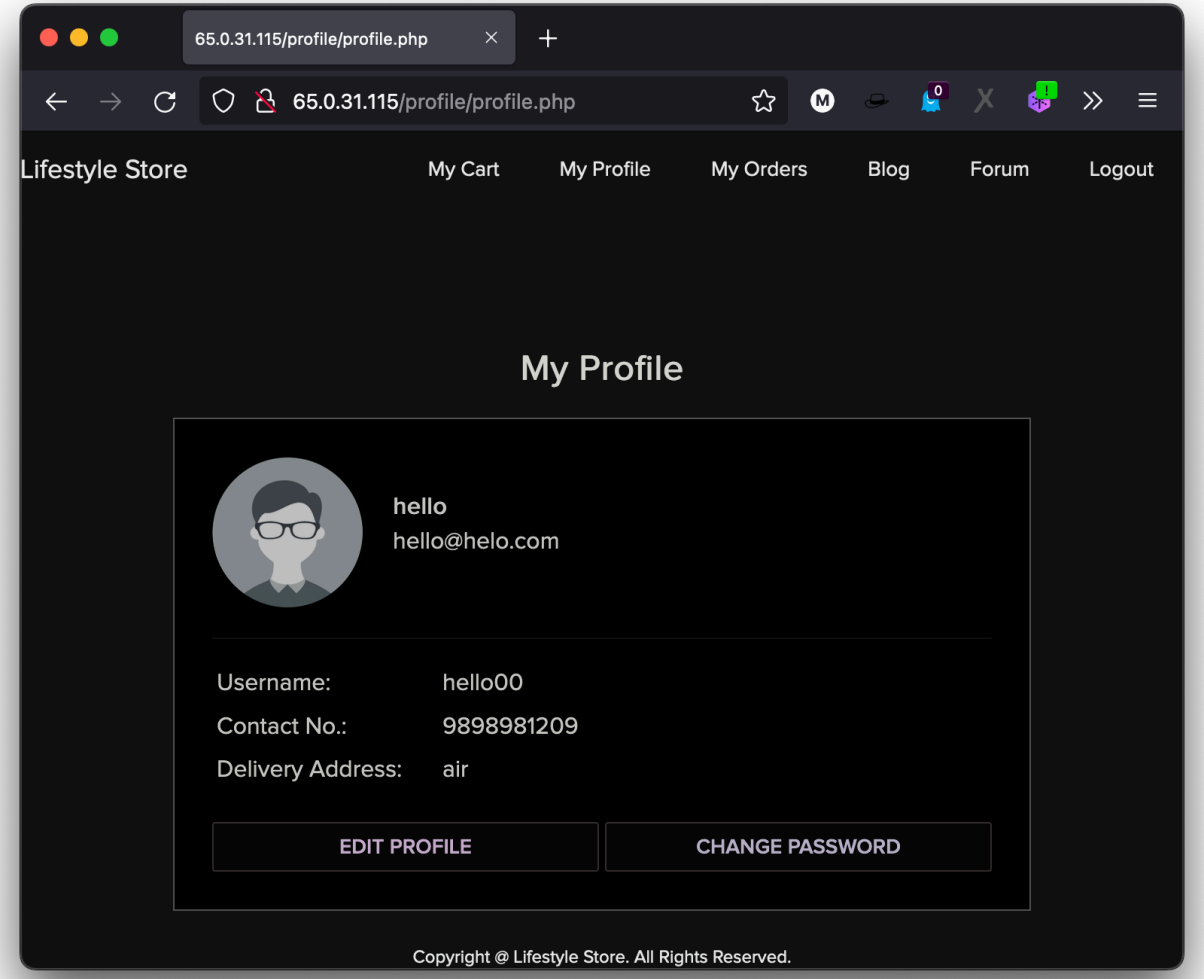
Observation

- Now, send the request to the repeater and edit the phone number.
- I changed it from 9898989812 to 9898981209 and hit Send.



PoC - Profile update Successfully

As a PoC, a short screen recording has been attached along with in-screen rec/client side filter bypass.



Business Impact – Moderate

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

Recommendation

Take the following precautions:

- Only use server-side code for all critical checks.
- 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:

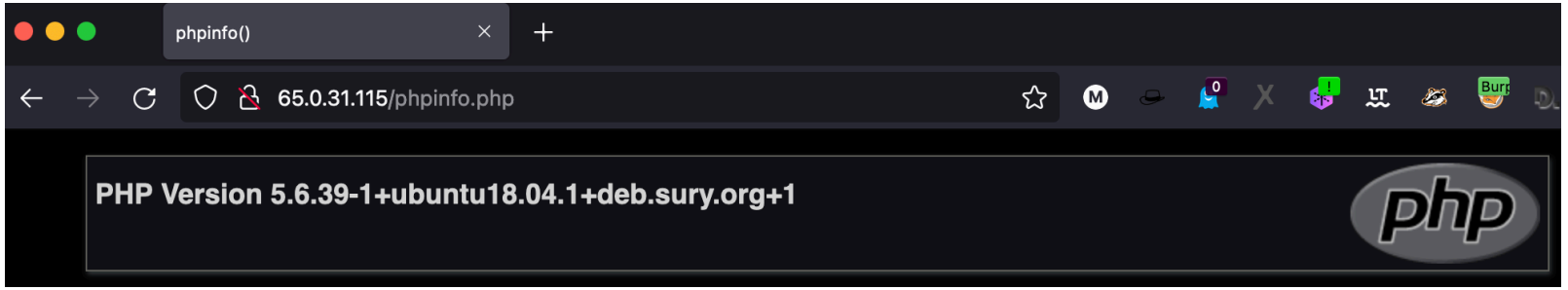
https://portswigger.net/support/using-burp-to-bypass-client-side-javascript-validation
https://www.slideshare.net/SamBowne/cnit-129s-ch-5-bypassing-clientside-controls

7. Components with known Vulnerabilities

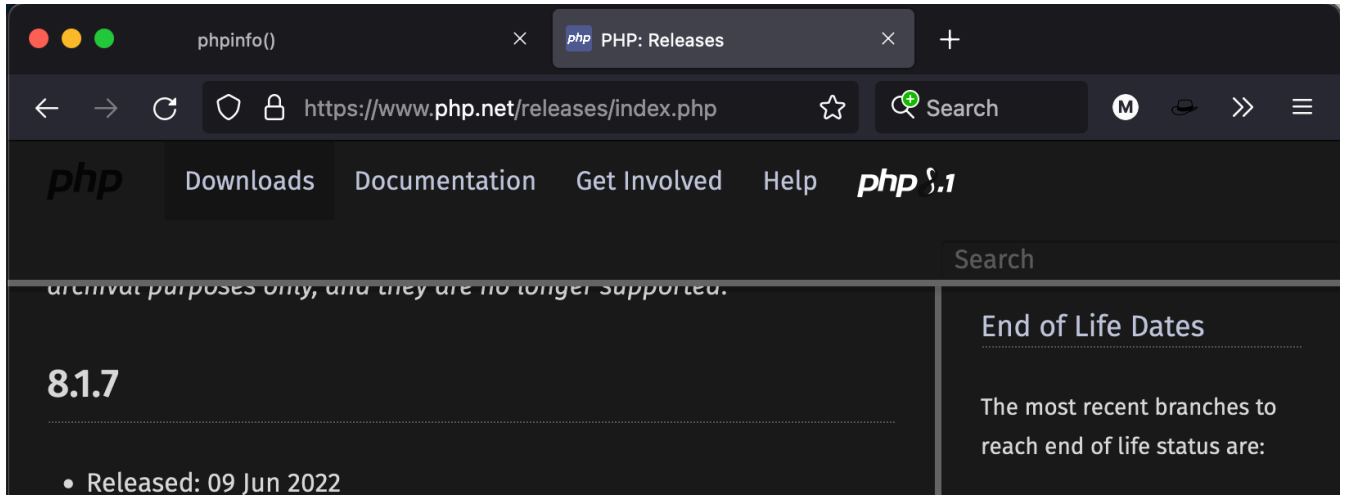
Components with known Vulnerabilities (Critical)	The below mentioned URL is vulnerable to Components with known Vulnerabilities. **Affected URL :** • http://65.0.31.115/wondercms/ • http://65.0.31.115/forum/ And the PHP Version.

Observation

- The PHP version of this website is 5.6.39-1, which is out of date.



- Latest PHP version is 8.1.7 .



Observation

- Upon checking the versions of these components, they turned out to be outdated.
- Versions that are in :

PoC -

- Codoforum has exploit

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	CVE-2014-9261	22	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.

PoC -

- Wondercms 2.3.1 has public exploit

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	CVE-2017-14523	74			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	CVE-2017-14522	79		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	CVE-2017-14521	434			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.														

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

Recommendation

Take the following precautions:

- Update all the components and the PHP version which is running on them.
- Remove the information about the current version from their pages.

References:

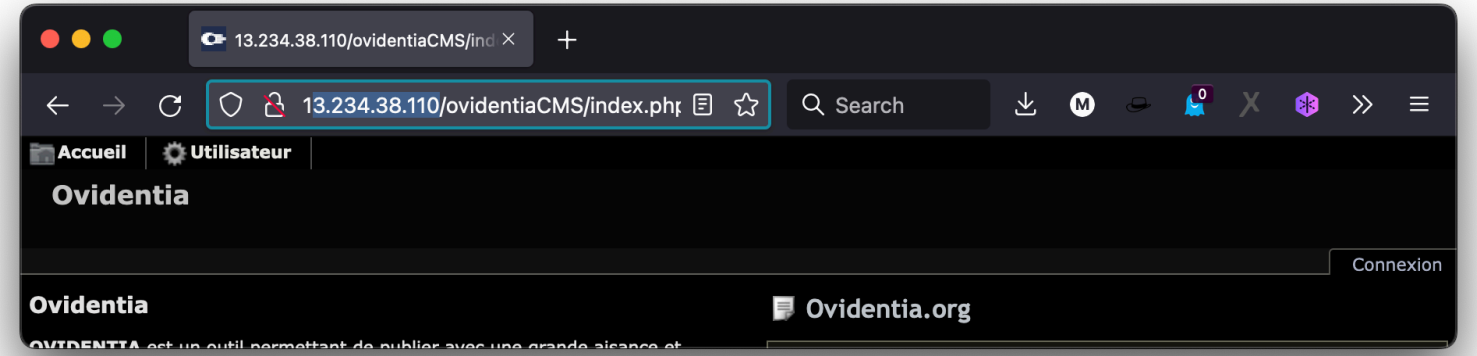
https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A9-Using_Components_with_Known_Vulnerabilities
https://www.cvedetails.com/vulnerability-list/vendor_id-15088/product_id-30715/version_id-235577/Wondercms-Wondercms-2.3.1.html
https://www.cvedetails.com/vulnerability-list/vendor_id-15315/Codoforum.html

8. Default Admin Password

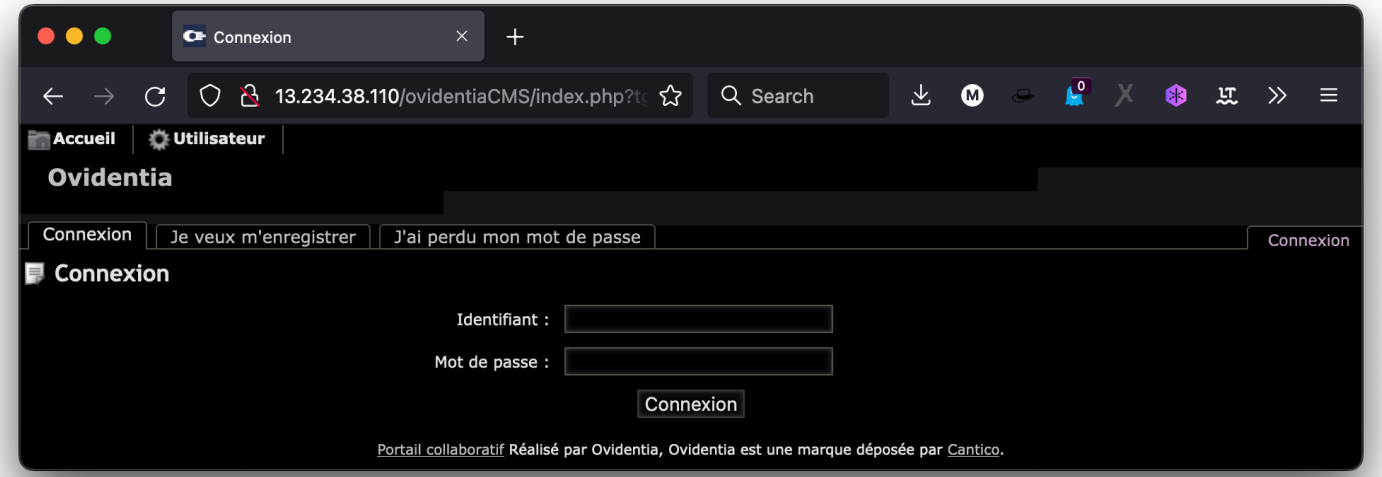
Default Admin Password (Critical)	The below mentioned URL is vulnerable to Default Admin Password. **Affected URL :** • http://13.234.38.110/ovidentiaCMS/index.php?tg=login&cmd=authform&msg=Connexion&err=&restricted=1 **Component Name:** • Ovidentia CMS(Content Management System

Observation

- Navigate to http://13.234.38.110/ovidentiaCMS/.
- In the evidential CMS page, there is an option called "Connection" to login as an admin.



- Upon clicking it, we can see this page.



PoC - OvidentiaCMS 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:

Business Impact - Extremely High

- The 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.
- Turn off the default debug pages.
- Hide the administrator login page.
- Remove all the default passwords and add your own, 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/

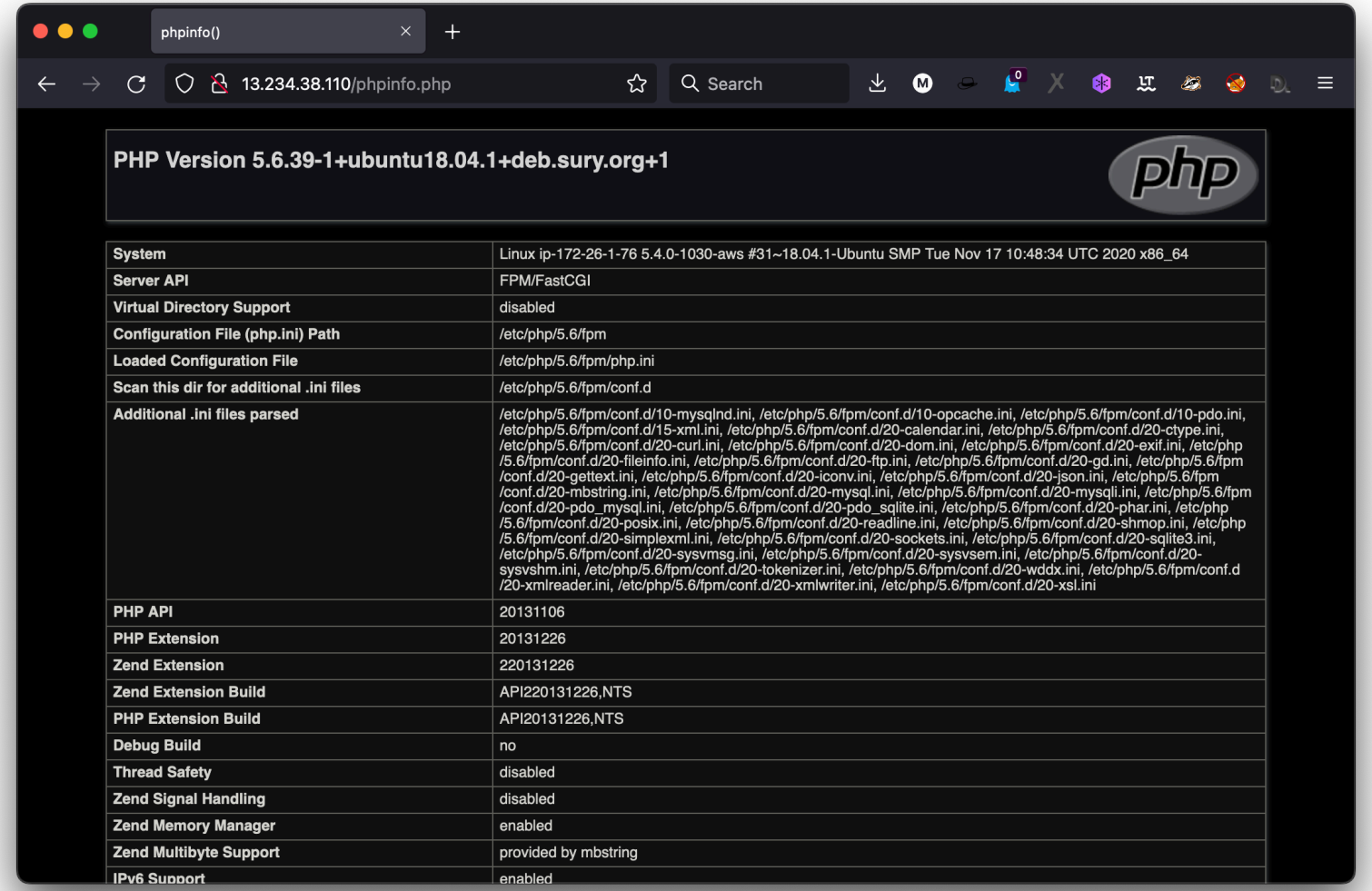
https://hdivsecurity.com/owasp-security-misconfiguration

https://www.tmdhosting.com/kb/question/ovidentia-hosting-requirements-ovidentia-manual installation/

10. Default File’s and Page’s

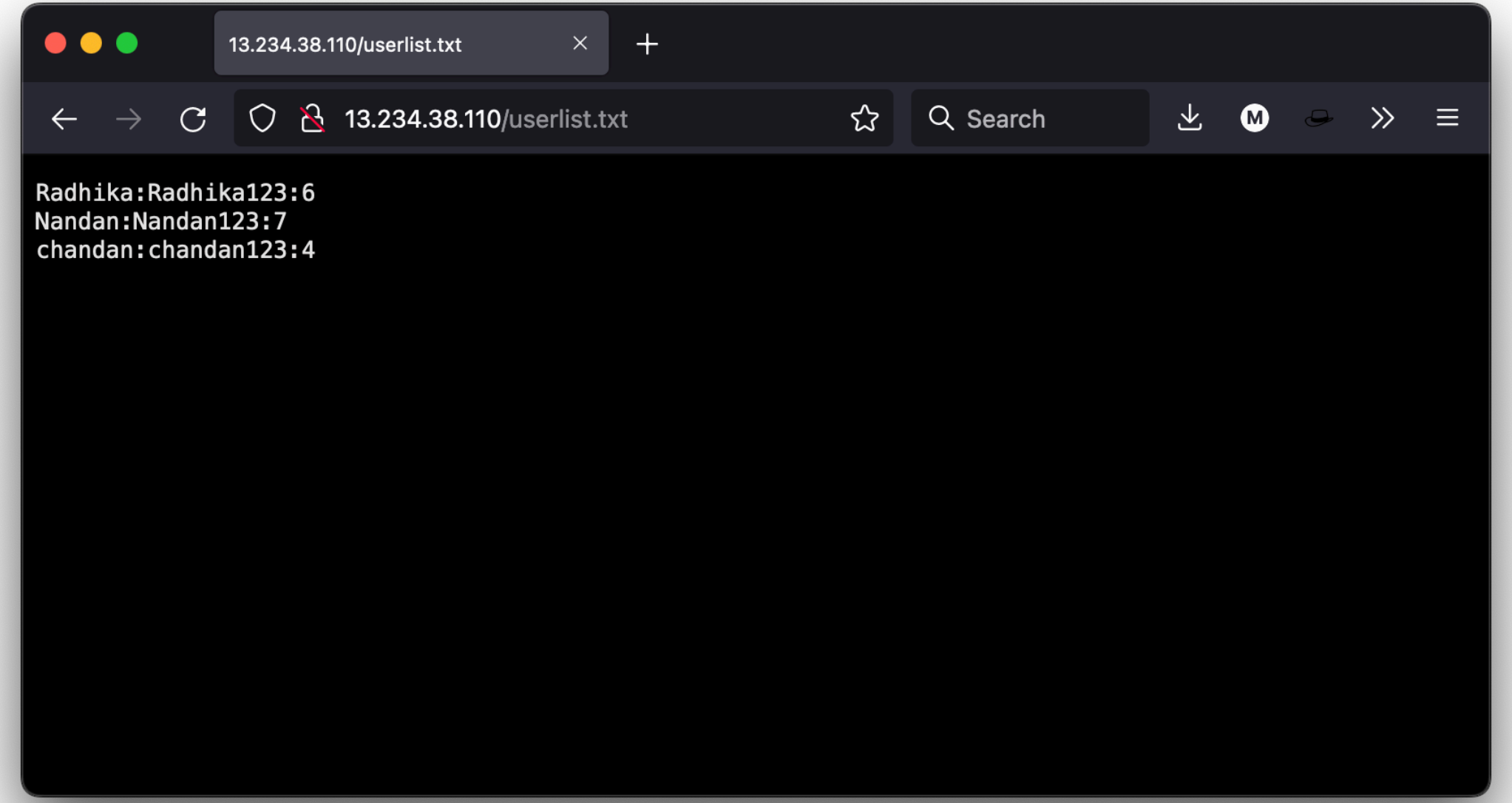
Default Files and Page’s (Low)	Below mentioned urls show the Default Files and Page’s **Affected URL :** • http://31.234.38.110/ **Default File and Pages present:** • Server-status • Robots.txt • Userlist.txt • Phpinfo.php • Composer.json

PoC - phpinfo.php

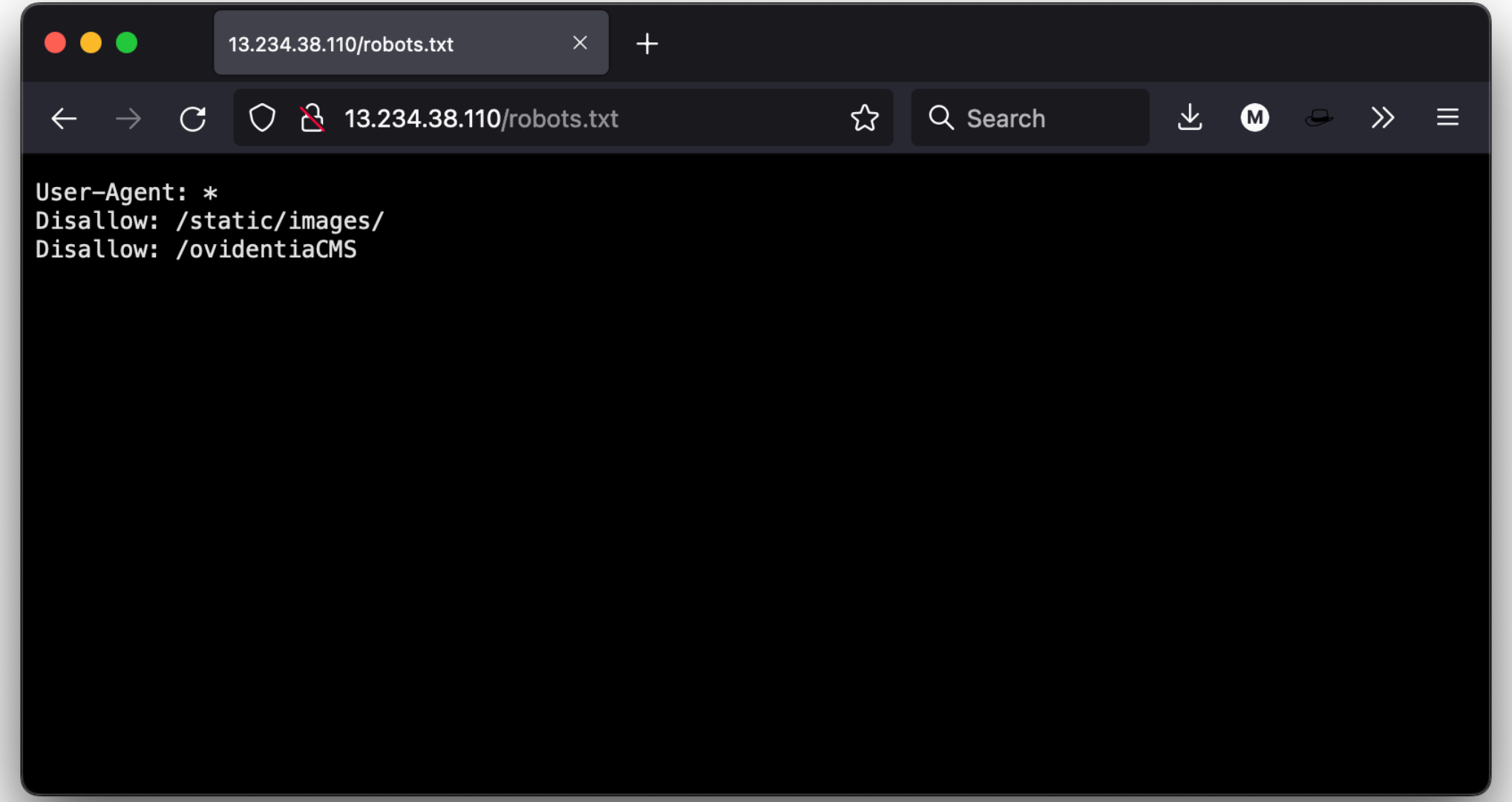


System	Linux ip-172-26-1-76 5.4.0-1030-aws #31~18.04.1-Ubuntu SMP Tue Nov 17 10:48:34 UTC 2020 x86_64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/5.6/fpm
Loaded Configuration File	/etc/php/5.6/fpm/php.ini
Scan this dir for additional .ini files	/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

PoC - userlist.txt



PoC - robots.txt



Business Impact – Moderate

It doesn’t harm the website directly, but it lets the hacker collect more internal information about the website, which the hacker might use against the organization.

Recommendation

Take the following precautions:

- Disable all default pages and folders by developer.

References:

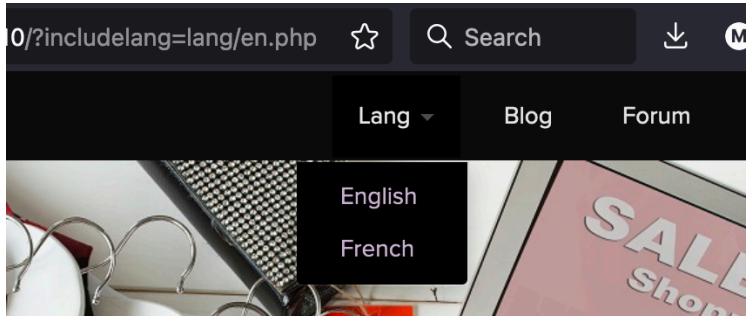
https://www.indusface.com/blog/owasp-security-misconfiguration/
https://hdivsecurity.com/owasp-security-misconfiguration

11. Remote File Inclusion

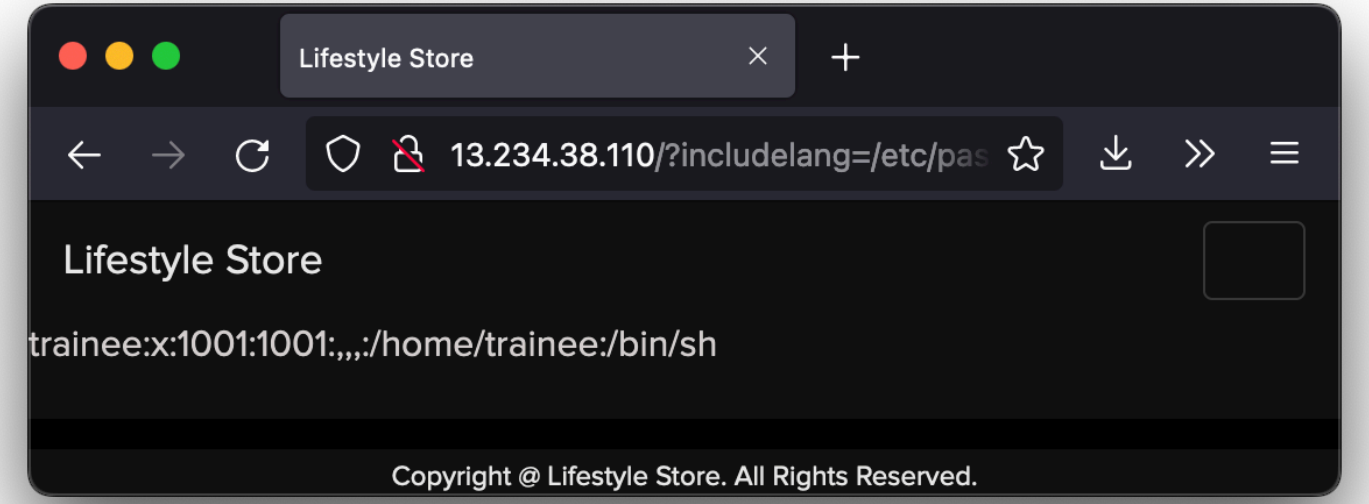
Remote File Inclusion (Critical)	The below mentioned URL is vulnerable to Remote File Inclusion. **Affected URL :** • http://13.234.38.110/?includelang=lang/fr.php **Affected parameter:** • /etc/passwd(/includelang=“here”) • google.com(/includelang="here")

Observation

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

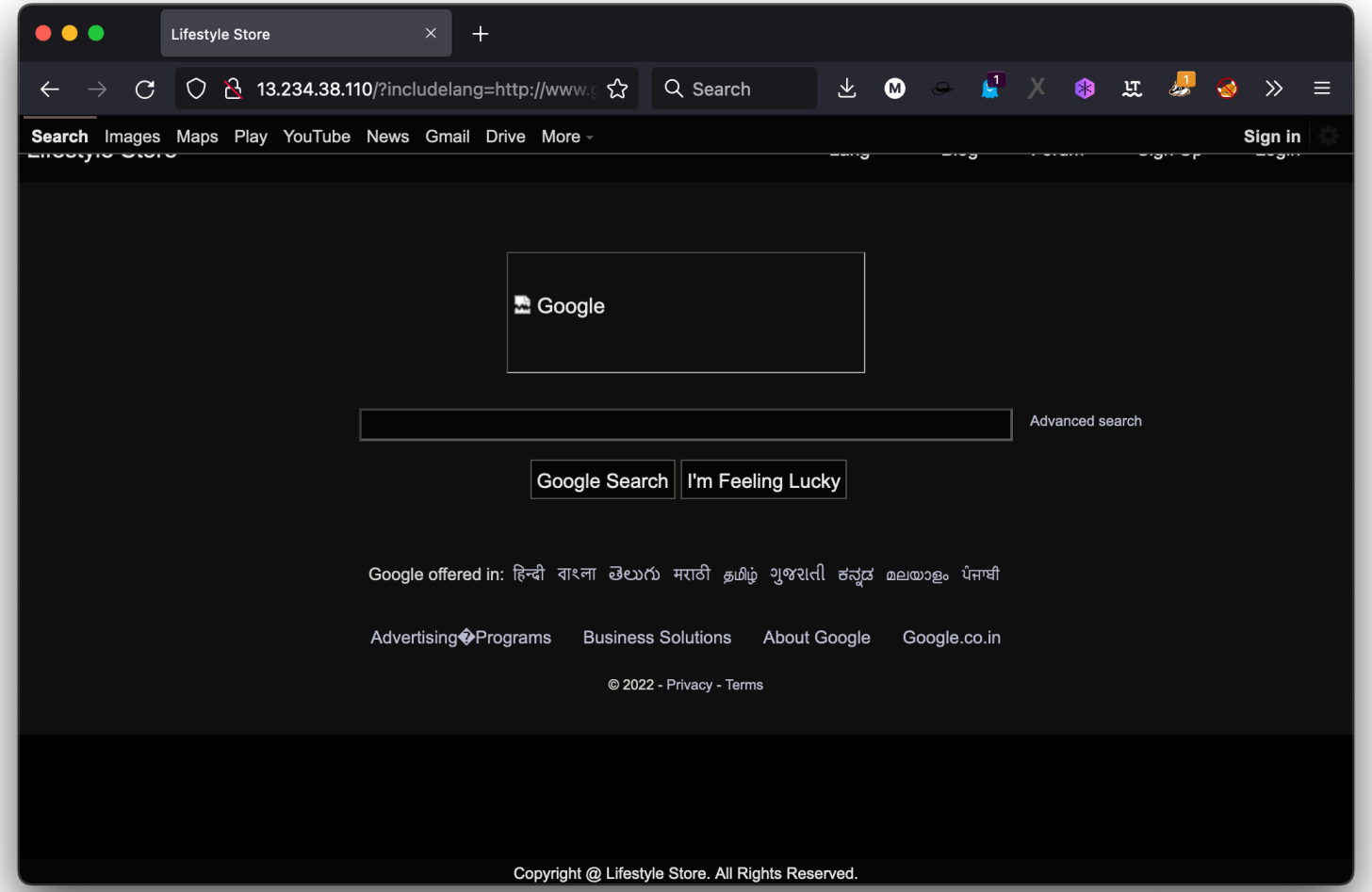


- Now, notice the URL, you get a ‘get’ parameter of includelang which is vulnerable to file inclusion.
- Here, we enter the payload: includelang=/etc/passwd and on executing this file gives us the username.



PoC - Attacker can Uploads Shell

- An 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 root access to your website.
- He can carry out commands.
- He can gain access to the server and infect other websites hosted on that server via the website.
- He can even deface your websites.

Recommendation

Take the following precautions:

- 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:

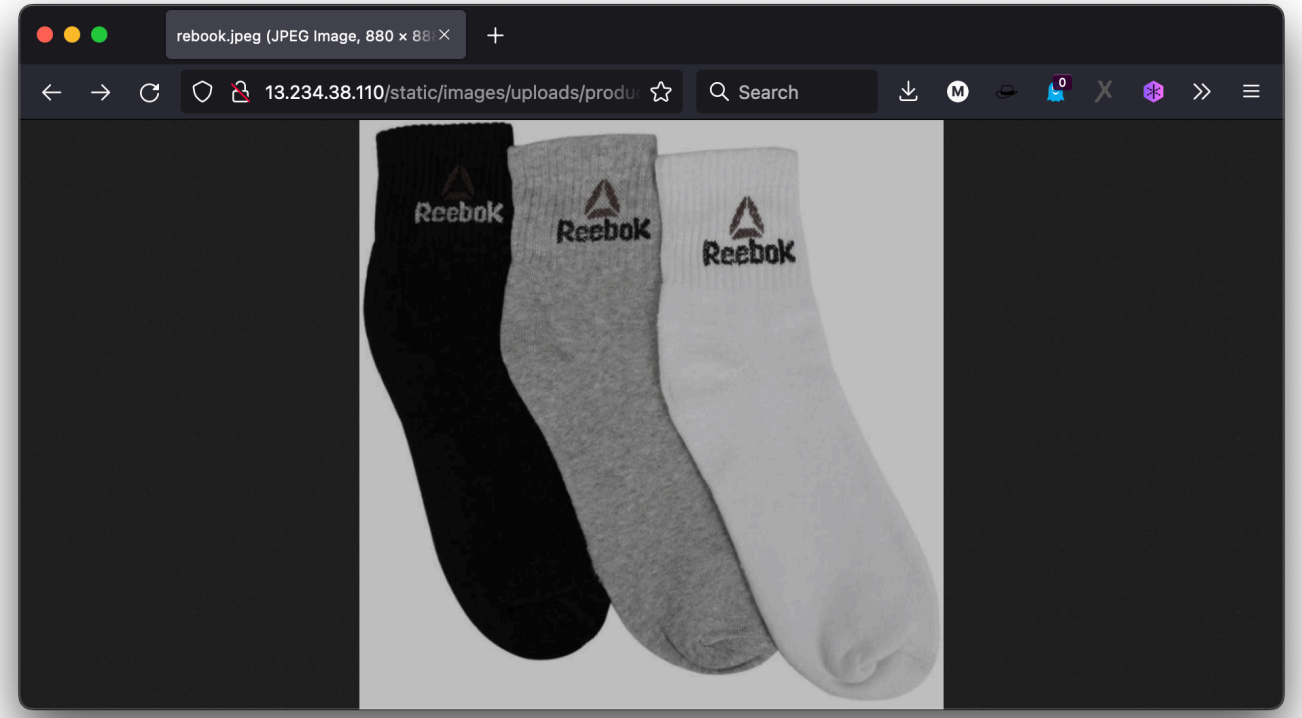
https://www.pivotpointsecurity.com/blog/file-inclusion-vulnerabilities/
https://www.netsparker.com/blog/web-security/local-file-inclusion-vulnerability/
https://en.wikipedia.org/wiki/File_inclusion_vulnerability

12. Directory Listing

Directory Listing (Moderate)	Here are some more of link that are vulnerable to directory listing. **Affected URL :** • http://13.234.38.110/static/images/uploads/products/rebook.jpeg Here are other similar URLs that leak critical information via directory listing vulnerability. • http://13.234.38.110/robots.txt

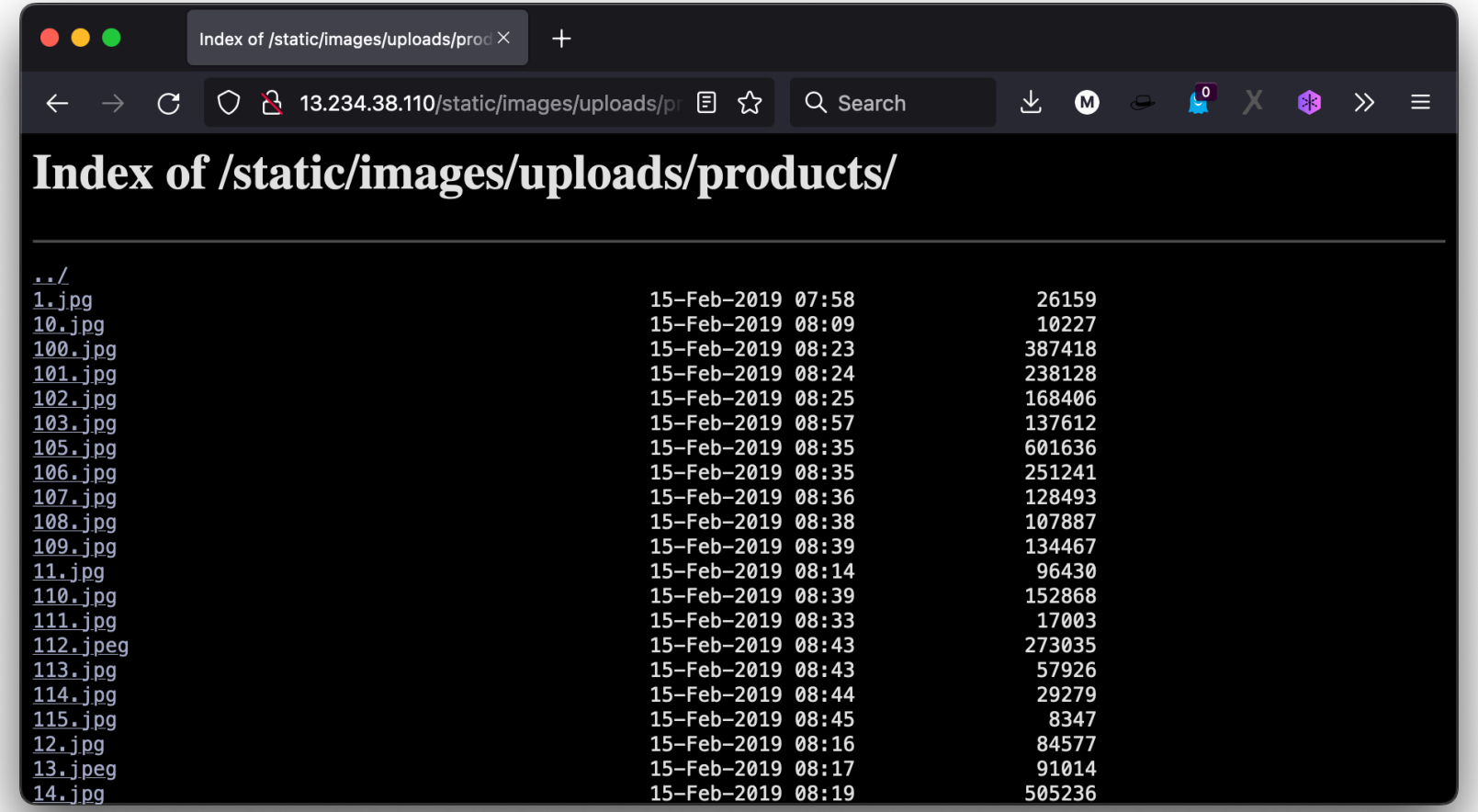
Observation

- Navigate to http://13.234.38.110/products.php.
- Now, right-click on any product image and select View Image, or you can drag the image to a new tab.
- The page loads up as shown below, with the image of the selected product.
- Take note of the URL; it reveals the Image's full path.



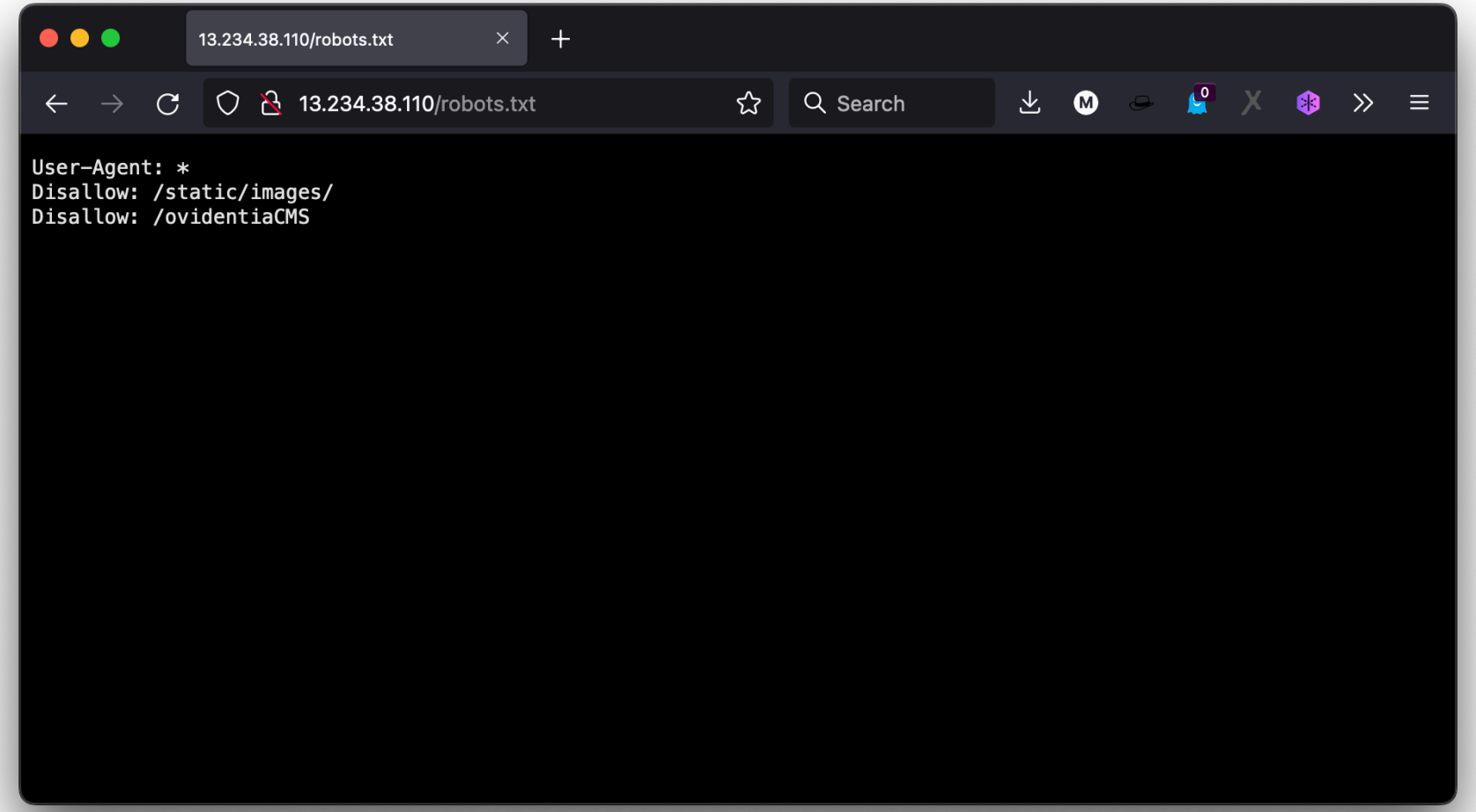
PoC - directory Listing

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



Observation

- Navigate to http://13.234.38.110/robots.txt. It shows all the sections of your server you don’t want robots to use or visit.



Business Impact - High

- Although this vulnerability does not have a direct impact on users or the server, 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 simply download the images, view them, and can even use them against the users or organisation.

Recommendation

Take the following precautions:

- Two-factor authentication for sensitive data should be added with strong passwords.
- Locate all PII stored and encrypt it using various techniques.
- Disable Directory Listing.

References:

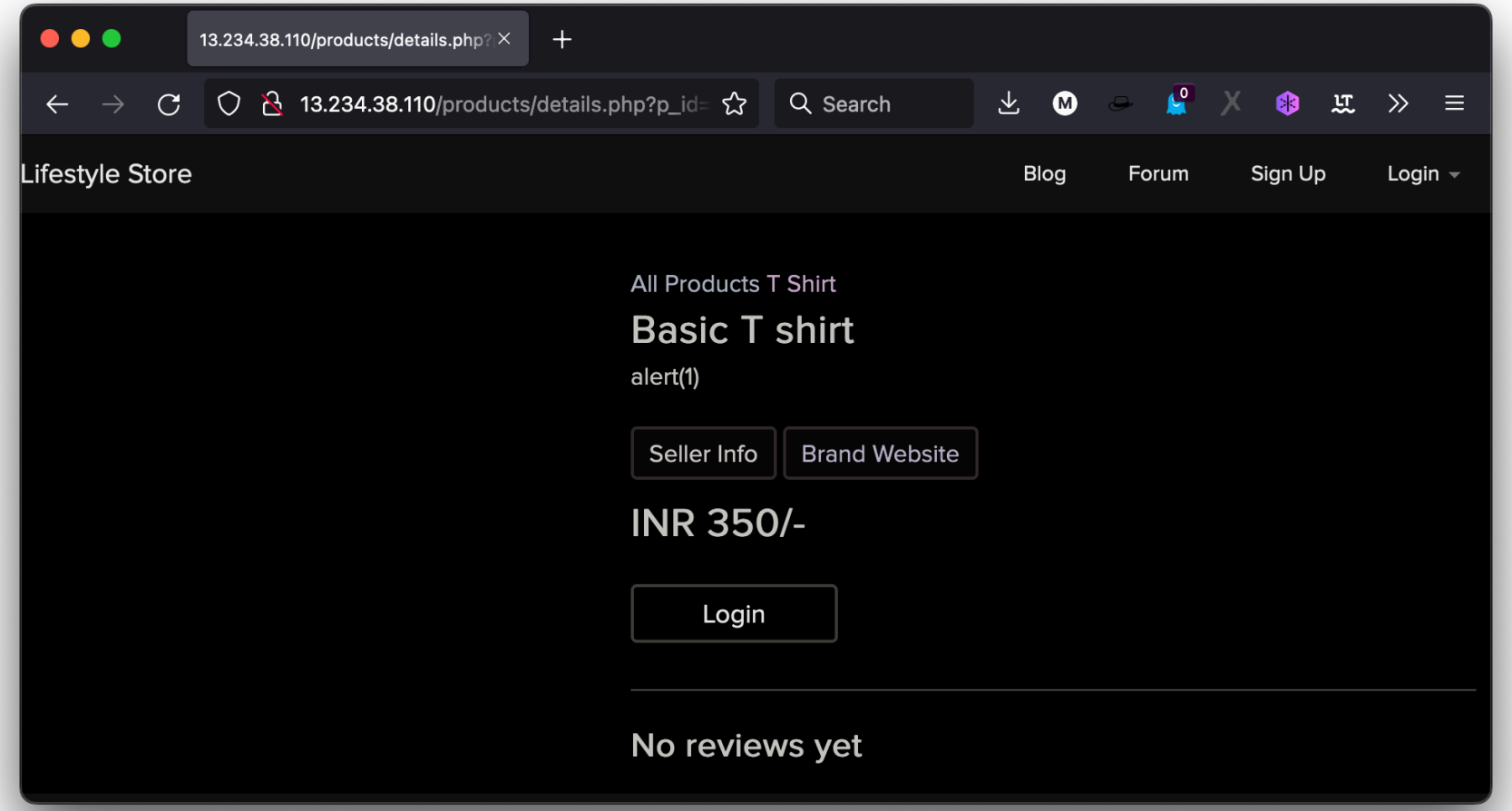
https://cwe.mitre.org/data/definitions/548.html
https://www.netsparker.com/blog/web-security/disable-directory-listing-web-servers/

13. PII Leakage

PII Leakage (Moderate)	Here are some more of link that are vulnerable to Personal Identifiable information leakage. **Affected URL :** • http://13.234.38.110/profile/16/edit

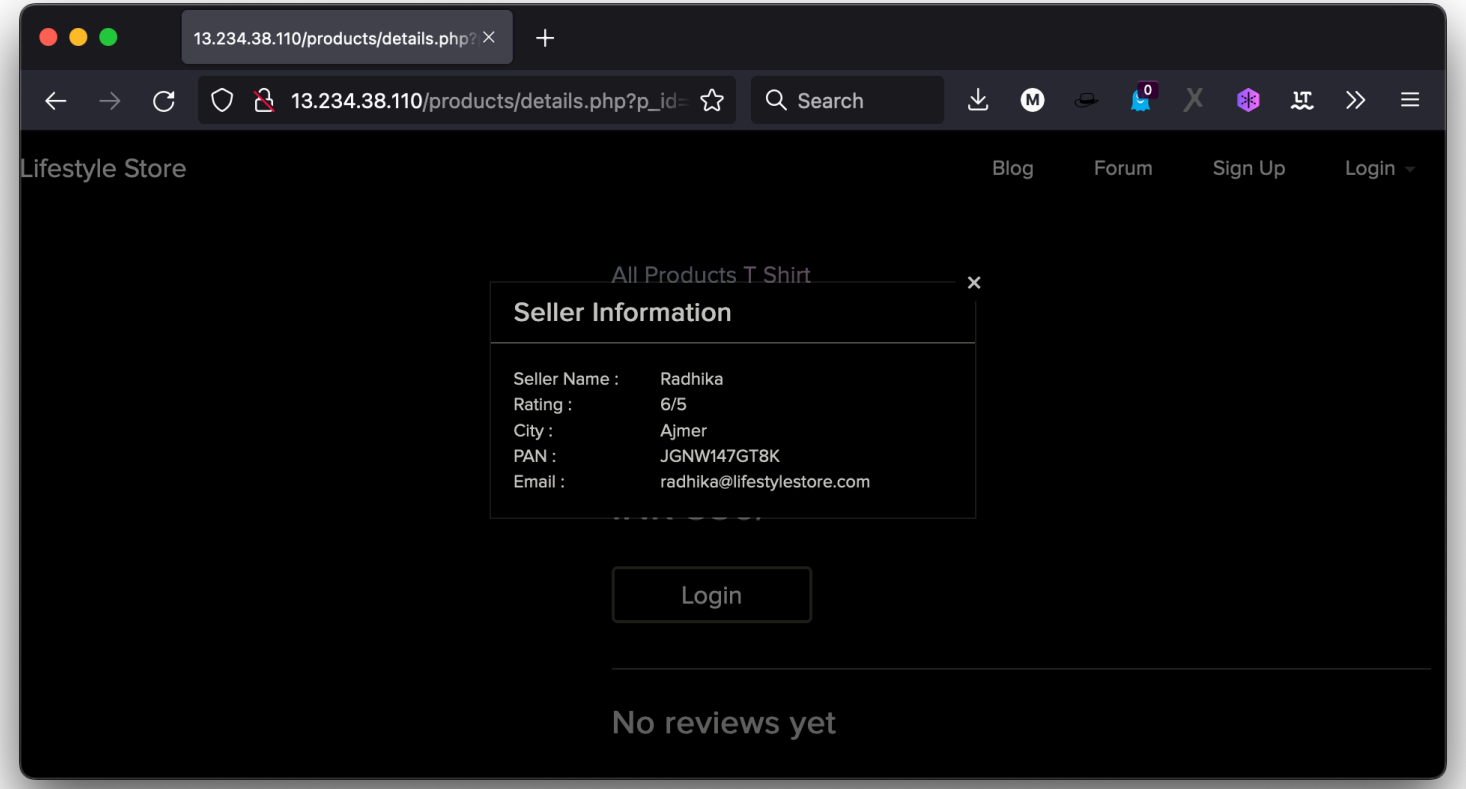
Observation

- Login to your account and go to the Products page. On every product page, the seller's info is available. Click on it.



PoC - Pan Card’s details

- Upon clicking on Seller Info, the Seller Name, Rating, City, and 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 too socially hack them.

Recommendation

Take the following precautions:

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

References:

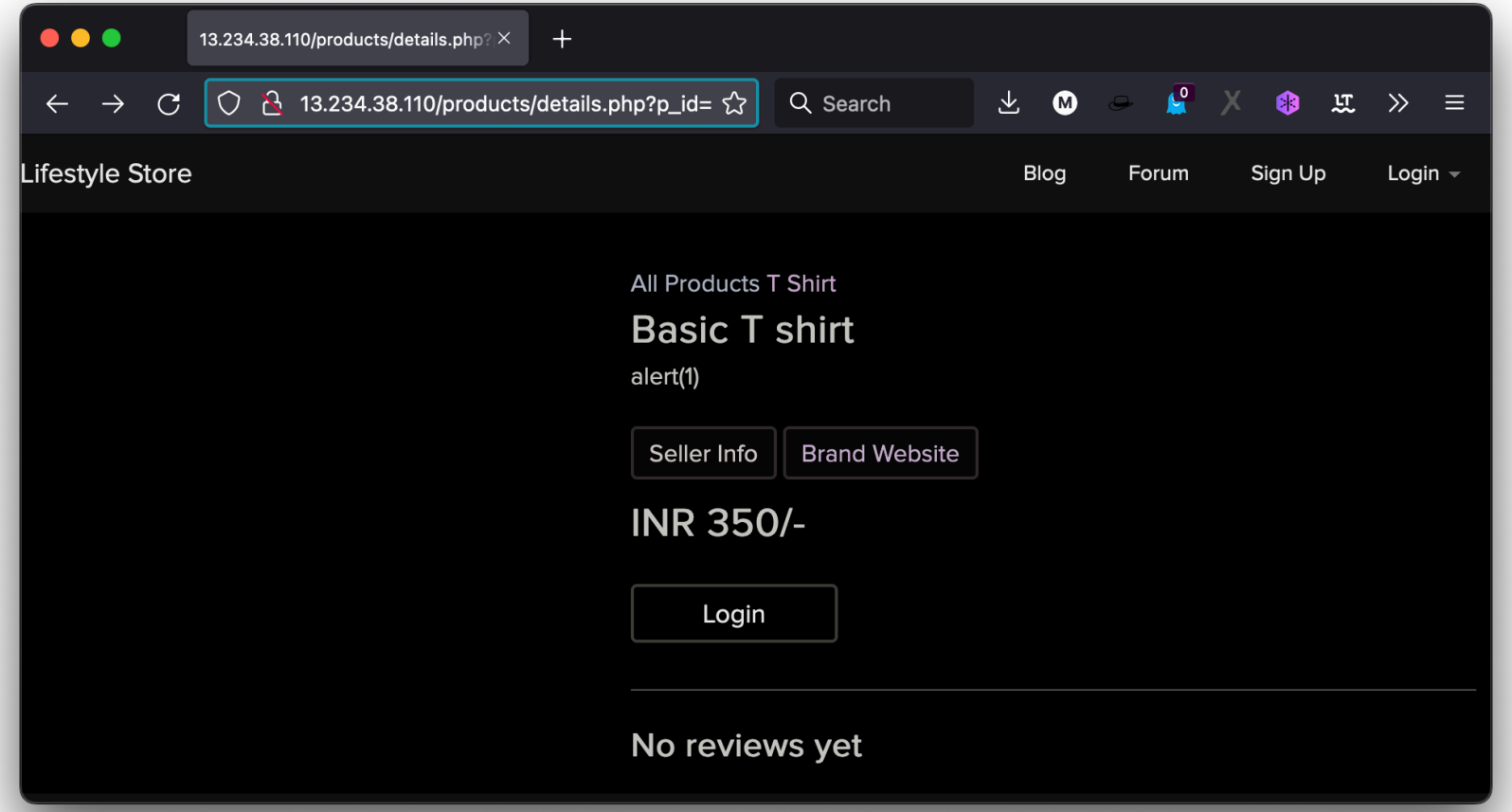
https://www.imperva.com/learn/data-security/personally-identifiable-information-pii/
https://hackerone.com/reports/374007

14. Open Redirection

Open Redirection (Sever)	Below mentioned parameters are vulnerable via Open Redirection. **Affected URL :** • http://13.234.38.110/redirect.php?url=www.radhikafancystore.com **Affected Parameters :** • Url

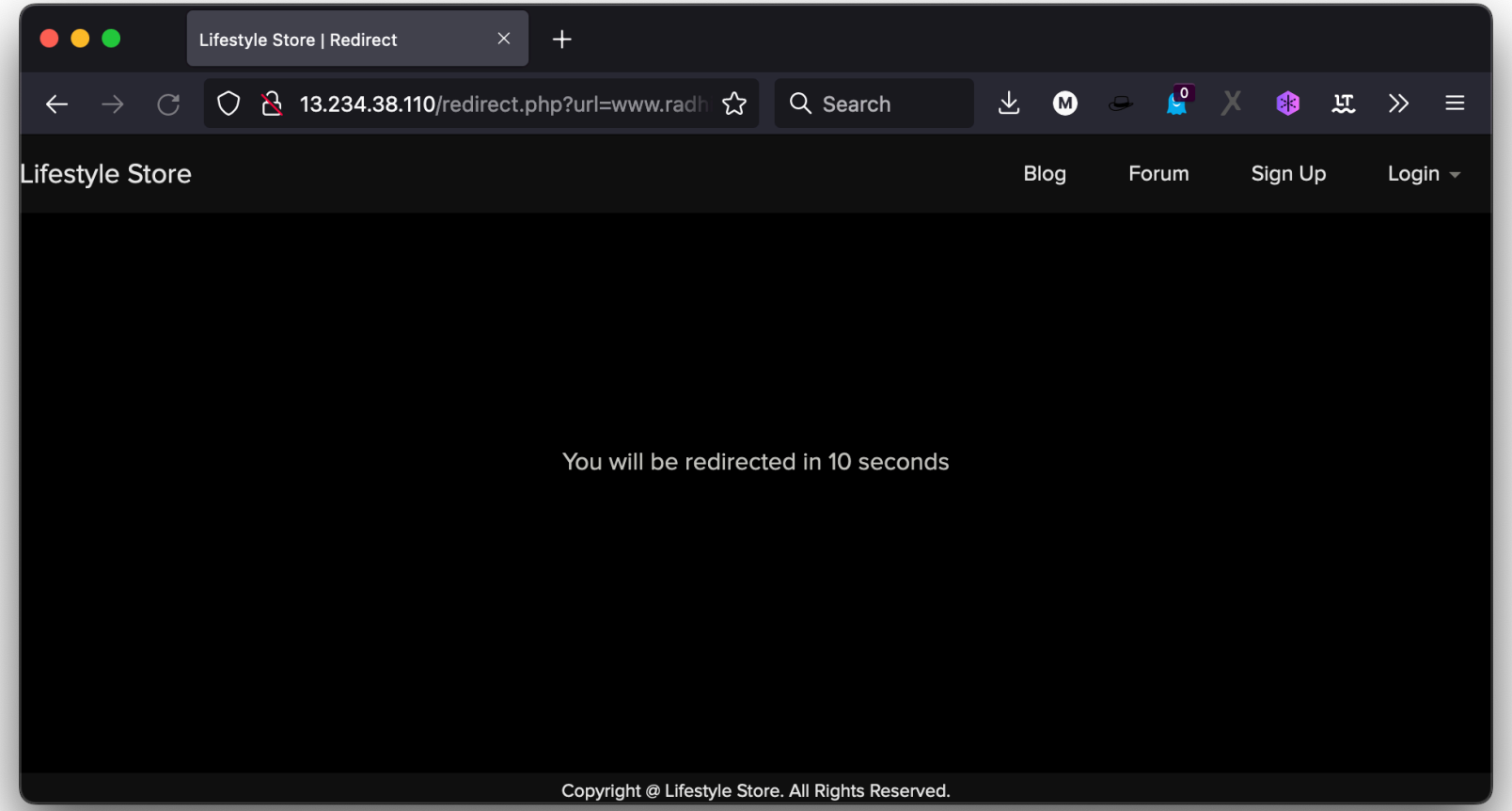
Observation

- Login to your account and go to the Products page. On every product page, the brand website is available. Click on it.



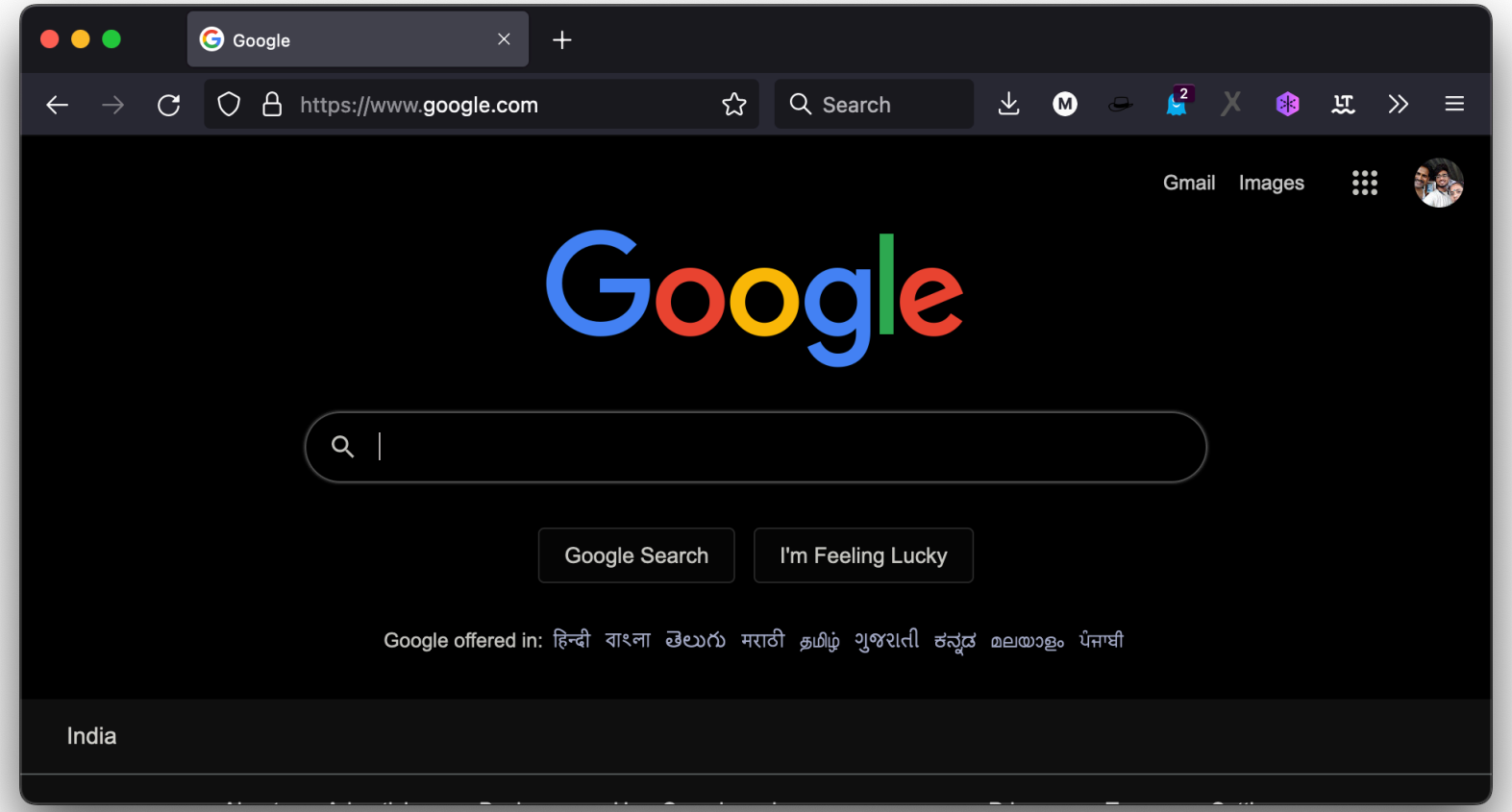
Observation

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



PoC - Using Url for redirection

- We use https://www.google.co.in/ and hit enter after changing the url from the brand website to another website. We redirected to Google.com.



Business Impact - Moderate

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

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.

Reference

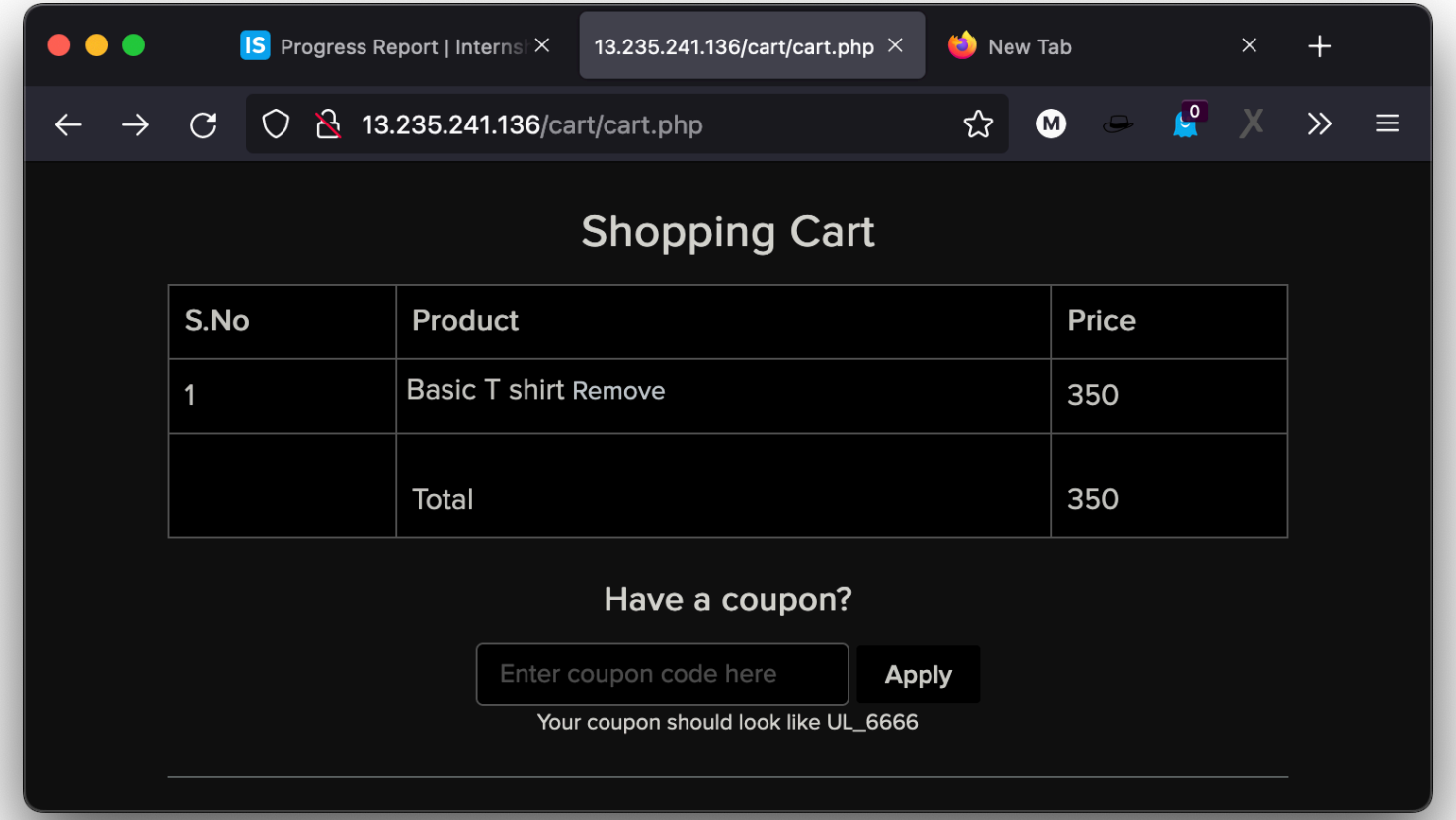
https://www.netsparker.com/blog/web-security/open-redirection-vulnerability-information prevention/
https://spanning.com/blog/open-redirection-vulnerability-web-based-application-security-part-1/
https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/understanding-and discovering-open-redirect-vulnerabilities

15. Brute-force Exploitation

Bruteforce Exploitation (Sever)	Below mentioned parameters are vulnerable via Brute-force exploitation for discount coupon. **Affected URL :** • http://13.235.241.136/cart/cart.php

Observation

- Upon adding items to the cart, you will end up on a screen like this, where we see the applied coupon section and an example.
- Type in UL_6666 in the apply coupon section and intercept the request using Burp Suite.



Observation

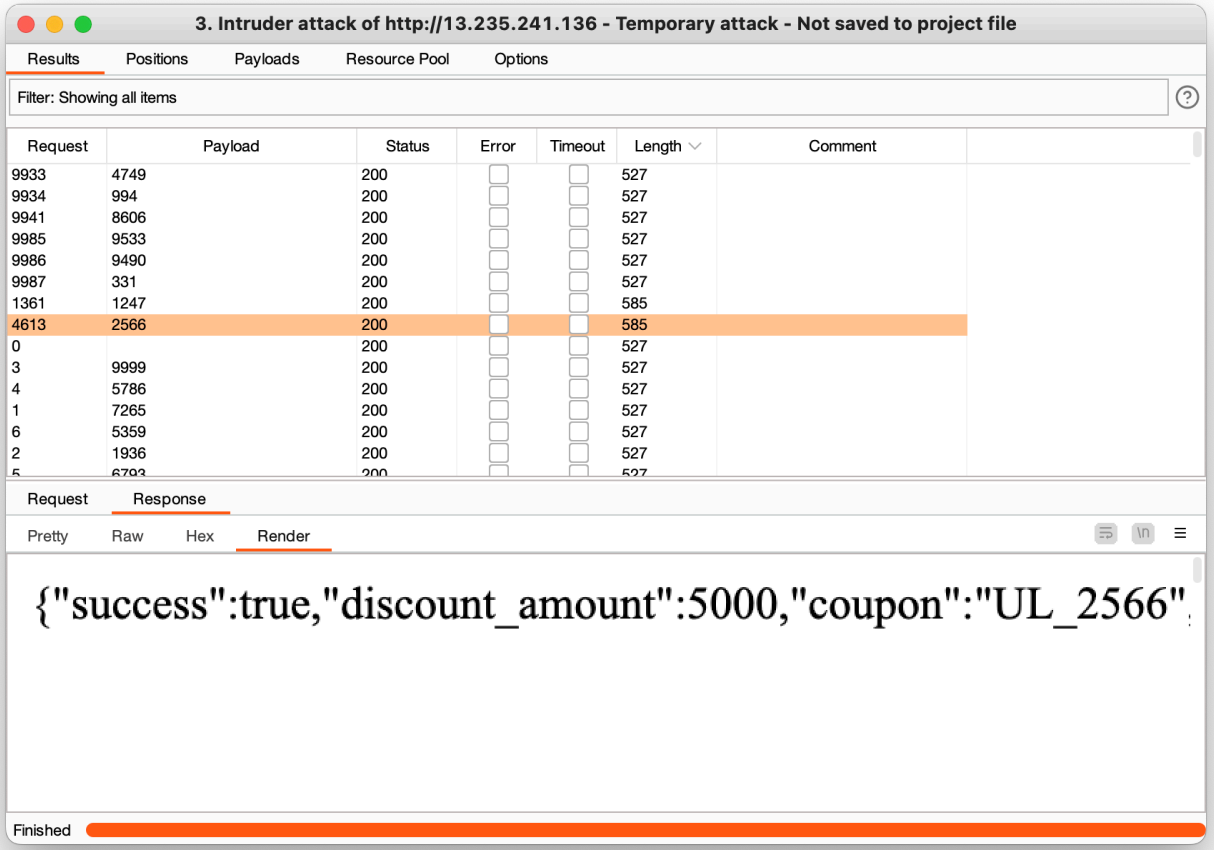
- A following request will be generated containing the coupon code.

Target: http://13.235.241.136

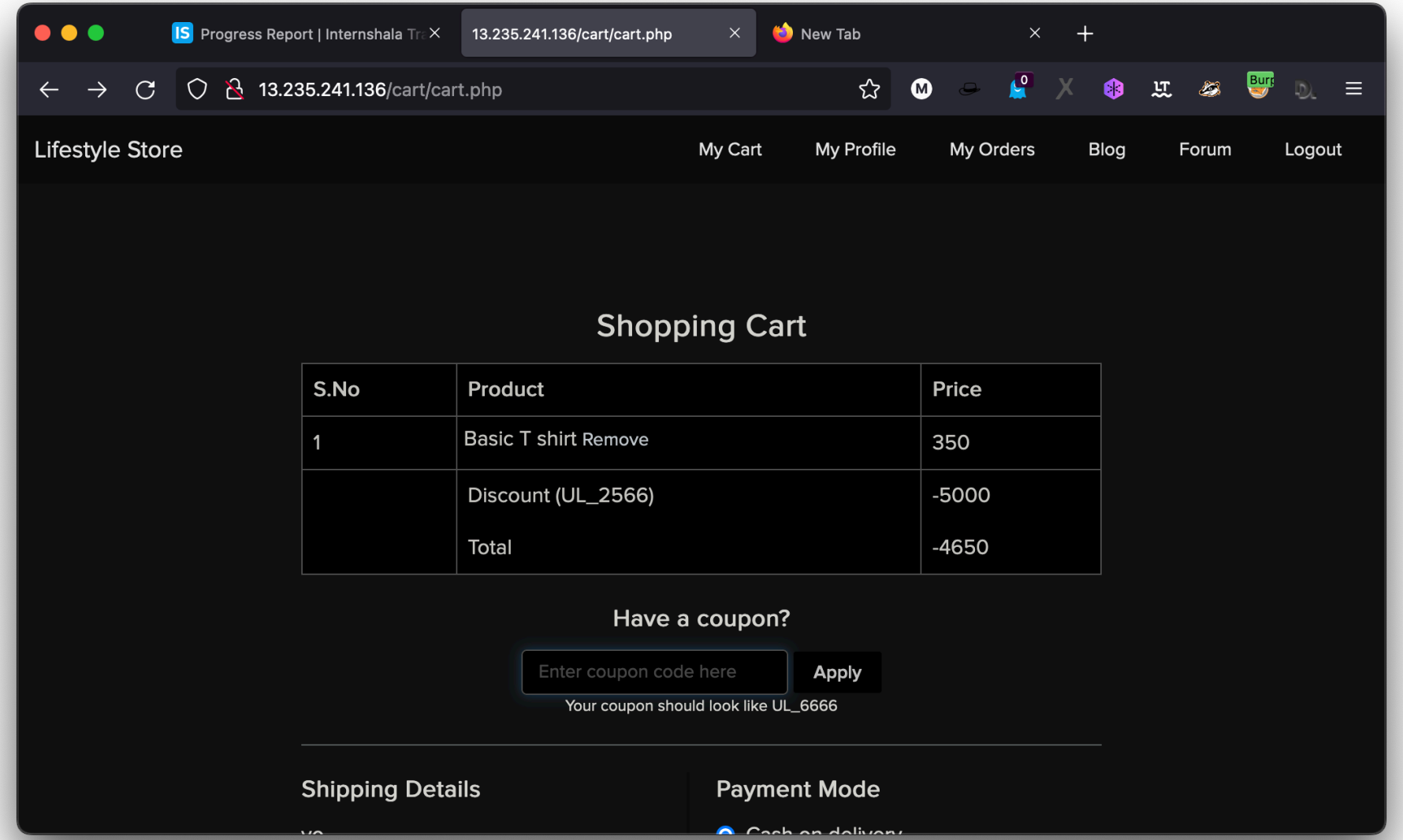
```
POST /cart/apply_coupon.php HTTP/1.1
Host: 13.235.241.136
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 92
Origin: http://13.235.241.136
DNT: 1
Connection: close
Referer: http://13.235.241.136/cart/cart.php
Cookie: key=1FB4A8A4-C7D5-A28B-9345-F0817ADA5770; PHPSESSID=vq176tl9g99ncalmef4piugh63; X-XSRF-TOKEN=
35a2d49c039a29967be388ea2162fafed8268920dbe056301df4676f48bf8b9c
Sec-GPC: 1

coupon=UL_§6666§&X-XSRF-TOKEN=35a2d49c039a29967be388ea2162fafed8268920dbe056301df4676f48bf8b9c
```

Observation

- We shoot the request with all possible combinations of 4 digits and, upon a successful hit, we get a response containing the valid coupon code.
- We can use this code to get the discount.
- The valid coupon code for this website is UL_2566.



PoC - Coupon Code Applied



Business Impact – High

Attackers can easily order the items at discounts, which in turn will cause a huge loss to the company.

Recommendation

Take the following precautions:

- Coupon codes should have a limited number of uses and should be regenerated after a while.
- The coupon code should be random alpha-numeric characters.

References:

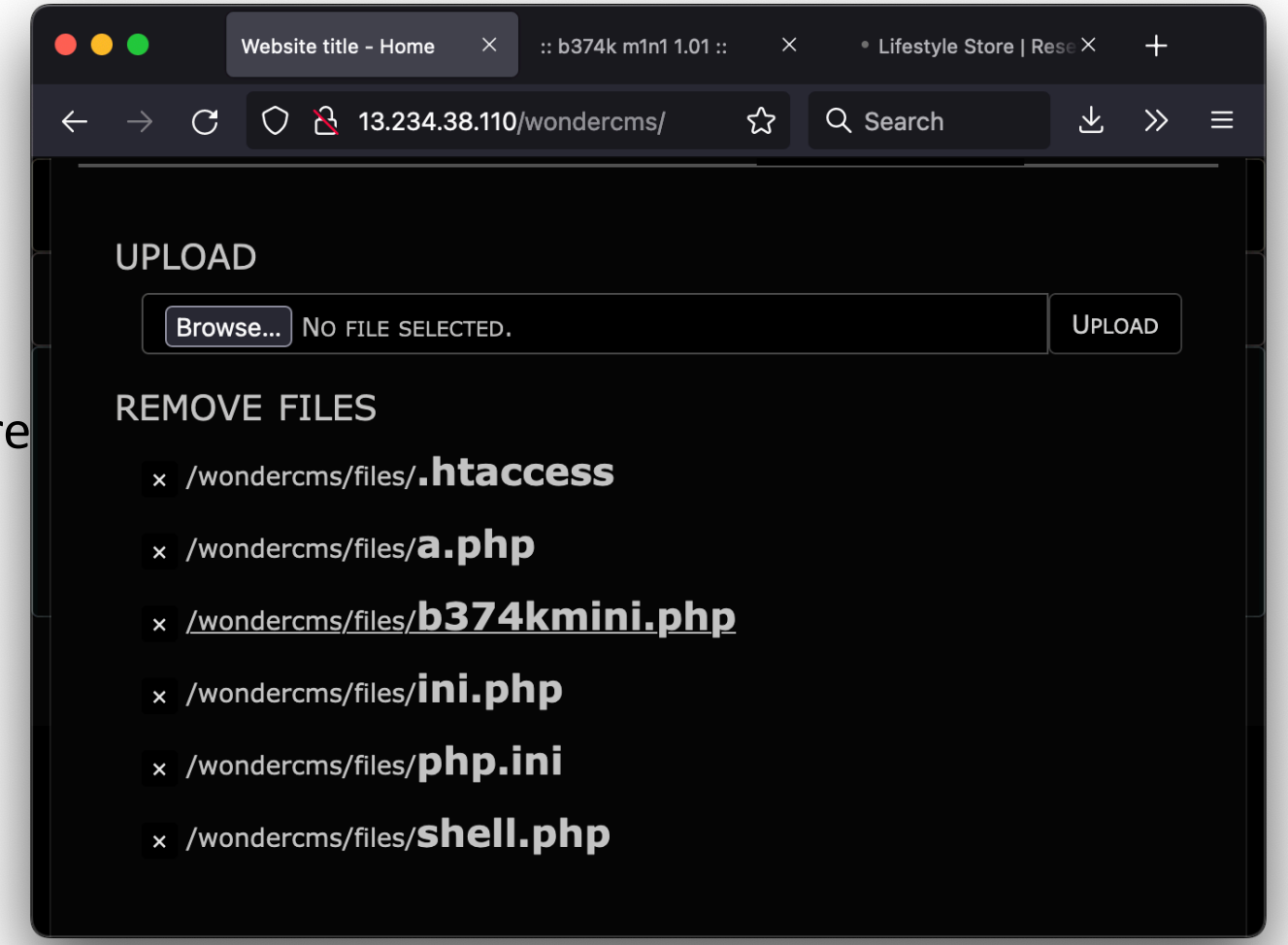
https://www.couponxoo.com/brute-force-attack-coupon-code

16. Command Executions Vulnerability

Command Execution Vulnerability (Critical)	The Command Executions Vulnerability can be found in below URL. **Affected URL :** • http://13.234.38.110/wondercms/files/b374kmini.php • http://13.234.38.110/admin31/console.php

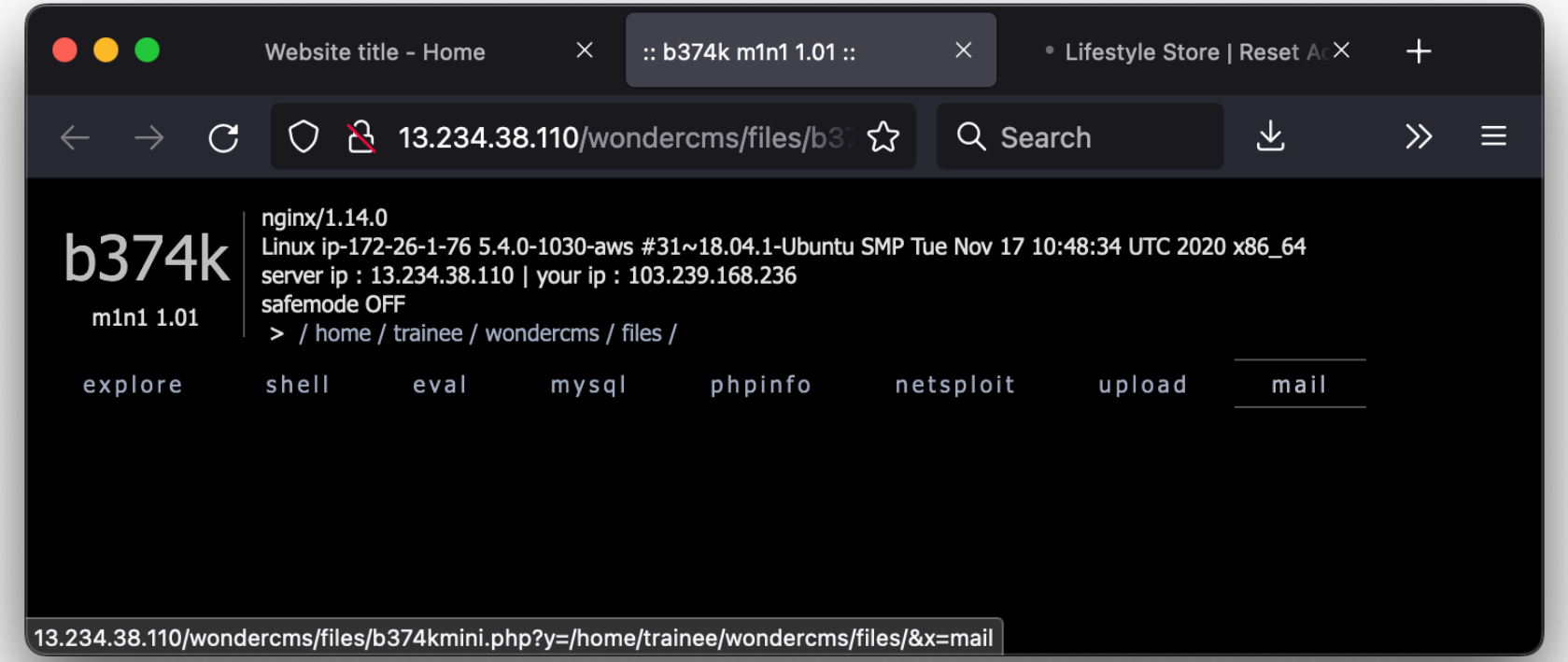
Observation

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



Observation

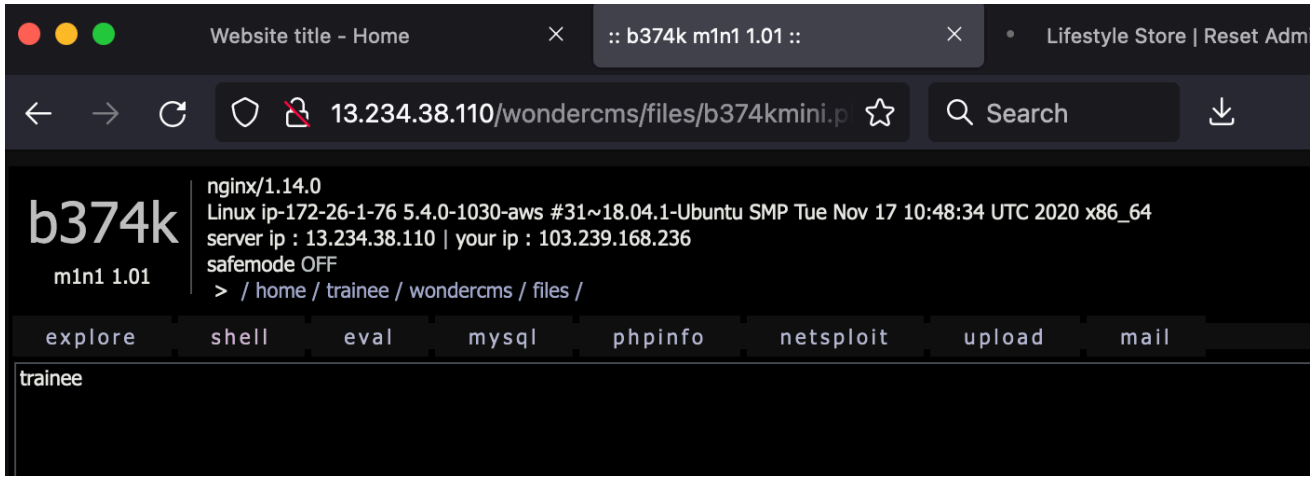
- It looks like this is a small and simple PHP-shell that has an explorer, allows shell command execution, mysql queries and more.



PoC - Command Execution

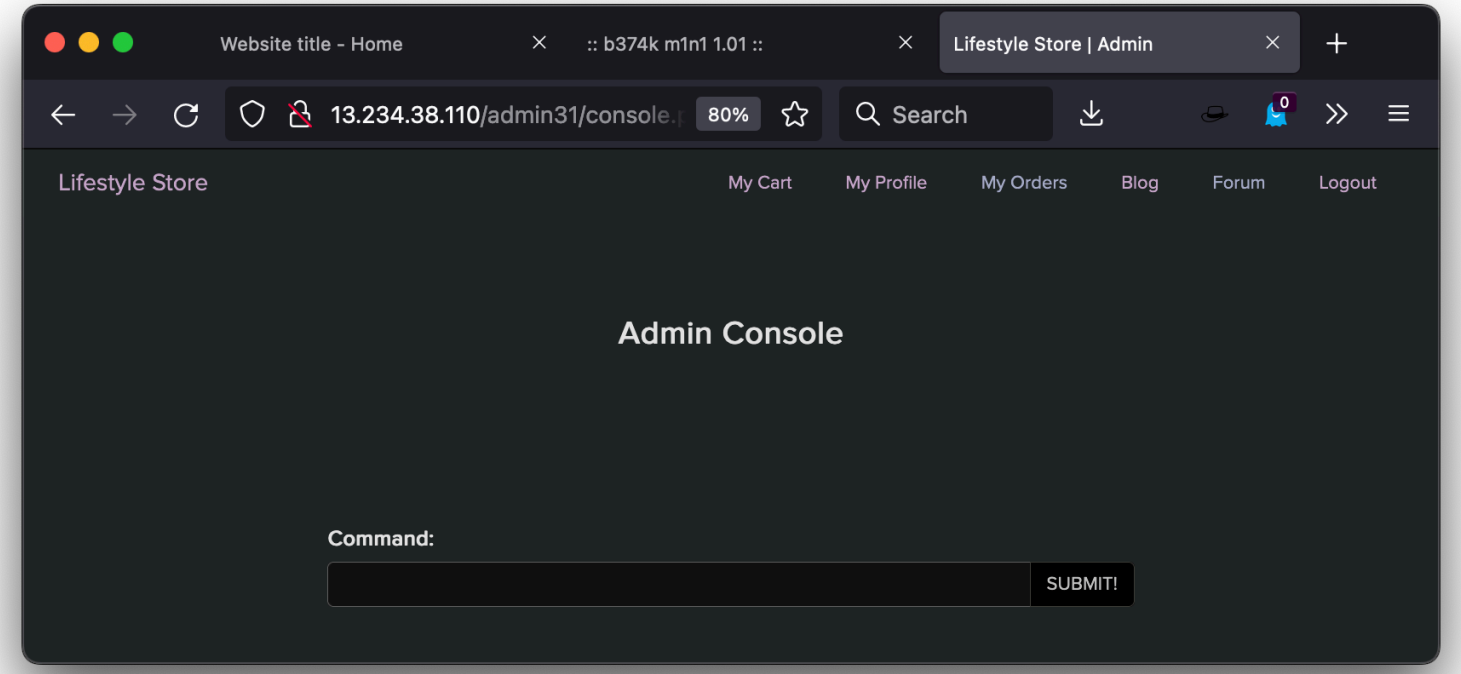
- Type in the command: "whoami" and press “Go!"

- The command was executed successfully.



Observation

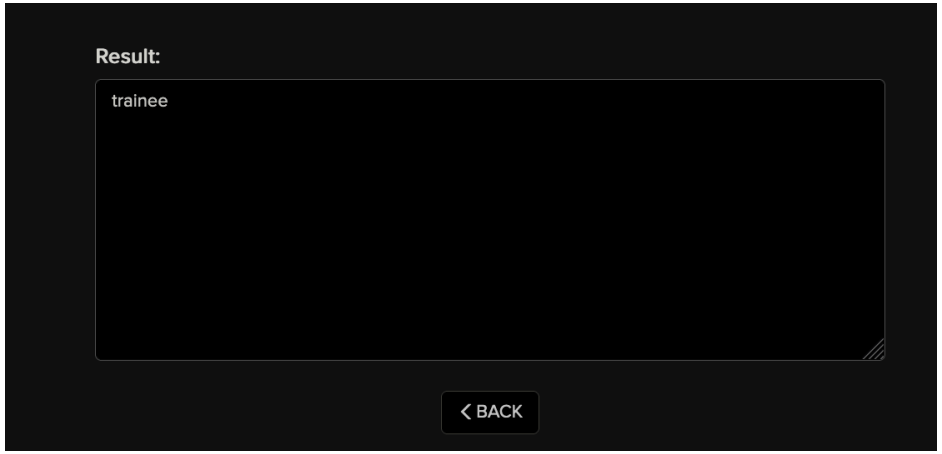
- As a customer, Login to your account.
- Now, type the following URL into your browser: http://13.234.38.110/admin31/console.php (you will come to know about this url while testing vulnerabilities for Vulnerability Report No. 4, Rate Limiting Flaws), and press enter.



PoC - Command Execution

- It seems like we can execute commands here. Let’s try by typing "whoami" and pressing “SUBMIT!”.

- Executed Successfully .



Business Impact - Extremely High

- Including complete system takeover, an overloaded file system, or a database.
- Forwarding attacks to back-end systems.
- client-side attacks or simple defacement.

Recommendation

Take the following precautions:

- Hide all the files on UPLOAD Screen.
- Get rid of the PHP shell's.

References:

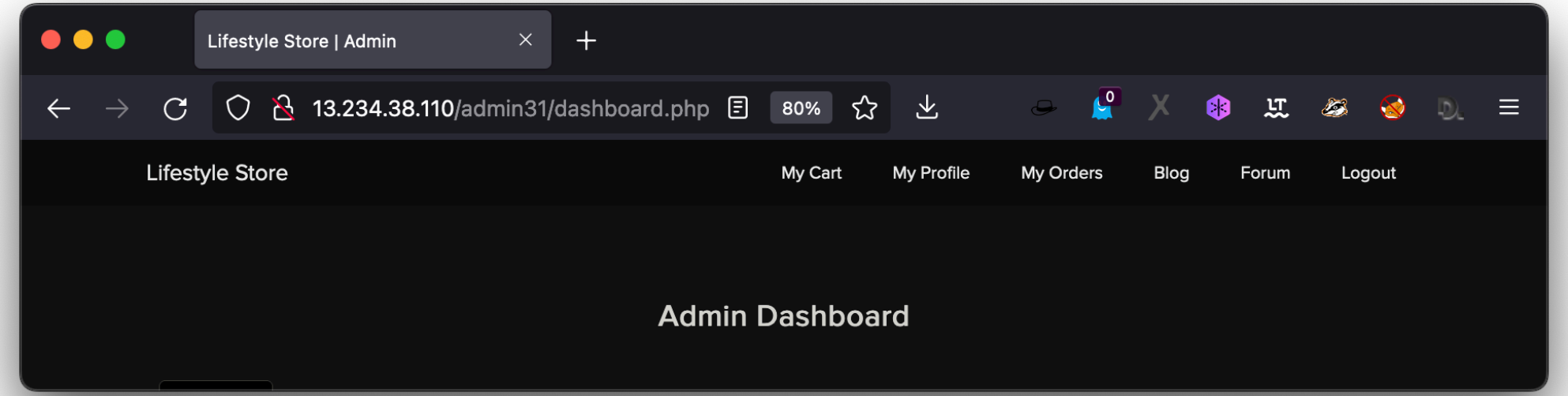
https://miniphpshell.wordpress.com/2009/10/13/b374k-mini-shell/
https://owasp.org/www-community/attacks/Command_Injection

17. Forced Browsing

Forced Browsing (Sever)	Below mentioned parameters are vulnerable via Open Redirection. **Affected URL :** • http://13.234.38.110 **Forced URL:** • http://13.234.38.110/admin31/dashboard.php • http://13.234.38.110/admin31/console.php

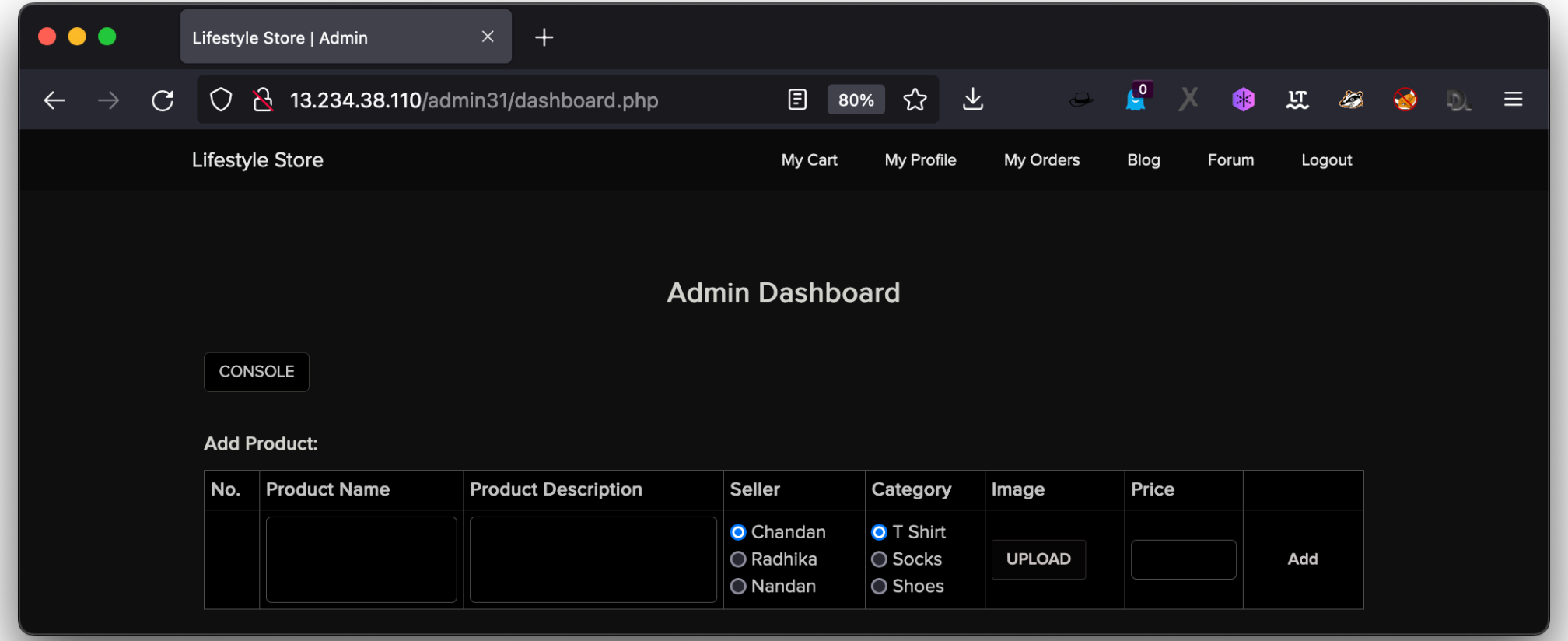
Observation

- As a customer, log in to your account.
- Now, type the following URL into your browser: http://13.234.38.110/admin31/dashboard.php(I came to know about this URL while testing vulnerabilities for Vulnerability Report No. 4, Rate Limiting Flaws.)



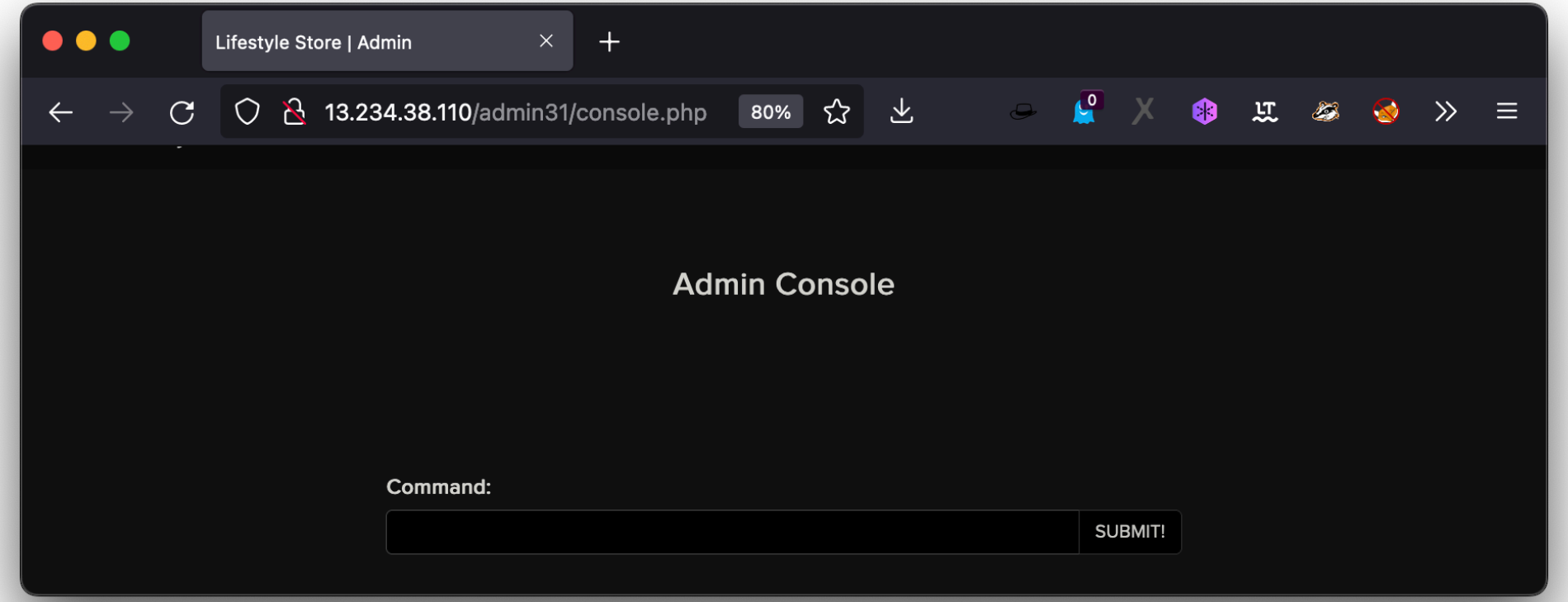
PoC - Admin Dashboard Access

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



PoC - Admin console Access

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



Business Impact – Sever

An attacker can have all the admin privileges like edit all the items, ability to run any malicious command from the console.

Recommendation

Take the following precautions:

- Server side security checks should be performed perfectly.
- Make the admin page url complicated so that it can’t be guessed

References:

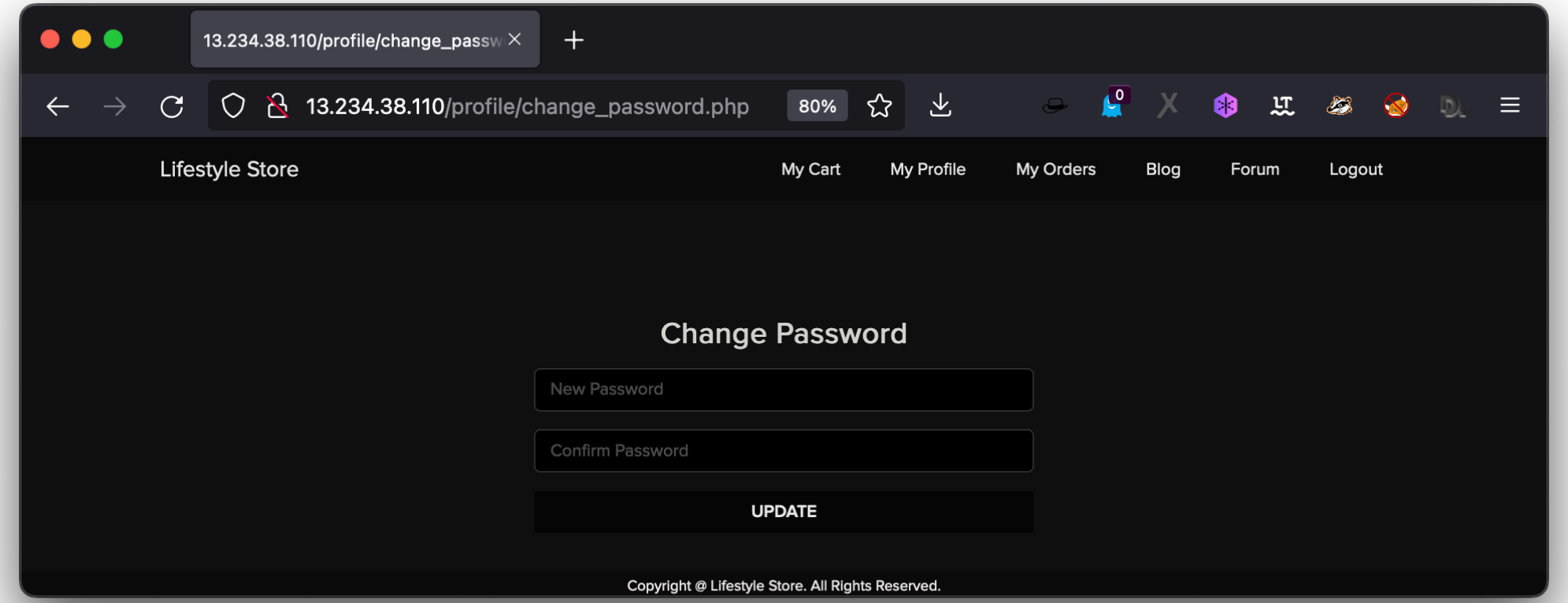
https://owasp.org/www-community/attacks/Forced_browsing
https://campus.barracuda.com/product/webapplicationfirewall/doc/42049348/forced-browsing attack/

18. Cross-Site Request Forgery

Cross-Site Request Forgery (Sever)	Below mentioned parameters are vulnerable via Cross-Site Request Forgery. **Affected URL :** • http://13.234.38.110/profile/change_password.php •

Observation

- As a customer, Login to your account.
- A change password page appears. Go to the My Profile section and click on the Change Password button. Let’s see if wea can forge the request some how. Let’s try it by creating an HTML page.



PoC - Change Password

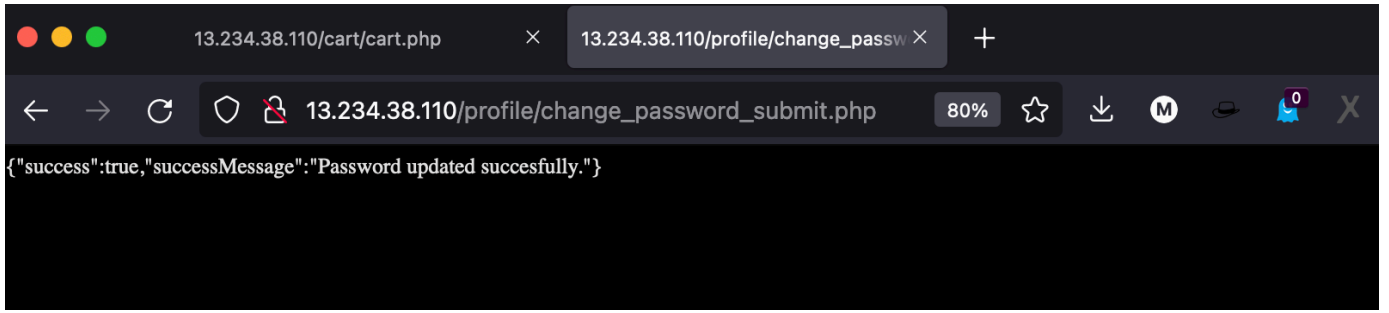
- Now, make an HTML page to update or change your password.

html_password.html

Users > black..... > Downloads > html_password.html > html

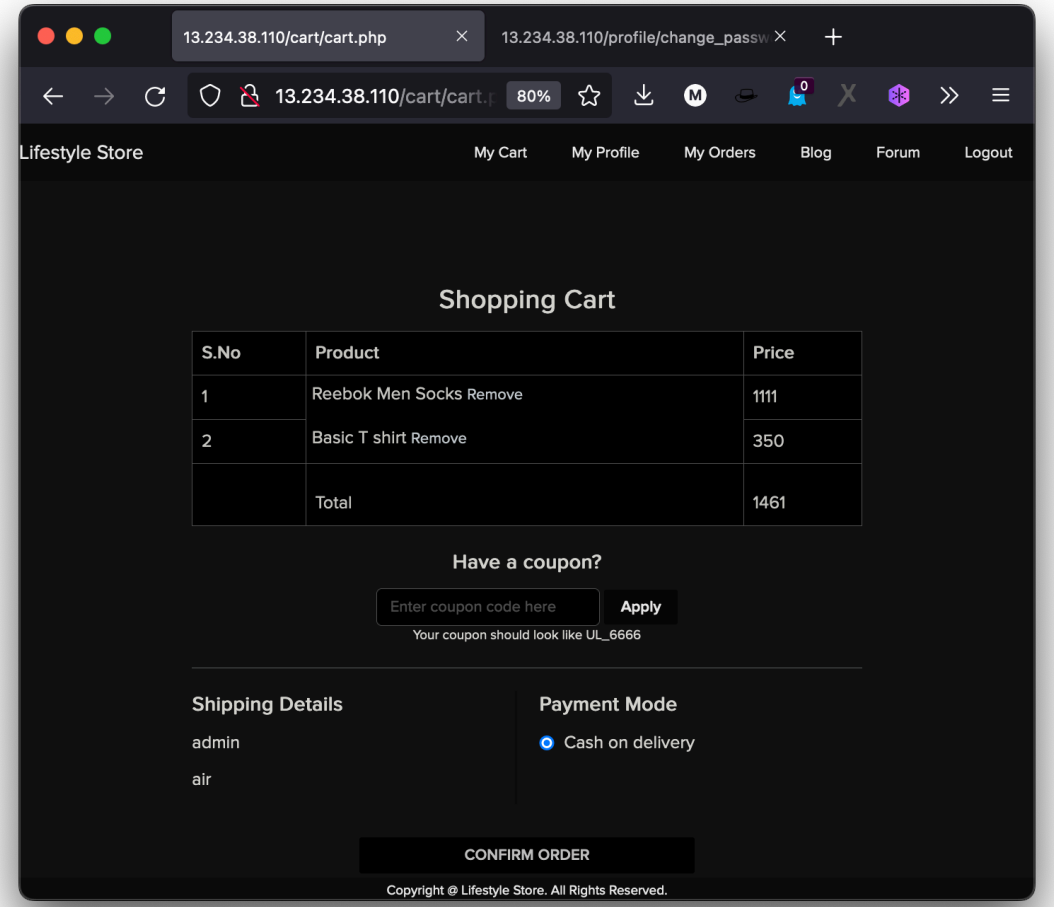
```
<html>
<head>
<title> CSRF POC </title>
</head>
<body>
    <form name='change-password' id='change-password' method='POST' action='http://13.234.38.110/profile/change_password_submit.php'>
    <input type='password' placeholder="New Password" name="password" id="password" value="1234">
    <input type='password' placeholder="Confirm Password" name="password_confirm" id="password_confirm" value="1234">
    <button type='submit' class="btn btn-primary">Update</button>
</body>
</html>
```

- On typing in a new set of passwords and clicking on the Update button, we get a Success Message. 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.
- Look for any product and add it to your shopping 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.



PoC - Order Confirmation Code

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

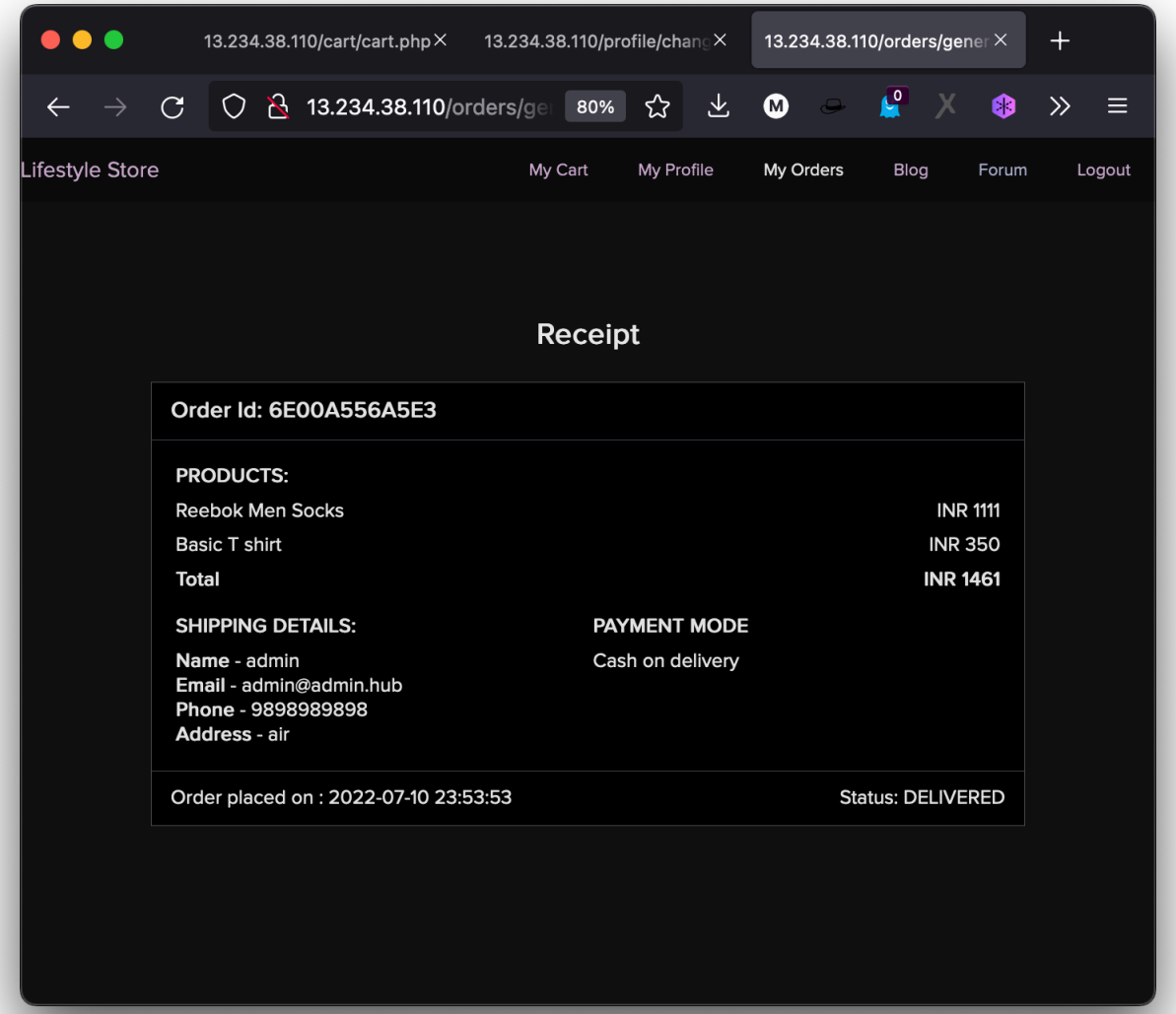
Html_cart.html

Users > black..... > Downloads > Html_cart.html > html

```
1   <html>
2   <head>
3   <title> CSRF POC </title>
4   </head>
5   <body>
6       <form action="http://13.234.38.110/orders/confirm.php" method='POST'>
7       <input type='Submit' value="Submit Request"></input>
8   </body>
9   </html>
10
11
```

PoC - Order Confirmation

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



Business Impact - Sever

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

Recommendation

Take the following precautions:

- Use tokens and session cookies.
- Ask the user for his password (temporary like OTP or permanent like login password) at every critical action like while deleting an 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>.
- Check the reference before carrying out any 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:

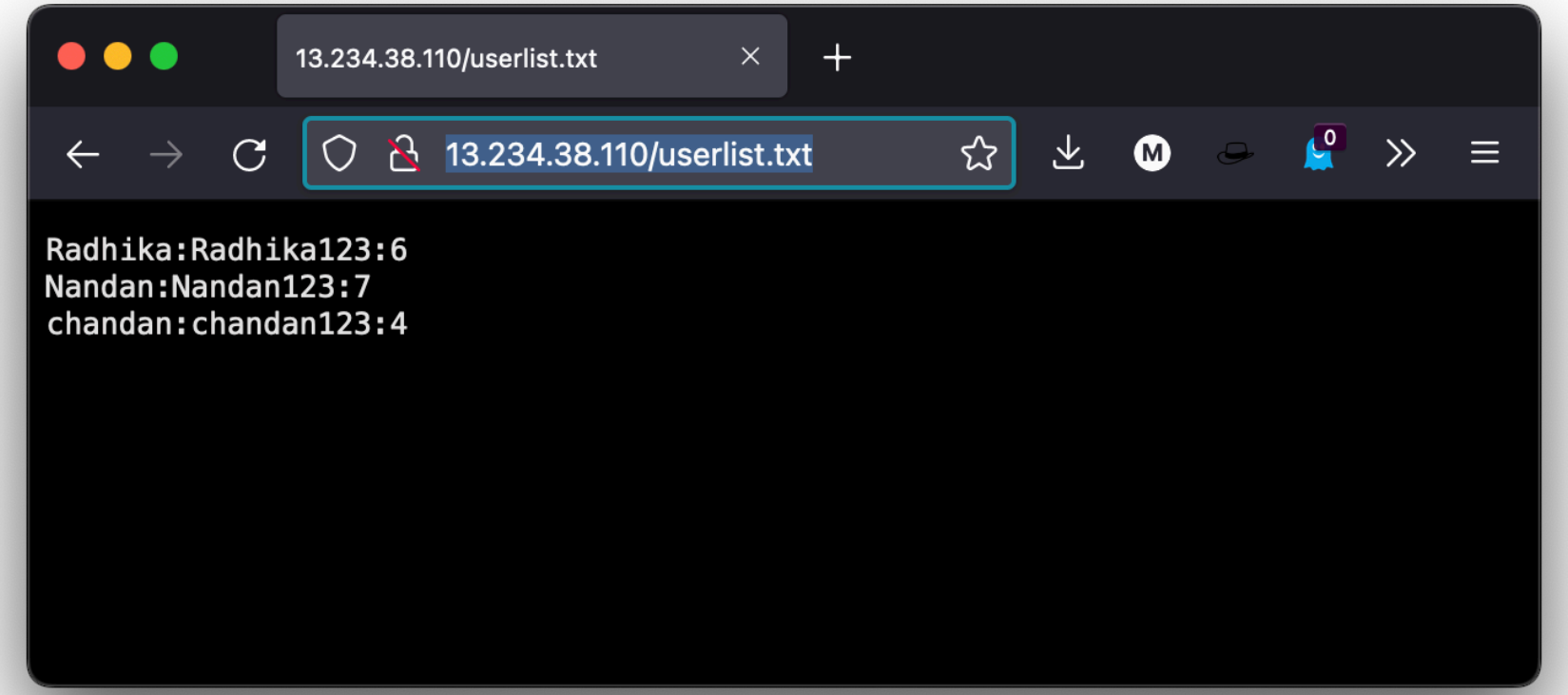
https://owasp.org/www-community/attacks/csrf
https://en.wikipedia.org/wiki/Cross-site_request_forgery
https://portswigger.net/web-security/csrf

19. Seller Account Access

Seller Account Access (Critical)	The Seller Account Access is vulnerable by consumer. **Affected URL :** • http://13.234.38.110/userlist.txt

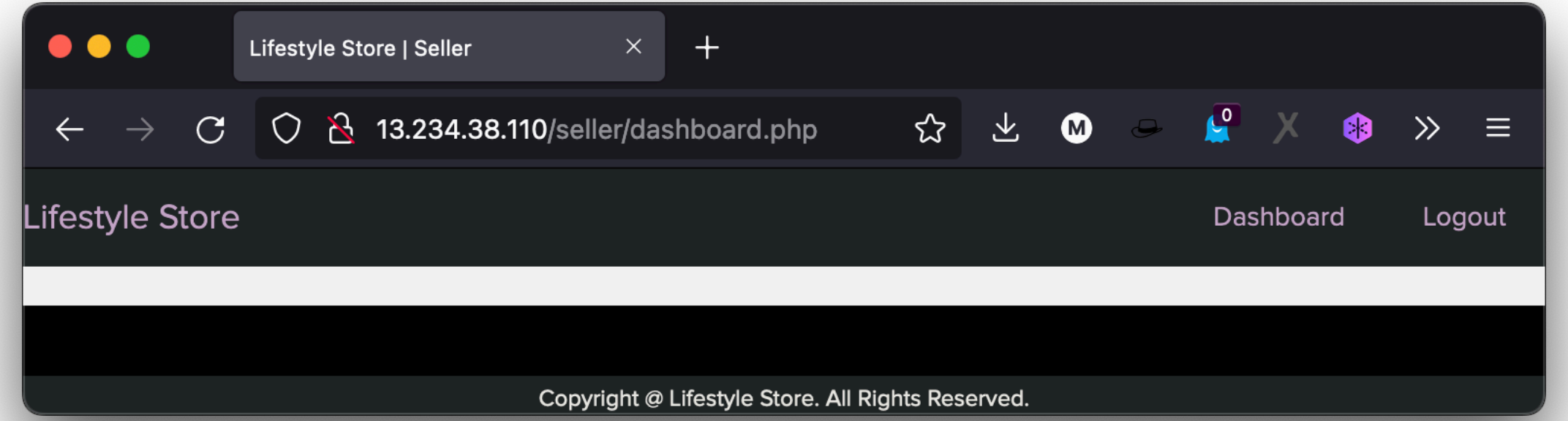
Observation

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



PoC - Seller dashboard

- On entering the credentials in the seller account we got from http://13.234.38.110/userlist.txt, we accessed the seller's dashboard.



Business Impact - Extremely High

- An attacker can access the seller's dashboard and then 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

Take the following precautions:

- The developer should disable these confidential default pages, which reveal 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:
mauryazammer786@gmail.com