ETHICAL HACKING
PROJECT REPORT

SUBMITTED BY:

TEEYA OJHA

17001012021

BRANCH:

Bachelors of technology in

COMPUTER SCIENCE AND ENGINEERING

STUDENT OF:

INDIRA GANDHI DELHI TECHNICAL UNIVERSITY FOR WOMEN

KASHMERE GATE, DELHI - 110077

ACKNOWLEDGEMENT

I would like to thank the platform internshala for providing the course, the knowledge and the environment (the hacking labs) for learning and completing the course successfully.
I will also like to thank my family and friends for supporting me through the process and helping me wherever needed. They also helped me stay focused on my goals and helped me in achieving them.

Teeya Ojha
17001012021
BTech CSE

INDEX

INTERNSHALA TRAININGS

Certificate of Training

Teeya Ojha

from Indira Gandhi Delhi Technical University for Women has successfully completed an 8-week online training on **Ethical Hacking**. In the training, Teeya learned Basics of Information Security, Computer Networking and Web Development, Information Gathering and VAPT of some important vulnerabilities in the OWASP top 10, Automating VAPT, and Documenting and Reporting Vulnerabilities.

In the final assessment, Teeya scored 68% marks.

We wish Teeya all the best for future endeavours.

Sarvesh Agarwal

FOUNDER & CEO, INTERNSHALA

Date of certification: 2022-08-30 Certificate no. : 34C9A6F9-31CA-8050-D374-2D7C0F9F8F5A

For certificate authentication, please visit https://trainings.internshala.com/verify_certificate

DECLARATION

I hereby declare that this project based on **ethical hacking** has been carried out by my own efforts and facts arrived at by my own observations. I am hence submitting this project to my college and I also promise that this project has not been submitted in any university before. As this is my original work.

Name: **Teeya Ojha**
Date: **30th Aug, 2022**

Abstract

This paper explores the ethics behind ethical hacking and whether there are problems that lie with this new field of work. Since ethical hacking has been a controversial subject over the past few years, the question remains of the true intentions of ethical hackers. The paper also looks at ways in which future research could be looked into to help keep ethical hacking, ethical.

INTRODUCTION

In this course, we were given videos to learn from. There were a total of 9 chapters in the course and each chapter was divided into modules. Each module had a small test after its completion and after each chapter, we had to attempt a test. Without which we won't be able to move to the next chapter. I made written notes while watching all the videos which later helped me, where needed. Doing this also helped me preserve this knowledge with myself forever.
At the end of the course, we were given a problem statement. Based on which, we had to create the project.
Internshala also provided us with hacking labs for real life experience and for project competition, we were provided a website we had to work on.

PROBLEM STATEMENT

ETHICAL HACKING TRAINING

PROBLEM STATEMENT

We are glad that you have completed the training and cleared the final test. Now, it's time to test your skills in a practical manner and for that, we have setup a real life-like web application in the form of an online e-commerce portal.

Your task is to test this e-commerce platform and find all possible vulnerabilities and loopholes in it, collect relevant PoCs and then prepare a Detailed Developer Level Report.

For reporting each vulnerability, you must follow the sample report given to you in Module 8 and make sure the following things are mentioned:

- Title of Vulnerability.
- A Short Description.
- Exact URL which has the vulnerability.
- The parameters which are vulnerable (with parameter type like GET, POST, Cookie, Header, etc.).
- Payload that you used to trigger the vulnerability.
- Observation slides containing step by step information to replicate the exploit with PoCs.
- Business Impact of the vulnerability, explaining in detail what can be done by a hacker.
- Recommendations on how to fix the vulnerability.
- Reputed References for the vulnerabilities.

Remember, each and every kind of vulnerability you learnt about, might be somewhere in this application. All you have to do is open the application and start exploring its features. Once you have understood each feature the website has, you can start playing around with it and fuzzing into various places.

A big part of the VA has been already done for you as you have the exact IP and the application which you have to test, but there could be hidden pages and components too, so keep that in mind.

To give you a benchmark and a target to achieve, here is a list of all the vulnerabilities which we have intentionally kept and which are supposed to be found and reported by you:

SQL Injection
Reflected and Stored Cross Site Scripting
Insecure Direct Object Reference
Rate Limiting Issues
Insecure File Uploads
Client Side Filter Bypass
Server Misconfigurations
Components with Known Vulnerabilities
Weak Passwords
Default Files and Pages
File Inclusion Vulnerabilities
PII Leakage
Open Redirection
Bruteforce Exploitation
Command Execution Vulnerability
Forced Browsing Flaws
Cross-Site Request Forgery

So, there are a total of 28 vulnerabilities (some vulnerabilities have more occurences than 1) intentionally kept but these do not include combinational vulnerabilities like Bruteforce Exploitation and Rate Limiting. If you are able to guess the password, you can either count it in Bruteforcing or count it in rate limiting but yes, while writing recommendations, write recommendations for both. Similarly, if you find a public software that allows PHP file upload, you can either count it in file upload or in Components with known vulnerabilities.

If you do find other general vulnerabilities apart from these you can report them too but do not count them in the 28.

Happy bug hunting!

Steps to access the Project:

1. Login to trainings.internshala.com
2. Go to Ethical Hacking Training
3. Go to Progress Tracker
4. Click on the 'GO TO PROJECT WEB APPLICATION' button.

Necessary scans

Zenmap

Scan Tools Profile Help

Target: 3.6.37.186 Profile:

Command: nmap -T4 -A -v -Pn 3.6.37.186

Hosts Services

OS ◂ Host

ec2-3-6-37-186.ap-

Nmap Output Ports / Hosts Topology Host Details Scans

nmap -T4 -A -v -Pn 3.6.37.186

```
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-08 14:36 India Standard Time
NSE: Loaded 151 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 14:36
Completed NSE at 14:36, 0.00s elapsed
Initiating NSE at 14:36
Completed NSE at 14:36, 0.00s elapsed
Initiating NSE at 14:36
Completed NSE at 14:36, 0.00s elapsed
Initiating Parallel DNS resolution of 1 host. at 14:36
Completed Parallel DNS resolution of 1 host. at 14:36, 0.98s elapsed
Initiating SYN Stealth Scan at 14:36
Scanning ec2-3-6-37-186.ap-south-1.compute.amazonaws.com (3.6.37.186) [1000 ports]
Discovered open port 80/tcp on 3.6.37.186
Discovered open port 22/tcp on 3.6.37.186
Completed SYN Stealth Scan at 14:37, 15.59s elapsed (1000 total ports)
Initiating Service scan at 14:37
Scanning 2 services on ec2-3-6-37-186.ap-south-1.compute.amazonaws.com (3.6.37.186)
Completed Service scan at 14:37, 6.77s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against ec2-3-6-37-186.ap-south-1.compute.amazonaws.com (3.6.37.186)
Retrying OS detection (try #2) against ec2-3-6-37-186.ap-south-1.compute.amazonaws.com (3.6.37.186)
Initiating Traceroute at 14:37
Completed Traceroute at 14:37, 6.21s elapsed
Initiating Parallel DNS resolution of 9 hosts. at 14:37
Completed Parallel DNS resolution of 9 hosts. at 14:37, 0.61s elapsed
NSE: Script scanning 3.6.37.186.
Initiating NSE at 14:37
Completed NSE at 14:37, 8.80s elapsed
Initiating NSE at 14:37
Completed NSE at 14:37, 1.23s elapsed
Initiating NSE at 14:37
Completed NSE at 14:37, 0.00s elapsed
Nmap scan report for ec2-3-6-37-186.ap-south-1.compute.amazonaws.com (3.6.37.186)
Host is up (0.12s latency).
Not shown: 998 filtered ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 b5:0b:27:ff:49:b9:ca:38:a0:39:a0:55:f7:e6:20:d9 (RSA)
|   256 82:da:44:73:f8:0b:65:da:44:88:03:1f:73:2d:b3:e7 (ECDSA)
|_  256 da:bf:7b:ec:ee:aa:e2:63:c2:89:33:89:a9:af:cf:10 (ED25519)
80/tcp open  http    nginx 1.14.0 (Ubuntu)
| http-cookie-flags:
|   /:
|     PHPSESSID:
|       httponly flag not set
```

Filter Hosts

3.6.37.186 Profile:

nd: nmap -T4 -A -v -Pn 3.6.37.186

Services

ost

c2-3-6-37-186.ap-

Nmap Output Ports / Hosts Topology Host Details Scans

Port	Protocol	State	Service	Version
22	tcp	open	ssh	OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
80	tcp	open	http	nginx 1.14.0 (Ubuntu)

Zenmap

Scan Tools Profile Help

Target: 3.6.37.186 Profile: Scan Cancel

Command: nmap -T4 -A -v -Pn 3.6.37.186

Hosts Services

OS ◂ Host

ec2-3-6-37-186.ap-

Nmap Output Ports / Hosts Topology Host Details Scans

nmap -T4 -A -v -Pn 3.6.37.186 Details

```
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
| http-methods:
|_  Supported Methods: GET HEAD POST
| http-robots.txt: 2 disallowed entries
|_/static/images/ /ovidentiaCMS
|_http-server-header: nginx/1.14.0 (Ubuntu)
|_http-title: Lifestyle Store
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
OS fingerprint not ideal because: Missing a closed TCP port so results incomplete
No OS matches for host
Uptime guess: 20.520 days (since Thu Jun 18 02:09:21 2020)
Network Distance: 28 hops
TCP Sequence Prediction: Difficulty=261 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 80/tcp)
HOP RTT       ADDRESS
1   2.00 ms   192.168.43.1
2   ...
3   66.00 ms  10.72.171.130
4   66.00 ms  172.25.119.184
5   66.00 ms  172.25.119.179
6   67.00 ms  172.16.8.68
7   ... 16
17  116.00 ms 52.95.64.185
18  97.00 ms  52.95.66.215
19  117.00 ms 52.95.66.193
20  ... 27
28  373.00 ms ec2-3-6-37-186.ap-south-1.compute.amazonaws.com (3.6.37.186)

NSE: Script Post-scanning.
Initiating NSE at 14:37
Completed NSE at 14:37, 0.00s elapsed
Initiating NSE at 14:37
Completed NSE at 14:37, 0.00s elapsed
Initiating NSE at 14:37
Completed NSE at 14:37, 0.00s elapsed
Read data files from: C:\Program Files (x86)\Nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 54.06 seconds
           Raw packets sent: 2150 (99.584KB) | Rcvd: 82 (4.290KB)
```

Filter Hosts

Applications Places System Sat Jun 27, 06:33

ashvish183@kali: ~

File Edit View Search Terminal Help

```
root@kali:/home/ashvish183# perl -v

This is perl 5, version 30, subversion 0 (v5.30.0) built for x86_64-linux-gnu-thread-multi
(with 48 registered patches, see perl -V for more detail)

Copyright 1987-2019, Larry Wall

Perl may be copied only under the terms of either the Artistic License or the
GNU General Public License, which may be found in the Perl 5 source kit.

Complete documentation for Perl, including FAQ lists, should be found on
this system using "man perl" or "perldoc perl".  If you have access to the
Internet, point your browser at http://www.perl.org/, the Perl Home Page.

root@kali:/home/ashvish183# perl nikto.pl
Can't open perl script "nikto.pl": No such file or directory
root@kali:/home/ashvish183# nikto -h http://13.235.42.148/
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          13.235.42.148
+ Target Hostname:    13.235.42.148
+ Target Port:        80
+ Start Time:         2020-06-27 06:03:22 (GMT5.5)
---------------------------------------------------------------------------
+ Server: nginx/1.14.0 (Ubuntu)
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Cookie PHPSESSID created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-3268: /static/images/: Directory indexing found.
+ Entry '/static/images/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ Entry '/ovidentiaCMS/' in robots.txt returned a non-forbidden or redirect HTTP code ()
+ "robots.txt" contains 2 entries which should be manually viewed.
+ /phpinfo.php: Output from the phpinfo() function was found.
+ Uncommon header 'x-limonade' found, with contents: Un grand cru qui sait se faire attendre
+ OSVDB-3092: /forum/: This might be interesting...
+ OSVDB-3233: /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
+ /composer.json: PHP Composer configuration file reveals configuration information - https://getcomposer.org/
+ /composer.lock: PHP Composer configuration file reveals configuration information - https://getcomposer.org/
+ 7919 requests: 2 error(s) and 13 item(s) reported on remote host
+ End Time:           2020-06-27 06:32:56 (GMT5.5) (1774 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:/home/ashvish183#
```

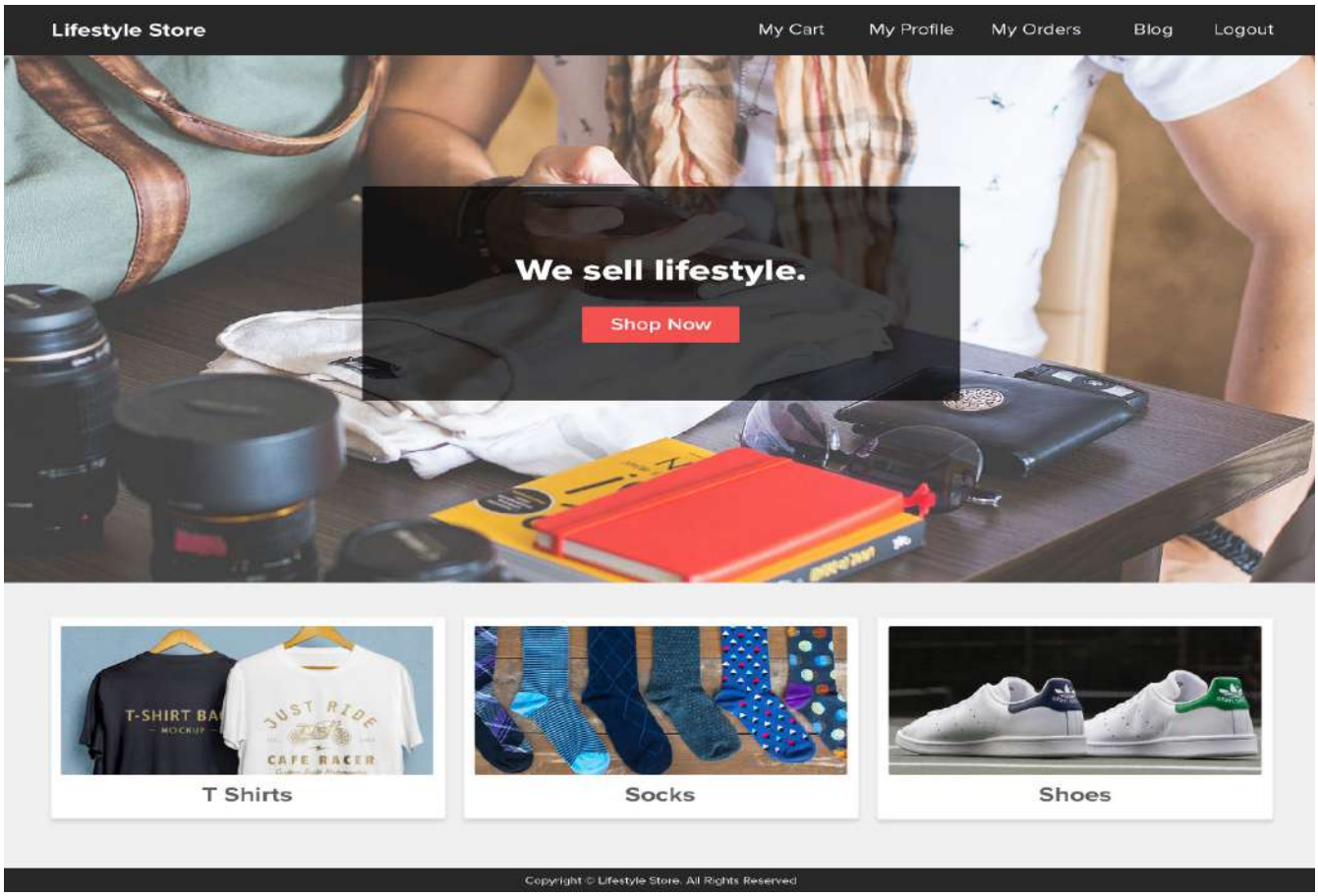
ashvish183@kali: ~ Problem loading page - ... ashvish183@kali: ~

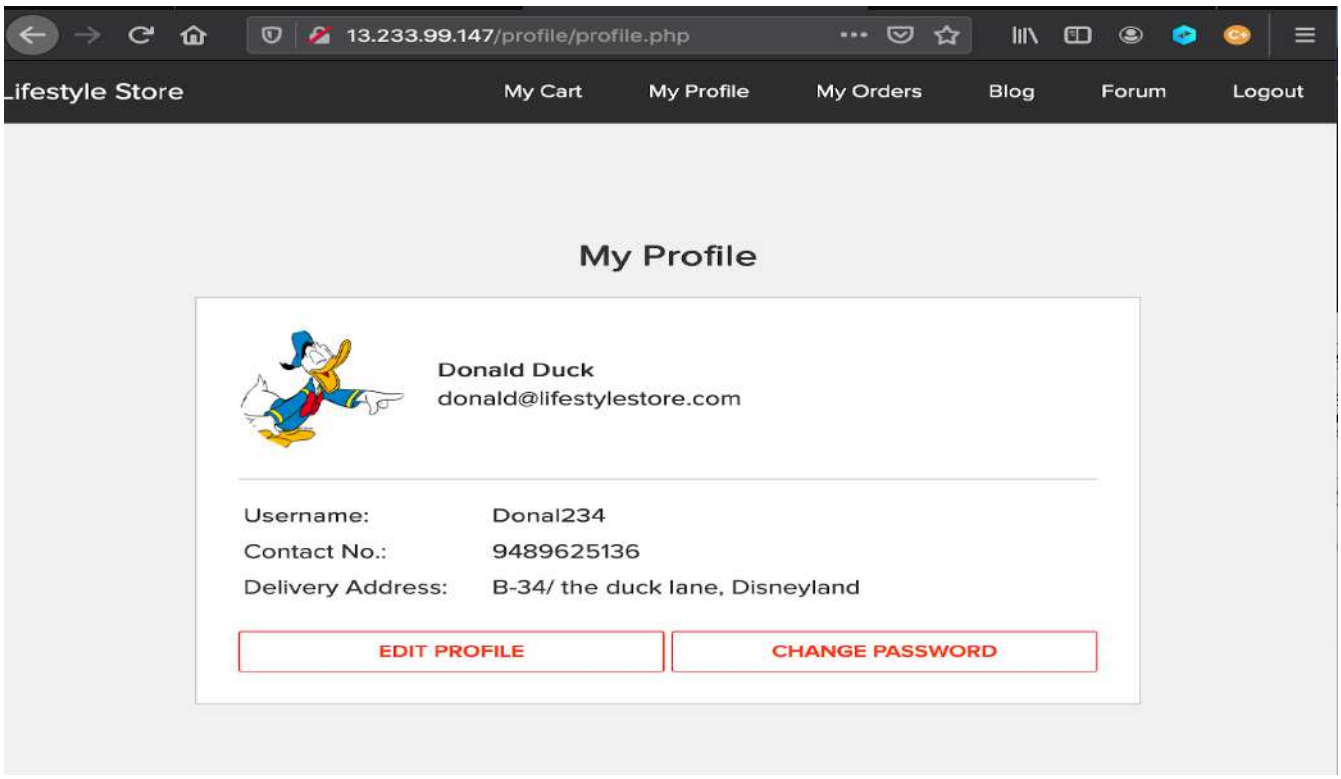
```
DirBuster 1.0-RC1 - Report
http://www.owasp.org/index.php/Category:OWASP_DirBuster_Project
Report produced on Fri May 22 04:55:48 IST 2020
--------------------------------

http://13.232.128.185:80
--------------------------------
Directories found during testing:

Dirs found with a 200 response:

/
/static/images/
/forum/
/static/images/products/
/static/images/icons/
/static/images/uploads/
/static/images/uploads/products/
/forum/admin/
/forum/admin/modules/
/static/images/customers/
/forum/sites/default/themes/
/forum/sites/default/themes/default/
/forum/install/
/static/images/uploads/customers/

Dirs found with a 403 response:

/search/
/products/
/login/
/static/
/static/js/
/static/js/includes/
/static/js/search/
/profile/
/static/js/login/
/static/js/profile/
/common/
/static/js/product/
/signup/
/static/uploads/
/static/js/signup/
/static/uploads/products/
/static/js/admin/
/forum/admin/img/
```

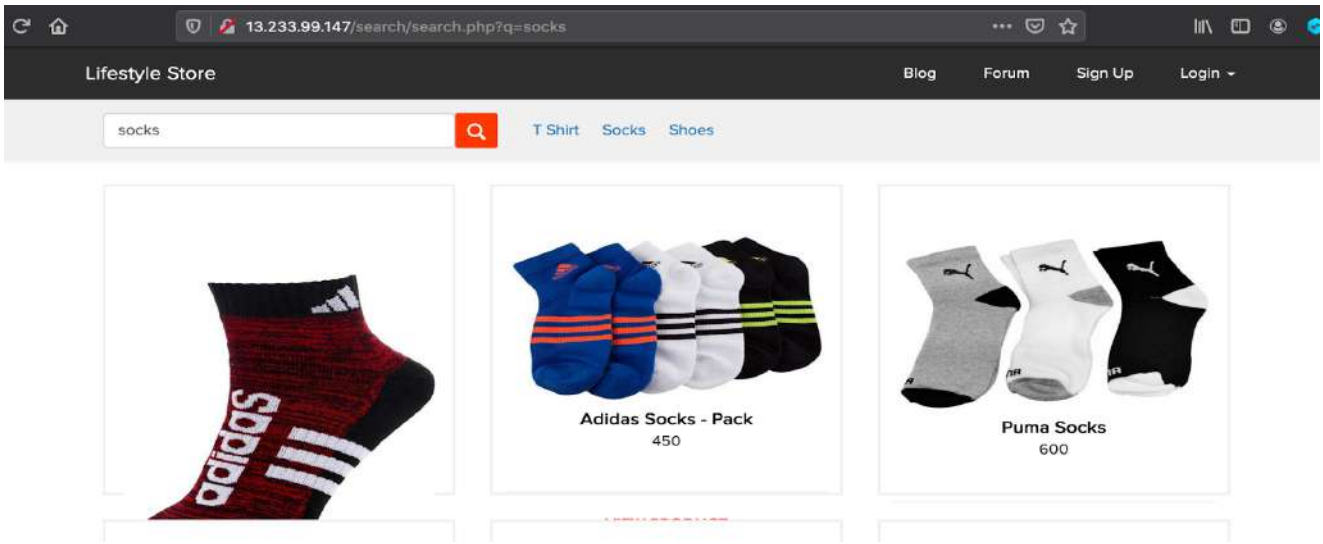
```
/forum/admin/img/
/cart/
/static/js/cart/
/forum/admin/modules/pages/
/forum/sites/
/static/css/
/static/css/search/
/forum/admin/modules/mail/
/forum/sites/default/
/static/css/login/
/static/css/profile/
/static/css/product/
/static/css/signup/
/forum/install/img/
/forum/sites/default/themes/default/img/
/forum/admin/css/
/forum/admin/css/images/
/static/css/admin/


-----------------------------
Files found during testing:

Files found with a 200 responce:

/index.php
/products.php
/static/js/includes/jquery-3.3.1.min.js
/search/search.php
/static/js/includes/bootstrap.min.js
/static/js/includes/nprogress.js
/forum/index.php
/static/js/includes/jquery.validate.js
/static/js/includes/jquery-migrate-3.0.1.min.js
/static/js/app.js
/static/js/includes/jquery-ui.js
/profile/profile.php
/login/submit.php
/profile/submit.php
/login/admin.php
/forum/admin/index.php
/common/header.php
/forum/admin/login.php
/redirect.php
/forum/admin/modules/index.php
/common/footer.php
```

```
/common/footer.php
/forum/admin/modules/login.php
/products/details.php
/forum/admin/modules/users.php
/profile/test.php
/profile/edit.php
/forum/admin/modules/mail/templates.php
/forum/admin/modules/categories.php
/cart/cart.php
/forum/admin/modules/pages/pages.php
/cart/add.php
/forum/install/index.php
/forum/sites/default/themes/default/info.php


-----------------------------
```

Required screenshots

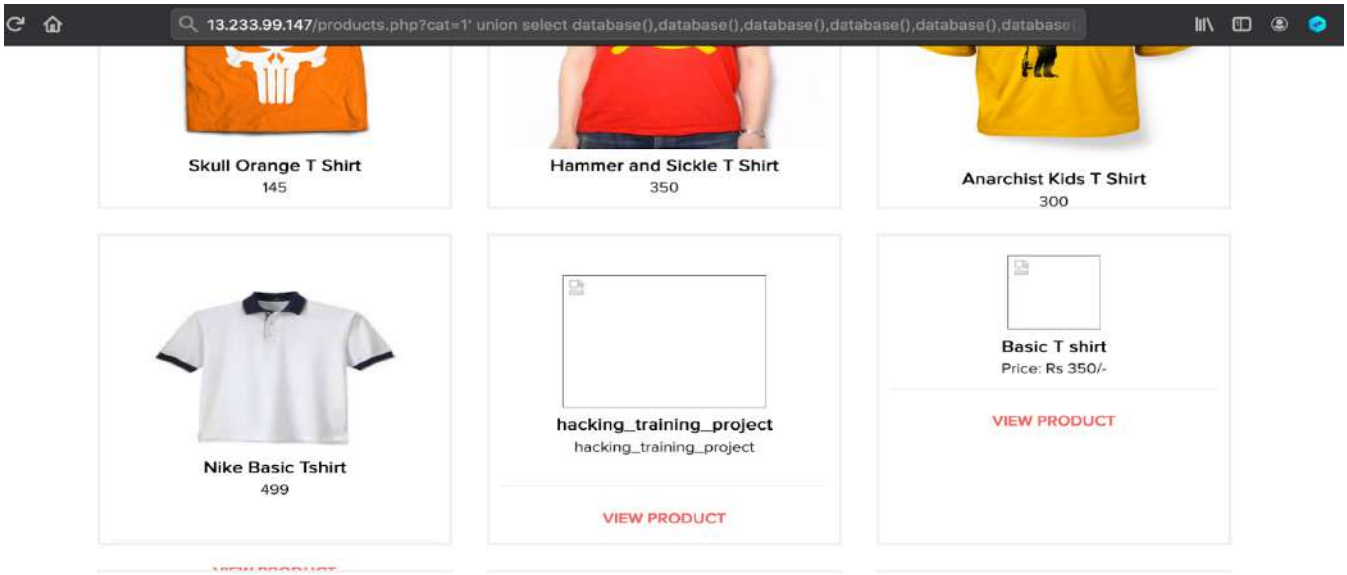


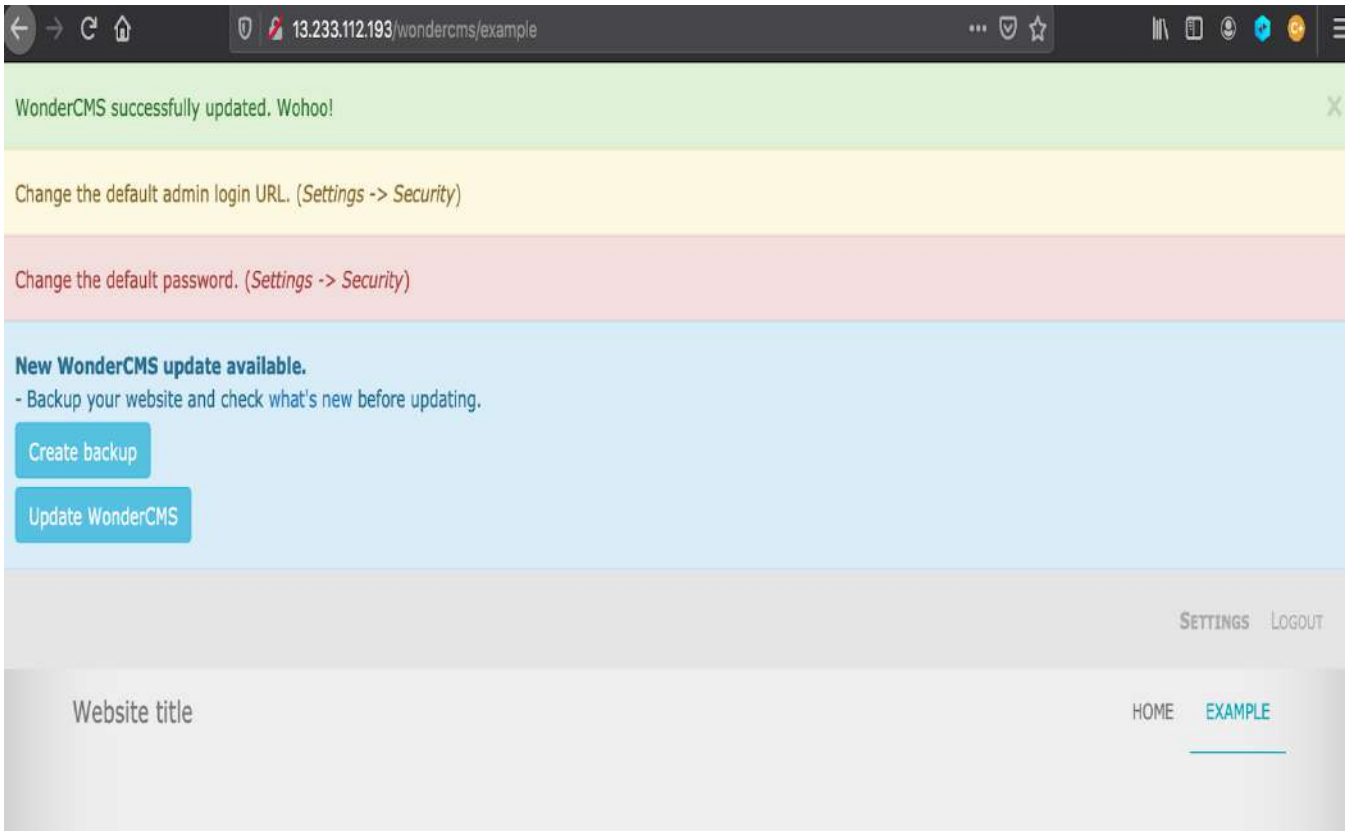


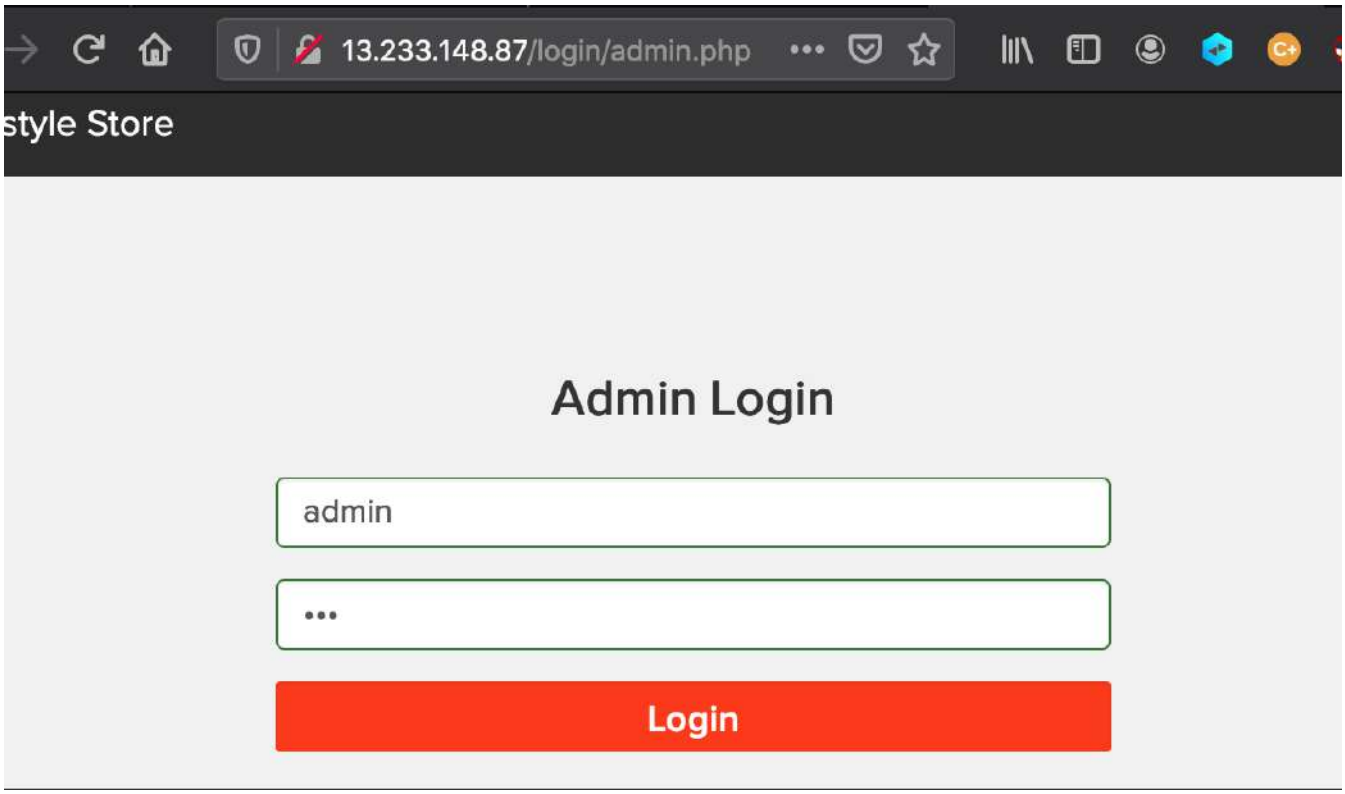


Index of /static/images/

../		
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




13.233.112.193/wondercms/example
WonderCMS successfully updated. Wohoo!
Change the default admin login URL. (Settings -> Security)
Change the default password. (Settings -> Security)
New WonderCMS update available.
- Backup your website and check what's new before updating.
Create backup
Update WonderCMS
SETTINGS
LOGOUT
Website title
HOME
EXAMPLE


13.233.148.87/login/admin.php
style Store
Admin Login
admin
•••
Login

DEVELOPER'S REPORT

E-COMMERCE WEBSITE

LIFESTYLE STORE

DETAILED PROJECT REPORT

SECURITY STATUS – EXTREMELY VULNERABLE

- **Hackers can steal all the records of Lifestyle store(SQLi)**
- **Hacker can take control of complete server including View, Add, Edit, Delete files and folders.(shell upload and weak passwords)**
- **Hacker can change source code of application to host malware, phishing pages or even explicit content.(Shell upload)**
- **Hacker can see details of any customer.(IDOR)**
- **Hacker can easily access or bypass admin account authentication.(bruteforcing)**
- **Hacker can get access to seller details and login into the website using customer of the month usernames (PII).**
- **Hacker can change the password , confirm order and remove item of customer(CSRF)**

VULNERABILITY STATISTIC

vulnerabilities:

S.NO.	SEVERITY	VULNERABILITY	COUNT
1	CRITICAL	SQL injection	3
2	CRITICAL	Access to admin panel	1
3	CRITICAL	Arbitrary file upload	2
4	CRITICAL	Account takeover by OTP bypass	1
5	CRITICAL	CSRF	3
6	SEVERE	Reflected cross site scripting	1
7	SEVERE	Stored cross site scripting	1
8	SEVERE	Common password	1
9	SEVERE	Component with known vulnerability	3
10	MODERATE	Server misconfiguration	1
11	MODERATE	Unauthorized access to user details (IDOR)	4
12	MODERATE	Directory listings	5
13	LOW	Personal Information leakage	2
14	LOW	Client side and server side validation bypass	1
15	LOW	Default error display	1
16	LOW	Open redirection	2

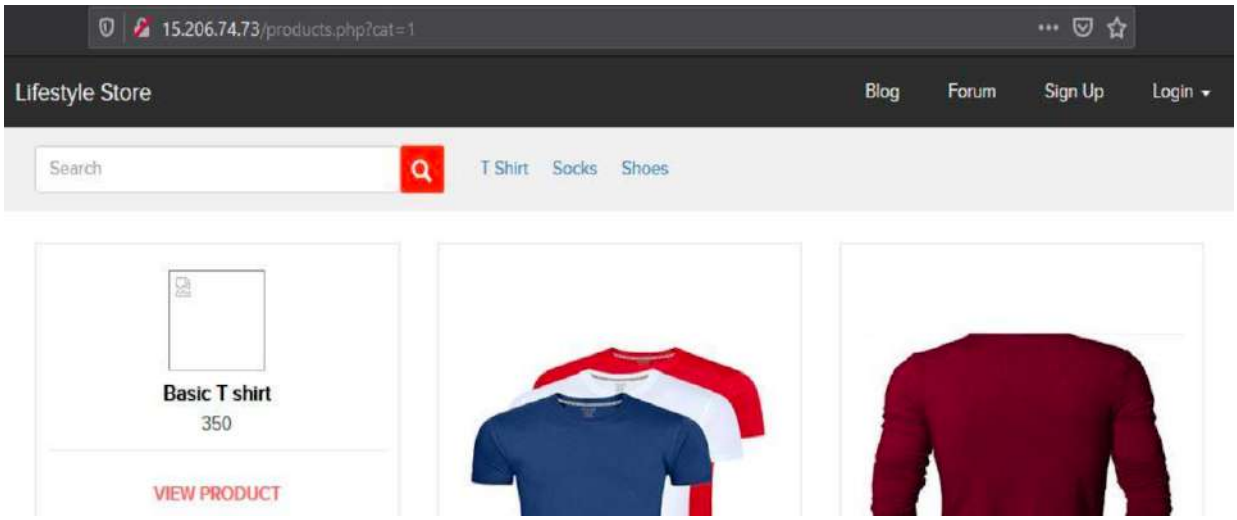
1. Sql injections

SQL Injection (Critical)	Below mentioned URL in the **T-shirt/socks/shoes** module is vulnerable to SQL injection attack Affected URL : •http://15.206.74.73/products.php?cat=1 Affected Parameters : •cat (GET parameter) Payload: •cat = 1' Affected URL : •http://15.206.74.73/products.php?q=socks Affected Parameters : •q (GET parameter) Payload: •q=socks'

SQL Injection (Critical)	Here are other similar SQLi in the application **Affected URL :** • http://15.206.74.73/products.php?cat=2 • http://15.206.74.73/products.php?cat=3

Observation

Navigate to the T-Shirt tab where you will see a number of T-shirts. Notice the GET parameter CAT in the URL:



We apply a single quote in cat parameter: **products.php?cat=1' and we get complete MySQL error:**

• We then put --+ : **products.php?cat=1'--+ and we error is removed confirming SQL injection.**

• Now hacker can inject sql or use sqlmap to get access to the database

Proof of Concept (PoC):- Attacker can dump arbitrary data

- **No of databases: 2**
  - information_schema
  - hacking_training_project
- **No of tables : 10**
  - brands
  - cart_items
  - categories
  - customers
  - order_items
  - orders
  - product_reviews
  - products
  - sellers
  - user

Database: hacking_training_project
Table: users
[15 entries]

user_name	password	phone_number	unique_key
admin	$2y$10$xkmdvrxSCxqdyWSrDx5YSelNAwX.7pQ2nQmaTCovH4CFssxgyJTki	8521479630	1546892795 5c66694cba1174.29688447
Donal234	$2y$10$PM.7nBSP5FMaldXiM/S3s./p5xR6GTKvjry7ysJtxOkBqOJURAHsO	9489625136	778522555c6669996f5a24.34991684
Pluto98	$2y$10$xkmdvrxSCxqdyWSrDx5YSelNAwX.7pQ2nQmaTCovH4CFssxgyJTki	8912345670	19486318945c666a037b1432.99985767
chandan	$2y$10$4cZBEIrgthXdvTlhwUlivuFELe03rR.GIcdp03NjrlS0VeiOKLVDa	7854126395	12404594545c666a3b49e0f8.08173871
Popeye786	$2y$10$FkvlRfwYTioWOw2CaZtAQuXvnhGAUjt/If/yTqkNPC5zTrsVm7EeC	9745612300	18430379145c666a53af8431.79566371
Radhika	$2y$10$RYxNhOyV/G4g7OtFwpqYaexvHi8rF6XXui8kTlWtrfghTutCA8JC.	9512300052	15611262655c666b312f73e0.70827297
Nandan	$2y$10$G.cRNLMEiG79ZFXElHg.R.o95334UOxmZu4.9MqzR5614ucwnk59K	7845129630	1587354115c666b65bb44a5.36505317
MurthyAdapa	$2y$10$mzQGzD4sDSj2EunpCioe4eK18clAbs0T2Pla1P6eVlDPR.11UubDG	8365738264	16357203785c68f640c699a2.83646347
john	$2y$10$GhDB8hlX6XjPMY12GZlvDO7Y3en97ul/.oXTZLmYqB6F18FBgecvG	6598325015	9946437385c6a435f76bef0.14675944
bob	$2y$10$kiUikn3HPFbuyTtK75lLNurxzqC0LX3eMGy0/Uxl6JOoG37dCGKLq	8576308560	4305822125c6a43ec507df0.68309267
jack	$2y$10$z/nyNlkRJ76m9ItMZ4N5loeRxy6Gkqi9N/UBcJu5ZeO7eM7N4pTHu	9848478231	15257114565c6a444692b707.17903432
bulla	$2y$10$HT5oiRMetqaZ7xGZPE9s2.Mk1yF4PnYDJHCWbm2w/xuKpjEEI/zjG	7645835473	18292501185c6a4493a5ddb0.87138000
hunter	$2y$10$pB3U9iFxwBgSbl2AkBpiEeIBdhiYfWy9y.xV23ql2gGbMCyn7N3q2	9788777777	13824560345c80704e821145.26019698
asd	$2y$10$At5pFZnRWpjCD/yNnJWDL.L3Cc4CvOW8Q/WEHmWzBFqVIkBQFpCF2	9876543210	8057400125c862a7f5916c9.06111587
acdc	$2y$10$J50B78.gpucuLTwpHwbcPedYcain.Yi.tsTLyQtK17FzdSpmIRRbi	9999999999	13104802695c86f43f0c3705.77019309

Business Impact – Extremely High

Using this vulnerability, an attacker can execute arbitrary SQL commands on a Lifestyle store server and gain complete access to internal databases along with all customer data inside it.
Previous slide has the screenshot of users table which shows user credentials being leaked that too in plain text without any hashing/encryption.
Attackers can use this information to login to admin panels and gain complete admin level access to the website which could lead to complete compromise of the server and all other servers connected to it.

RECOMMENDATIONS

1. **Use whitelists, not blacklists**
2. **Don't trust any user input**
3. **Adopt the latest technologies**
4. **Ensure Errors are Not User-Facing**
5. **Disable/remove default accounts, passwords and databases**

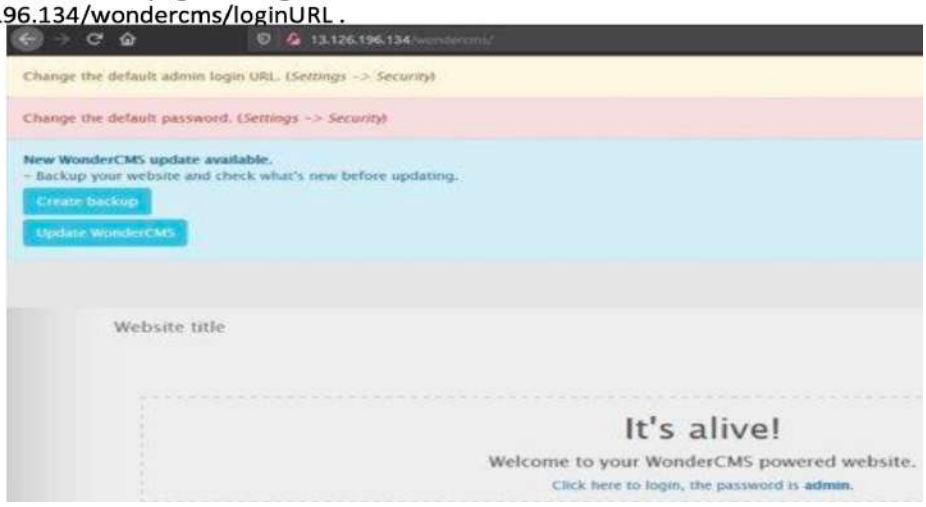
References: https://www.owasp.org/index.php/SQL_Injection and https://en.wikipedia.org/wiki/SQL_injection

2. Access to admin panel

Access to admin panel (Critical)	Below mentioned URL is vulnerable to **Arbitrary File Upload and making other admin level** changes. Affected URL : •http://13.126.196.134/wondercms/loginURL

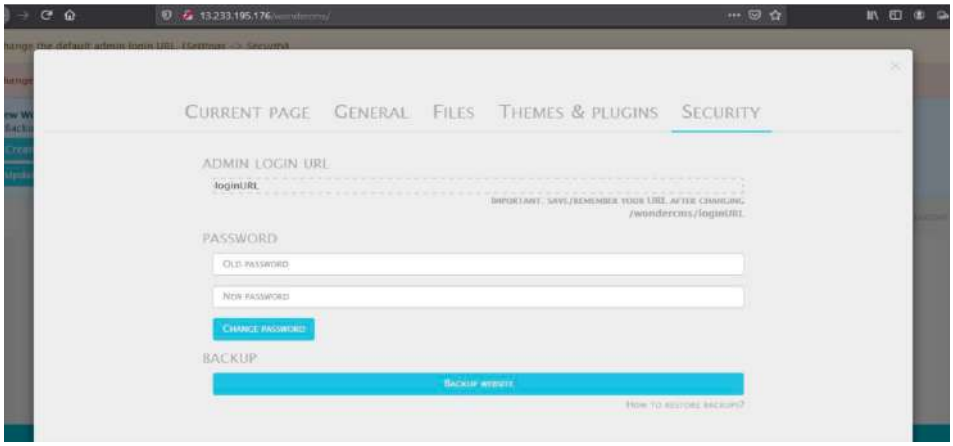
Observation

- When we navigate to http://13.126.196.134/wondercms/ url
- we get the password on the page and login as : admin in the url http://13.126.196.134/wondercms/loginURL .



Proof of Concept (PoC)

Hackers can change the admin password . Hackers can also add and delete pages. Hackers can upload any malicious file.



Business impact - Extremely High

• Hacker can do anything with the page, he will have full access to the page and can govern the page according to it's will.
• It is a massive business risk.
• Loss can be very high.

RECOMMENDATIONS

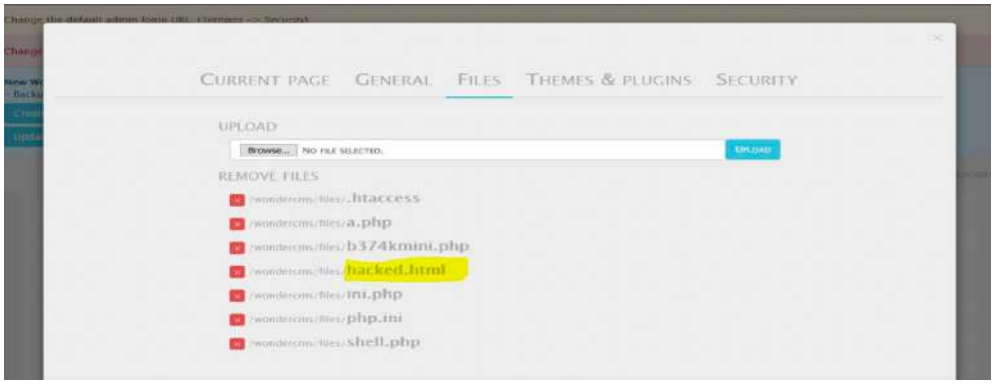
1. The default password should be changed and a strong password must be set up.
2. The admin url must also be such that it's not accessible to normal users.
3. Password changing option must be done with 2 to 3 step verification.

References: https://www.owasp.org/index.php/Default_Passwords and https://www.us-cert.gov/ncas/alerts/TA13-175A

3. Arbitrary file upload

Arbitrary file upload (Critical)	The attacker can upload insecure shells and files and gain access over the entire database and login as the admin and the vesion is known to have vulnerabilities . Affected URL : •http://13.126.196.134/wondercms/Affected Parameters : •File Upload (POST parameter) The attacker can upload files with extension other than .jpeg . Affected URL : •http://13.126.196.134/profile/2/edit/ Affected Parameters : •Upload Profile Photo (POST parameter)

Observations:



Proof of concept

•Weak password - admin.
•Arbitrary File Inclusion.

Business Impact – Extremely High

A malicious user can access the Dashboard which discloses many critical information of an organisation including: Important files, Passwords, and much more...
Any backdoor file or shell can be uploaded to get access to the uploaded file on a remote server and data can be exfiltrated. The presence of an actual malicious file can compromise the entire system leading to system takeover/ data stealing.

Recommendation

•Change the Admin password to something strong and not guessable.
•The application code should be configured in such a way that it should block uploading of malicious files extensions such as exe/ php and other extensions with a thorough server as well as client validation. CVE ID allocated:CVE-2017-14521.

References: https://www.owasp.org/index.php/Unrestricted_File_Upload and https://www.opswat.com/blog/file-upload-protection-best-practices

Recommendation

Take the following precautions:

1. Use a strong password 8 character or more in length with alphanumerics and symbols.
2. It should not contain personal/guessable information.
3. Do not reuse passwords.
4. Disable default accounts and users.
5. Change All Passwords To Strong Unique Passwords.

References:

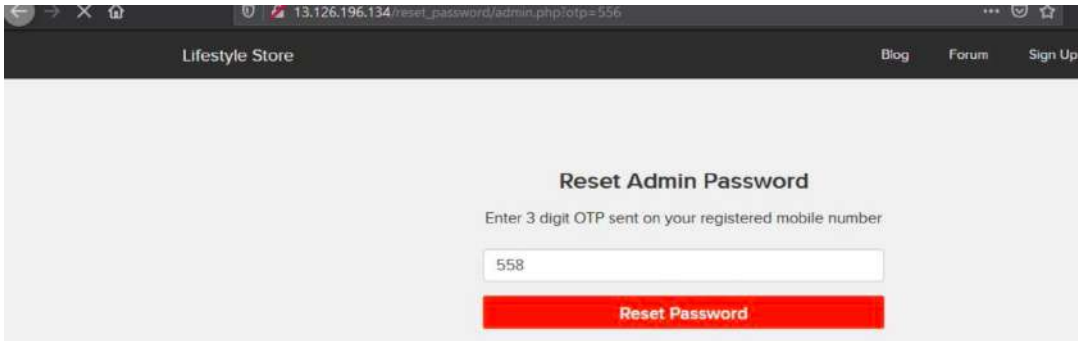
https://www.owasp.org/index.php/Testing_for_weak_password_change_or_reset_functionalities_(OTG-AUTHN-009) and *https://www.owasp.org/index.php/Default_Passwords* and *https://www.us-cert.gov/ncas/alerts/TA13-175A*

4. Account takeover using OTP bypass

Account Takeover Using OTP Bypass (Critical)	The below mentioned login page allows login via OTP which can be bruteforced **Affected URL :** • http://13.126.196.134/reset_password/admin.php?otp= **Affected Parameters :** • OTP (POST parameters)

Observation

• Navigate to http://13.126.196.134/reset_password/admin.php?otp= and You will see the user login page via OTP.



• Following request will be generated containing OTP parameters.

• Now We're Brute Forcing It.

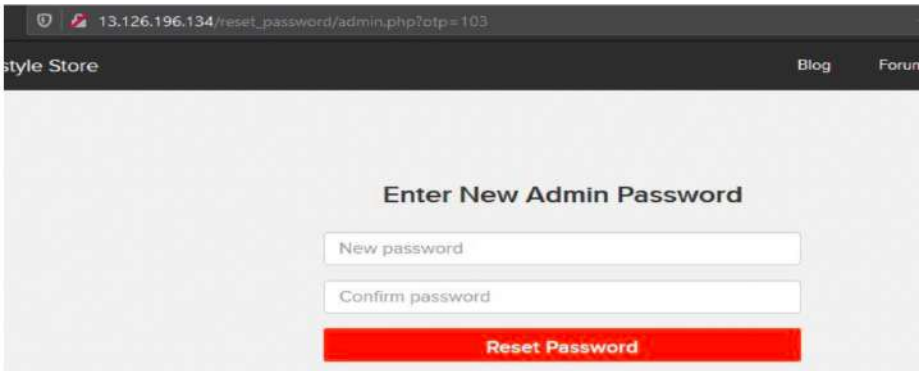
• And we easily got the valid otp.

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
4	103	200	☐	☐	4476	
0		200	☐	☐	4380	
1	100	200	☐	☐	4380	
2	101	200	☐	☐	4380	
3	102	200	☐	☐	4380	
5	104	200	☐	☐	4380	
6	105	200	☐	☐	4380	
7	106	200	☐	☐	4380	
8	107	200	☐	☐	4380	
9	108	200	☐	☐	4380	
10	109	200	☐	☐	4380	
11	110	200	☐	☐	4380	
12	111	200	☐	☐	4380	

POC

• Now a hacker can change the password of the admin dashboard.



Business Impact – Extremely High

A malicious hacker can gain complete access to any account just by brute forcing the otp. This leads to complete compromise of personal user data of every customer.
Attackers once logged in can then carry out actions on behalf of the victim which could lead to serious financial loss to him/her.

Recommendation

Take the following precautions:
• Use proper rate-limiting checks on the no of OTP checking and Generation requests.
• Implement anti-bot measures such as ReCAPTCHA after multiple incorrect attempts.
• OTP should expire after a certain amount of time like 2 minutes.
• OTP should be at least 6 digit and alphanumeric for more security.

References:

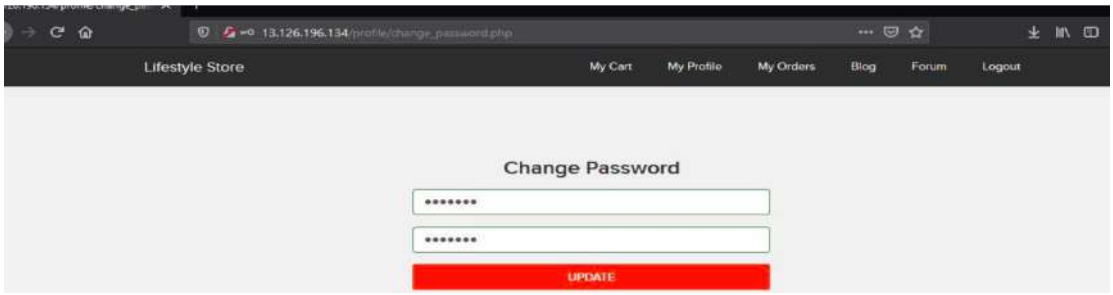
https://www.owasp.org/index.php/Testing_Multiple_Factors_Authentication_(OWASP-AT-009) and *https://www.owasp.org/index.php/Blocking_Brute_Force_Attacks*

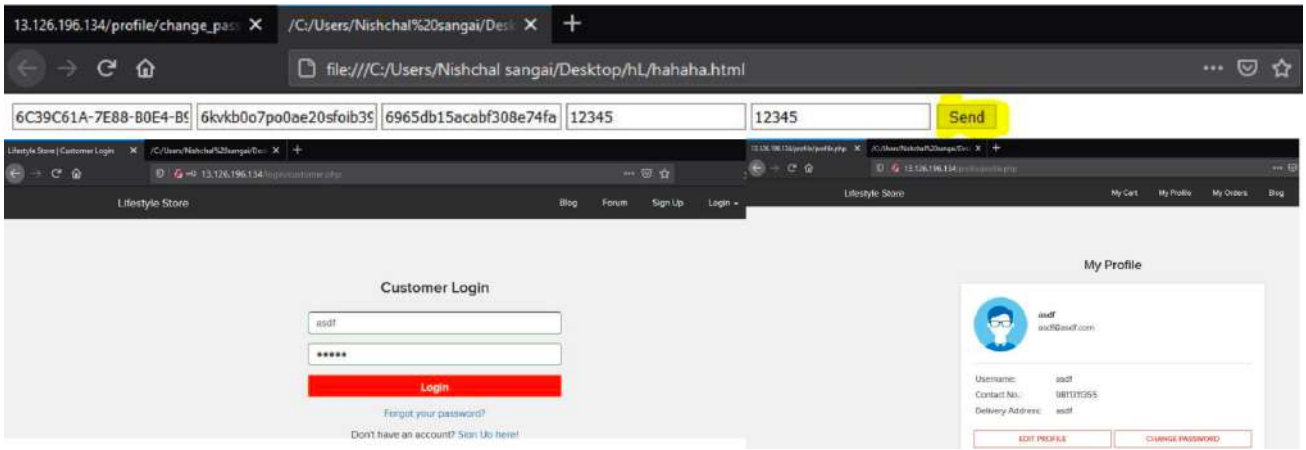
5. CSRF

Observation:

• Here you can see a 7 digit password ,but due to csrf I'll change the password at the moment he wants to update.
• Here's the file I opened while changing password , when we click on send the password will change to 12345.

Unauthorised Access to Customer Details (Critical)	The below mentioned login page allows you to change password without verification and view details of other customers (CSRF). Affected URL : •http://13.126.196.134/profile/change_password.php Affected Parameters : •Update button (POST parameter) We can change the password. Affected URL : •http://13.126.196.134/cart/cart.php Affected Parameters : •Remove option (POST parameter) Affected URL : •http://13.126.196.134/cart/cart.php Affected Parameters : •Confirm order option (POST parameter)



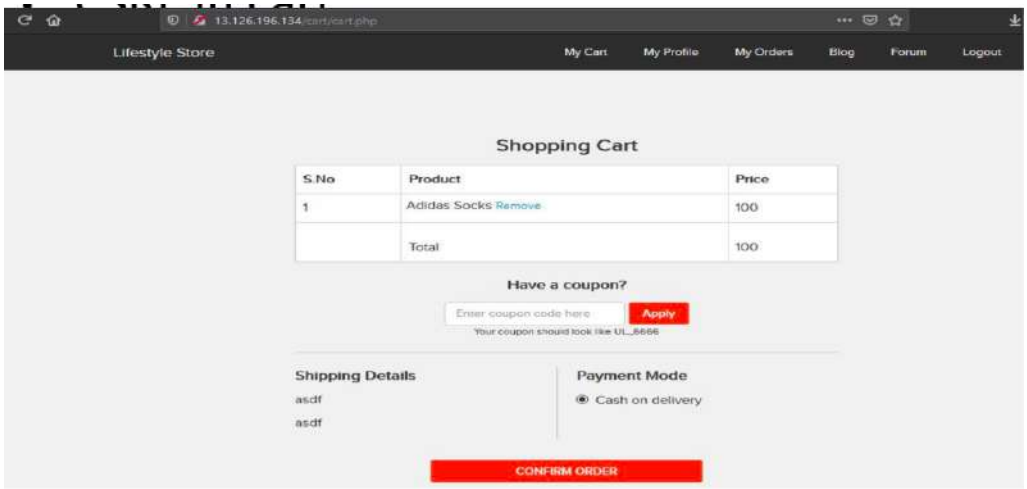


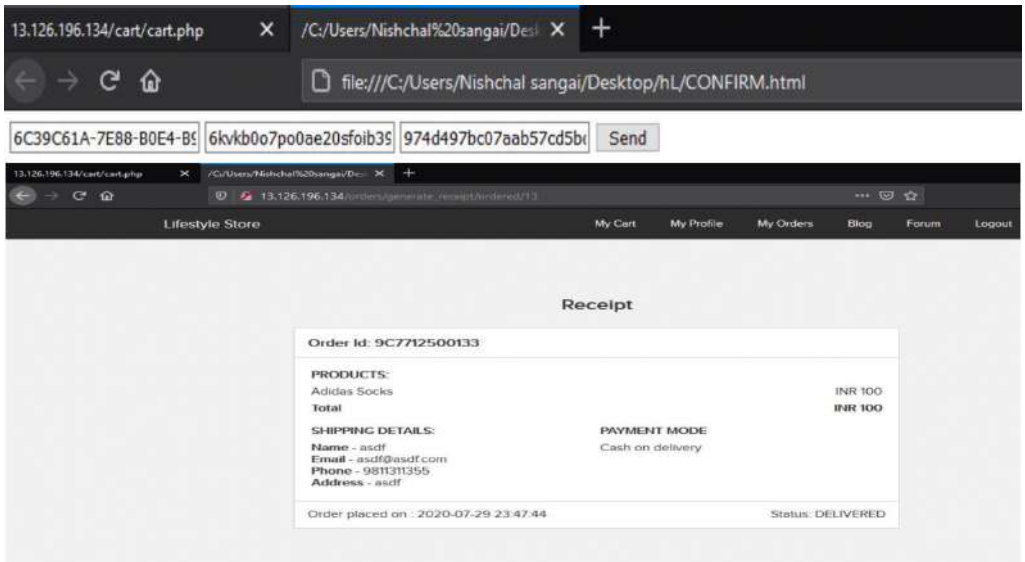
POC: Here's the code generated by burp suite community edition.

```html
<!DOCTYPE html>
<html>
  <!-- CSRF PoC - generated by Burp Suite i0 SecLab plugin -->
<body>
    <form method="POST" action="http://13.126.196.134:80/profile/change_password_submit.php">
        <input type="text" name="key" value="6C39C61A-7E88-B0E4-B9D5-FC7EBB773CB1">
        <input type="text" name="PHPSESSID" value="6kvkb0o7po0ae20sfoib398mn4">
        <input type="text" name="X-XSRF-TOKEN" value="
        6965db15acabf308e74fa61bde40c623856201cbfe80ff1f28178fa5f13b28f3">
        <input type="text" name="password" value="12345">
        <input type="text" name="password_confirm" value="12345">
        <input type="submit" value="Send">
    </form>
</body>
</html>
```

Observation: CSRF in cart

Here you can see, the order is placed unwantedly by the user through CSRF.





POC: Here's the code generated by burp suite community edition.

```html
<!DOCTYPE html>
<html>
  <!-- CSRF PoC - generated by Burp Suite i0 SecLab plugin -->
<body>
    <form method="POST" action="http://13.126.196.134:80/orders/confirm.php">
        <input type="text" name="key" value="6C39C61A-7E88-B0E4-B9D5-FC7EBB773CB1">
        <input type="text" name="PHPSESSID" value="6kvkb0o7po0ae20sfoib398mn4">
        <input type="text" name="X-XSRF-TOKEN" value="
        974d497bc07aab57cd5bdcfa5ebbdcbe91798fbbb03b1f0d7f9a04ff6e4f44e6">
        <input type="submit" value="Send">
    </form>
</body>
</html>
```

Business Impact – Very High

1. Hackers can change the password of any user .
2. Hackers can make users do unwanted things.
3. It makes a very bad impact on the website in front of the user.
4. Hackers can remove and confirm orders in the cart of the user.

Recommendations: Take the following precautions:

•Implement an Anti-CSRF Token.

•Do not show the customers of the month on the login page.
•Use the Same Site Flag in Cookies.
•Check the source of the request made.
•Take some extra keys or tokens from the user before processing an important request.
•Use 2 factor confirmations like otp , etc. for critical requests.

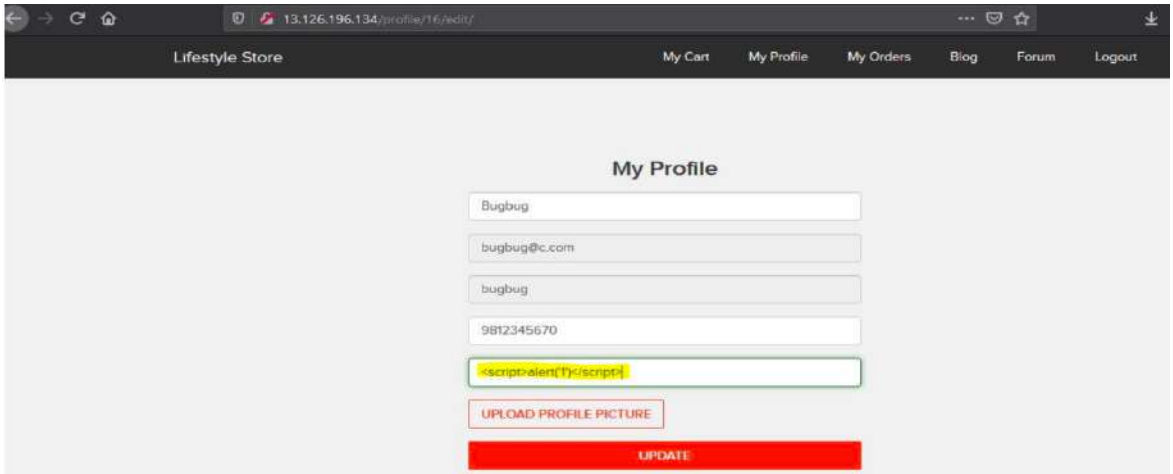
References: https://www.netsparker.com/blog/web-security/csrf-cross-site-request-forgery/ and https://digitalguardian.com/blog/how-secure-personally-identifiable-information-against-loss-or-compromise

6. Reflected Cross Site Scripting (XSS)

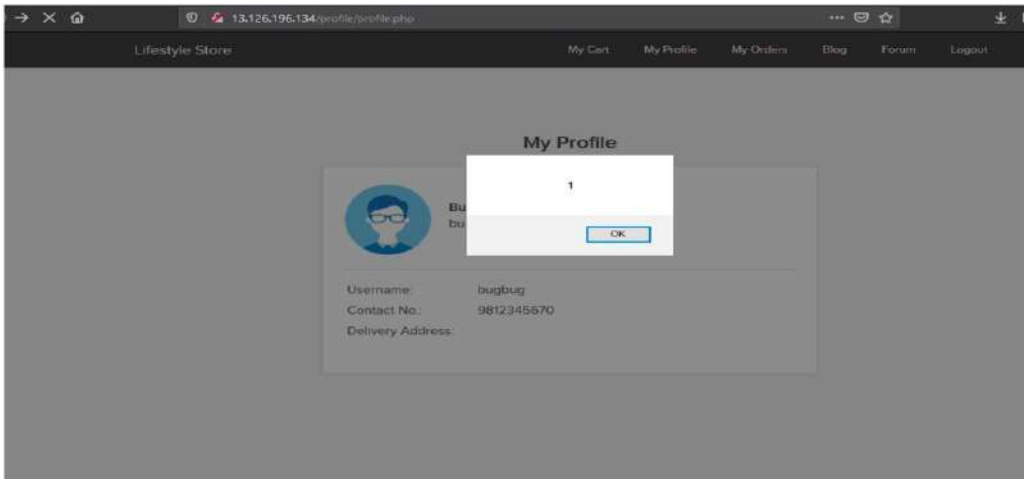
Reflected Cross Site Scripting (Severe)	Below mentioned parameters are vulnerable to reflected XSS **Affected URL :** • http://13.126.196.134/profile/16/edit/ **Affected Parameters :** • address(POST parameters) **Payload:** • <script>alert(1)</script>

Observation

Open edit profile through URL and write a script on the address bar.



POC



Business impact - High

As an attacker can inject arbitrary HTML CSS and JS via the URL, the 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 organisation.

All an attacker needs to do is send the link with the payload to the victim and the victim would see hacker controlled content on the website. As the user trusts the website, he/she will trust the content.

Recommendation: Take the following precautions:

• 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.

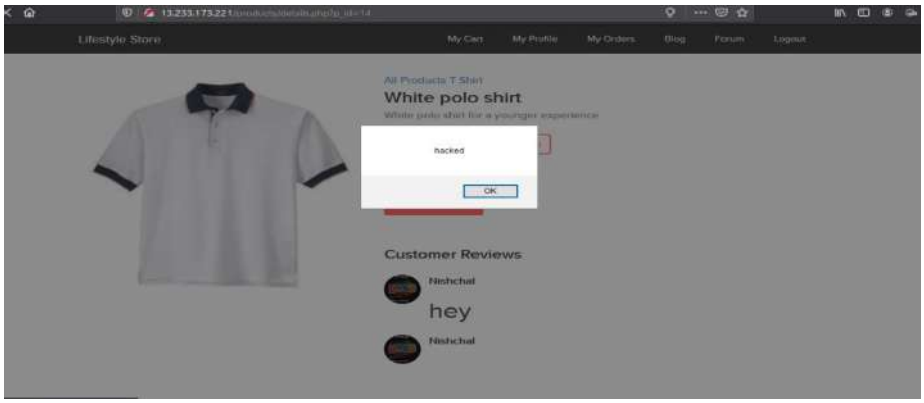
References: *https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)* *and* https://en.wikipedia.org/wiki/Cross-site_scripting , https://www.w3schools.com/html/html_entities.asp

7. Stored Cross Site Scripting (XSS)

Stored Cross Site Scripting (Severe)	Below mentioned parameters are vulnerable to reflected XSS **Affected URL :** • http://13.126.196.134/products/details.php?p_id=14 **Affected Parameters :** • POST button under Customer Review (POST parameters) **Payloads:** • <script>alert('Hacked')</script> • hey

Observations: Now try entering the payload in the review box.

Hit the post button , you can see stored XSS or permanent XSS.



Business impact - High

As an attacker can inject arbitrary HTML CSS and JS via the URL, the 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 organisation.

All an attacker needs to do is send the link with the payload to the victim and the victim would see hacker controlled content on the website. As the user trusts the website, he/she will trust the content.

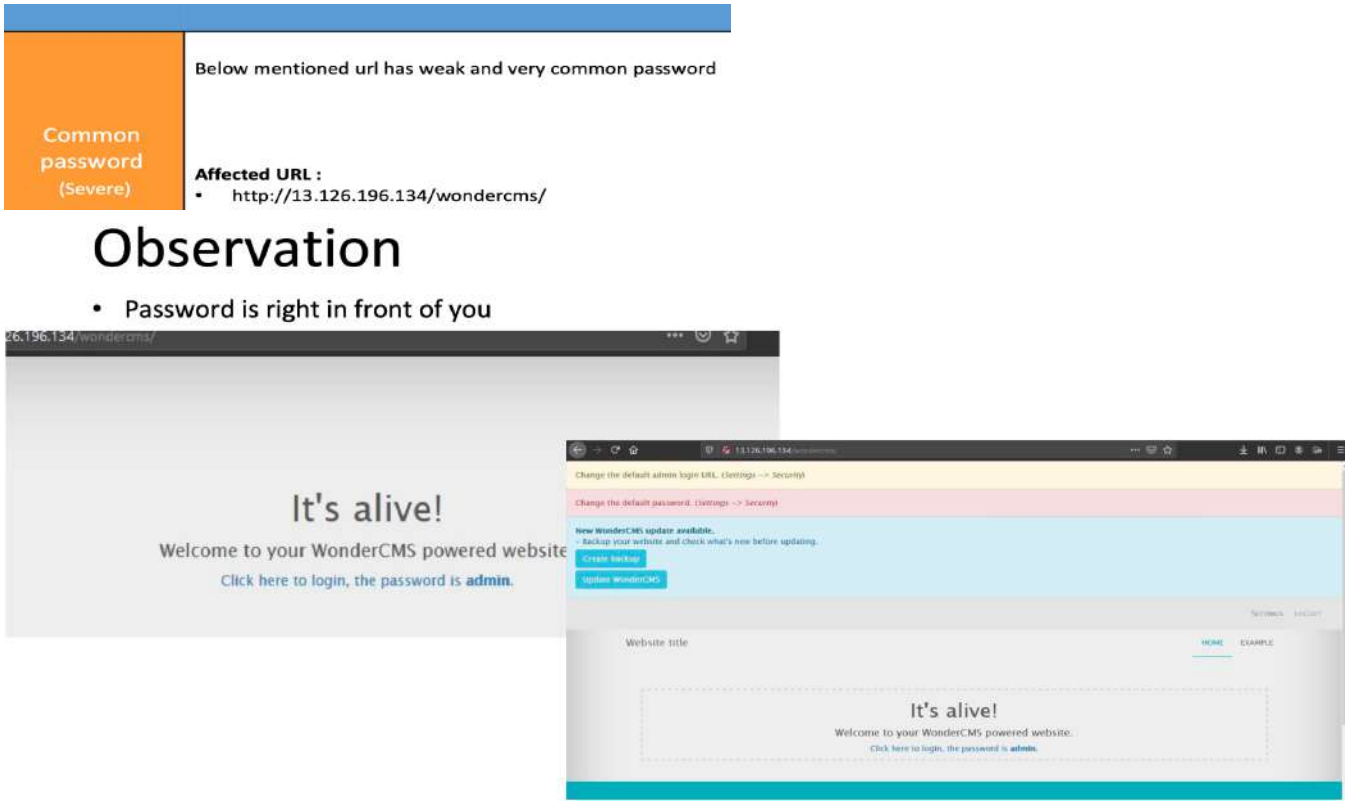
Recommendation: Take the following precautions:

• Sanitize 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.

References: *https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)* *and* https://en.wikipedia.org/wiki/Cross-site_scripting , https://www.w3schools.com/html/html_entities.asp

8. COMMON PASSWORD



Business Impact – high

Easy, default and common passwords make it easy for attackers to gain access to their accounts, illegal use of them and can harm the website to any extent after getting logged into privileged accounts.

Recommendation

•There should be password strength check at every creation of an account.

•There must be a minimum of 8 characters long password with a mixture of numbers , alphanumerics, special characters ,etc.

•There should be no repetition of password ,neither on change nor reset.

•The password should not be stored on the web, rather should be hashed and stored.

References: https://www.acunetix.com/blog/articles/weak-password-vulnerability-common-think/ and https://www.owasp.org/index.php/Testing_for_Weak_password_policy_(OTG-AUTHN-007)

9. Component with known vulnerability

Component with known vulnerability (Severe)	•Server used is nginx/1.14.0 appears to be outdated (current is at least 1.17.3) i.e it is known to have exploitable vulnerabilities. •WonderCMS •Codoforum (Powered by codologic)

Observation

Codologic Vulnerability:- Now you can see that they have blind sql injection vulnerability

POC

Codologic Vulnerability,
It has multiple sql injection vulnerability,
Chack the link of exploit-db in reference.

```
Proof of Concept:


        http://localhost/codoforum/index.php?u=/page/6 and
1=1%23/terms-of-service
        -> true (terms and services displayed)
        http://localhost/codoforum/index.php?u=/page/6 and
1=2%23/terms-of-service
        -> false ("You do not have enough permissions to view this page!")

Code:

        routes.php:593

            $pid = (int) $id;
            $user = \CODOF\User\User::get();

            $qry = 'SELECT title, content FROM ' . PREFIX . 'codo_pages p '
                    . ' LEFT JOIN ' . PREFIX . 'codo_page_roles r ON
r.pid=p.id '
                    . ' WHERE (r.rid IS NULL OR  (r.rid IS NOT NULL AND
r.rid IN (' . implode($user->rids) . '')))'
                    . ' AND p.id=' . $id;
```

Business Impact – high

Exploits of every vulnerability detected are regularly made public and hence outdated software can very easily be taken advantage of.If the attacker comes to know about this vulnerability ,he may directly use the exploit to take down the entire system, which is a big risk.

Recommendations:

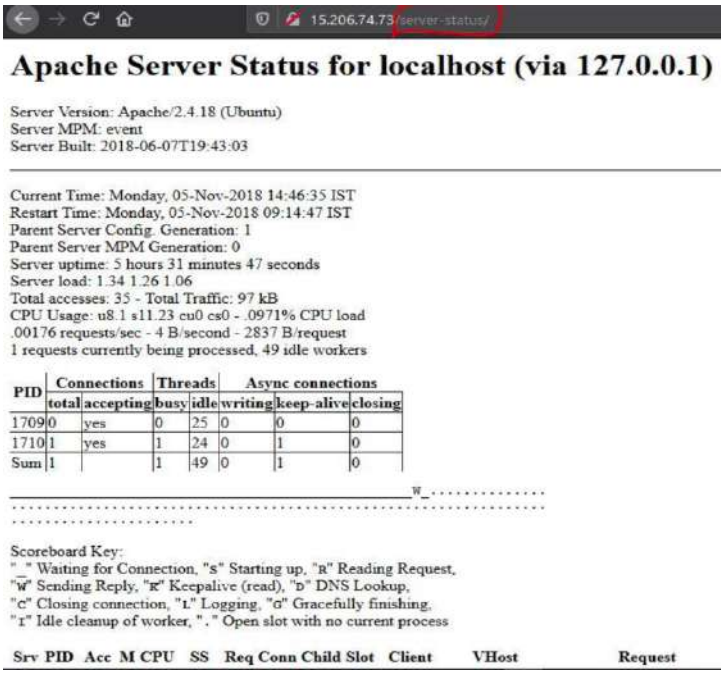
•Upgrade to the latest version of Affected Software/theme/plugin/OS which means latest version.
•If upgrade is not possible for the time being, isolate the server from any other critical data and servers.

References: https://usn.ubuntu.com/4099-1/ (for ubuntu) and

https://www.exploit-db.com/exploits/37820 and
https://securitywarrior9.blogspot.com/2018/01/vulnerability-in-wonder-cms-leading-to.html

10. Server misconfiguration

Server misconfiguration (Moderate)	Below mentioned url will show you the server related info **URL** http://13.126.196.134/server-status http://13.126.196.134/server-info



Observations and POC:

Recommendation:

1. Keep the software up to date.
2. Disable all the default accounts and change passwords regularly.
3. Develop strong app architecture and encrypt data which has sensitive information.
4. Make sure that the security settings in the framework and libraries are set to secured values.
5. Perform regular audits and run tools to identify the holes in the system.

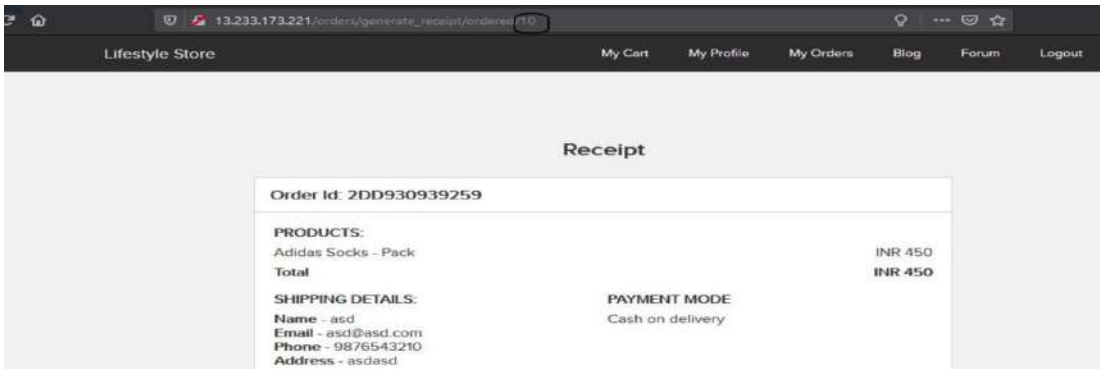
References: https://www.ifourtechnolab.com/blog/owasp-vulnerability-security-misconfiguration

11. Unauthorised access to user details(IDOR)

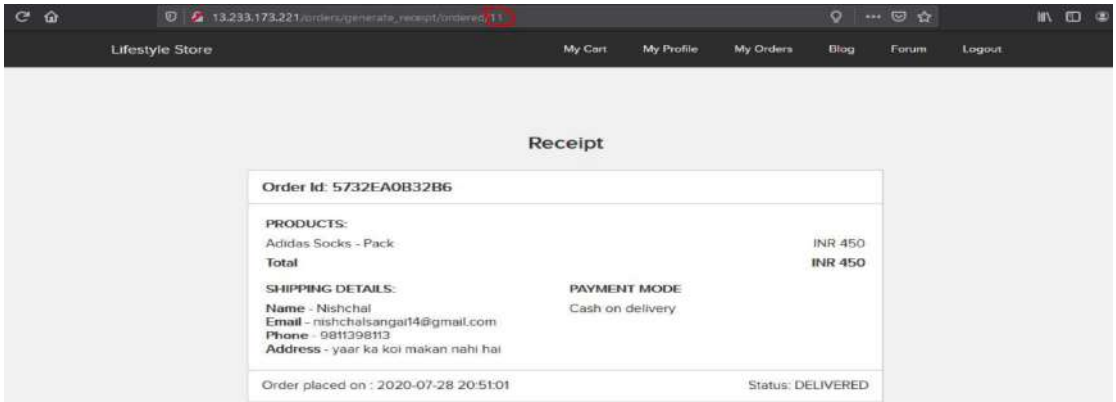
Unauthorized access to user details (Moderate)	Below mentioned url will have vulnerabilty through which anyone can see the details of another user **URL** http://13.233.173.221/generate_receipt/ordered/10 Affected parameter Ordered/**10** **Payload** http://13.233.173.221/generate_receipt/ordered/11

Unauthorized access to user details (Moderate)	Below mentioned url will have vulnerabilty through which anyone can see the details of another user You just have to change the numeric value given in the url's . They can be seen as customer id. **URL'S effected:-** http://13.127.159.1/orders/orders.php?customer=13/ http://13.127.159.1/profile/16/edit/ http://13.127.159.1/forum/index.php?u=/user/profile/4

Observations: When we change the payload we can see the receipts of other users or customers.



POC: Here you can clearly see the receipt of another user.



Business Impact – Extremely High

A malicious hacker can read bill information and account details of any user just by knowing the customer id and User ID. This discloses critical billing information of users including:

•Mobile Number
•Bill Number
•Billing Period
•Total number of orders ordered by customer
•Bill Amount and Breakdown
•Phone no. and email address
•Address

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/blackmarket. Moreover, as there are no rate limiting checks, attackers can bruteforce the user_id for all possible values and get bill information of each and every user of the organisation resulting in a massive information leakage.

Recommendation: Take the following precautions:

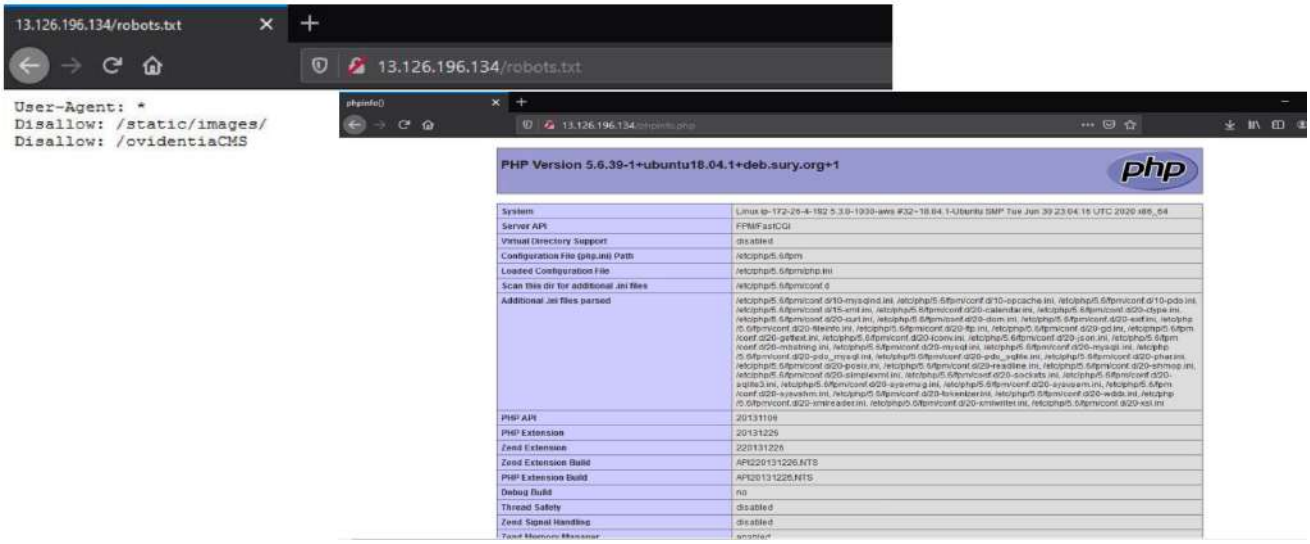
•Implement proper authentication and authorisation checks to make sure that the user has permission to the data he/she is requesting.
•Use proper rate limiting checks on the number of requests coming from a single user in a small amount of time.
•Make sure each user can only see his/her data only.

References: https://www.owasp.org/index.php/Insecure_Configuration_Management and https://www.owasp.org/index.php/Top_10_2013-A4-Insecure_Direct_Object_References

12. Directory Listings

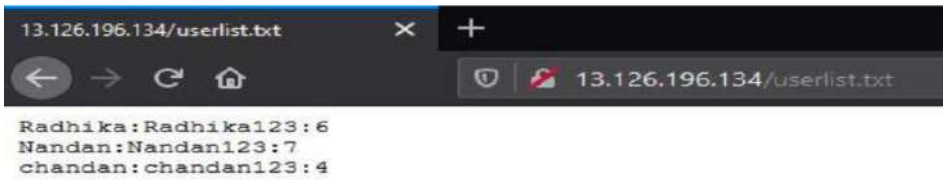
Directory listings (Moderate)	Below mentioned urls disclose server information. Affected URL : •http://13.126.196.134/phpinfo.php •https://13.126.196.134/robots.txt •http://13.126.196.134/composer.lock •http://13.126.196.134/composer.json •http://13.126.196.134/userlist.tx

Observation



POC:

1. In the above observation you can see that a hacker can go through these directories easily and gather as much information as he/she wants.
2. In Fact it also shows some accounts of sellers.



Business Impact – Moderate: 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. Information Disclosure due to default pages are not exploitable in most cases, but are considered as web application security issues because they allow malicious hackers to gather relevant information which can be used later in the attack lifecycle, in order to achieve more than they could if they didn't get access to such information.

Recommendation

1. Disable all default pages.
2. Enable multiple security checks.

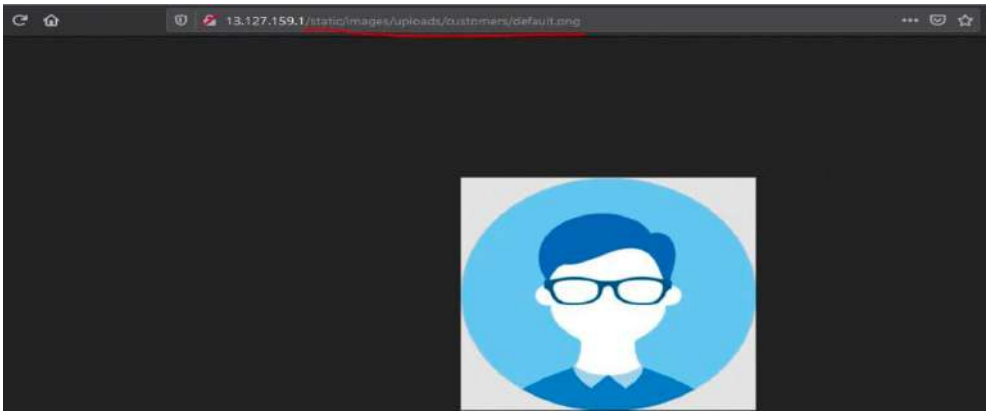
References: https://www.netsparker.com/blog/web-security/information-disclosure-issues-attacks/ and https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/information-disclosure-phpinfo/

13. Personal information leakage

Personal Information Leakage (Low)	Below mentioned urls disclose personal inforamtion **Affected URL :** • http://13.127.159.1/static/images/upload/customers/default.png • http://13.127.159.1/products/details.php?p_id=2

Observations:

Navigate to the URL, And you can see the whole path where everyone's photo is stored.



POC

- Here if you see the url , you will know that we just chnaged it little bit and we hit jackpot where we can see photos uploaded by customer and may more...

Business Impact – Moderate: Although this vulnerability does not have a direct impact on users or the server, it can help the attacker in mapping the personal information of any account and plan further attacks on any specific account.

Recommendations:

• You can apply encryption to the personal data.
• You can add authenticity and authorization to access the other data.

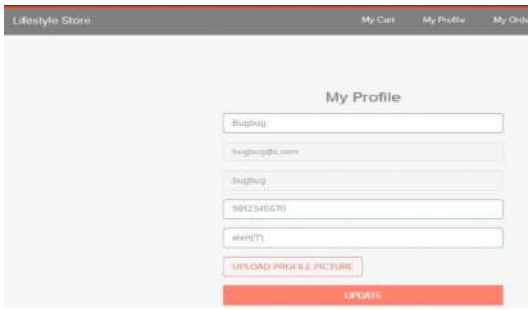
References: https://cipher.com/blog/25-tips-for-protecting-pii-and-sensitive-data/ and https://digitalguardian.com/blog/how-secure-personally-identifiable-information-against-loss-or-compromise

14.Client side and server side validation bypass

Observation: Here we intercepted the request and made changes in the contact number field.

POC: mobile number is saved as zero.

Client side and server side validation by pass (Low)	In below mentioned urls , we can easily bypass client side and server side validation **Affected URL :** • http://13.126.121.253/profile/16/edit/Affected parameter: •Contact Number (POST Parameter) Payload used: •123465890000000



Business Impact – Moderate: The data provided by the user ,if incorrect, is not a very big issue but still must be checked for proper validatory information.

Recommendations:

1. Implement all critical checks on server side code only.
2. Client-side checks must be treated as decoratives only.
3. All business logic must be implemented and checked on the server code.

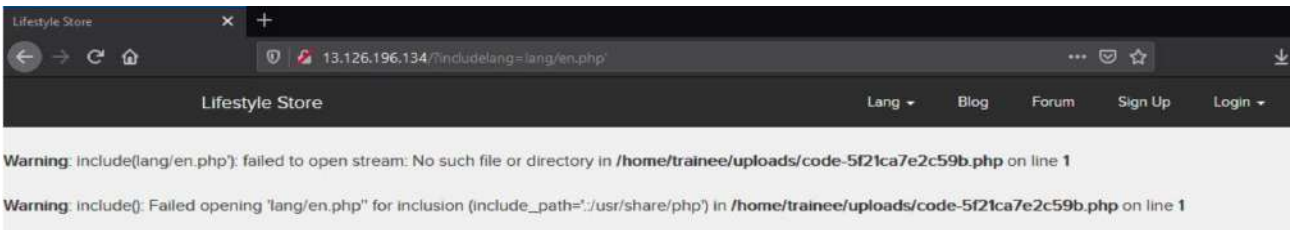
References:

http://projects.webappsec.org/w/page/13246933/Improper%20Input%20Handling and https://www.owasp.org/index.php/Unvalidated_Input

15. Default messages:

Observation & POC: Here we added payload as shown above and we got an error.

Default messages (Low)	In below mentioned urls ,if add a specific payload it will show deault messages **Affected URL :** •http://13.126.196.134/?includelang=lang/en.php **Payload** •en.php' (GET Parameter)



Business Impact – Moderate

Although this vulnerability does not have a direct impact on users or the server, it can help the attacker in mapping the server architecture and plan further attacks on the server.

Recommendations: Do not display the default error messages because it not only 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.

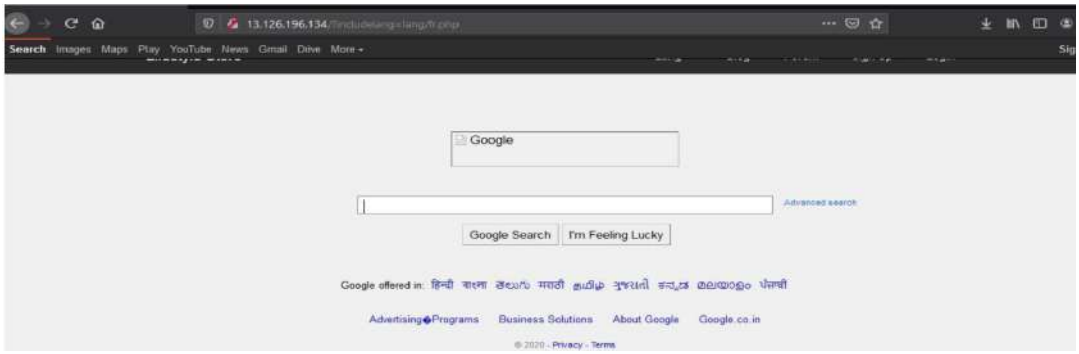
References: https://www.owasp.org/index.php/Improper_Error_Handling

16. Open redirecting:

Open Redirection (Low)	In below mentioned urls we can change the path of redirection **Affected URL :** • http://13.126.196.134/?inclludelang=lang/en.php • http://13.126.196.134/?inclludelang=lang/fr.php **Payload:-** • http://13.126.196.134/?inclludelang=https/www.google.com?lang/en.php

Observations: Here we made changes to the url according to the payload.

POC: we are redirected to google.



Business Impact – low:

An http parameter may contain a URL value and could cause the web application to redirect the request to the specified URL. By modifying the URL value to a malicious site.

Recommendations:

1. Disallow Offsite Redirects.
2. If you have to redirect the user based on URLs, instead of using untrusted input you should always use an ID, which is internally resolved to the respective URL.
3. If you want the user to be able to issue redirects you should use a redirection page that requires the user to click on the link instead of just redirecting them.
4. You should also check that the URL begins with http:// or https:// and also invalidate all other URLs to prevent the use of malicious URIs such as javascript:

References: https://cwe.mitre.org/data/definitions/601.html and https://www.hacksplaining.com/prevention/open-redirects

Conclusion

We were successfully able to find all the vulnerabilities and discussed their impacts and solutions. Hence completing the project along with the report. As we say above, some vulnerabilities were very harmful for the website, whereas, some were moderately harmful. Learning ethical hacking helped us identify them and solve them. I therefore find this skill very helpful and hope to work with it in the future.

Project solution:

ETHICAL HACKING TRAINING

PROJECT SOLUTION

The project web application we gave you, had 28 vulnerabilities.
Here is the breakdown of the various types of vulnerabilities that were present in the web application:

SQL injection - 2
Reflected and Stored Cross Site Scripting - 3
Insecure Direct Object Reference - 4
Rate Limiting Issues - 2
Insecure File Uploads - 1
Client Side filter bypass - 1
Server Misconfigurations - 1
Components with known vulnerabilities - 2
Weak Passwords - 2
Default files and pages - 3
File inclusion vulnerabilities - 1
PII Leakage - 1
Open Redirection - 1
Bruteforce Exploitation - 1
Command Execution Vulnerability - 1
Forced Browsing flaws - 1
Cross-Site Request Forgery - 1

Congratulations to all those who have been able to complete the project and have found all these vulnerabilities.
For those of you who have not been able to find them, it's not the end. A good ethical hacker strives till he/she achieves the target. So, if you still have time left in the training, we recommend you to try further and find the remaining vulnerabilities. All the best!