Lifestyle Store

E-Commerce Platform

Detailed Developer Report

Security Status — Extremely Vulnerable

- Hacker can steal all records in Lifestyle Store databases (SQLi)
- Hacker can take control of complete server including View, Add, Edit, Delete files and folders (Shell Upload and Weak Passwords)
- Hacker can change source code of application to host malware, phishing pages or even explicit content (Shell Upload)
- Hacker can get account details of another customer like by changing the number in edit profile link(IDOR)
- Hacker can get access to seller details and login into the website using customer of the month usernames(PII)
- Hacker can send multiple requests (Rate Limiting Flaw).
- Hacker can add /remove items in the cart (CSRF)
- Use off http instead of https.
- Cookie Flaws.

Vulnerability Statistics

Critical	Severe	Moderate	Low
13	15	8	4

Vulnerabilities:

No	Severity	Vulnerability	Count
1	Critical	SQL Injection	2
2	Critical	Insecure /Arbitrary File Uploads	2
3	Critical	Access to admin panel	1
4	Critical	Access via OTP Bypass	2
5	Critical	Unauthorized Access To Customer Details	4
6	Critical	Command Execution	2
7	Severe	Cross site scripting	2
8	Severe	Crypto Configuration Flaw	1
9	Severe	Common Passwords	2
10	Severe	Unauthorised availability of Details	7
11	Severe	Open Redirection	3
12	Moderate	Information disclosure due to Default Pages	5
13	Moderate	Unnecessary Details about Sellers	3

Vulnerabilities

No	Severity	Vulnerability	Count
14	Moderate	Components with known vulnerabilities	2
15	Low	Improper Server side and client side filters	2
16	Low	Default error display	2

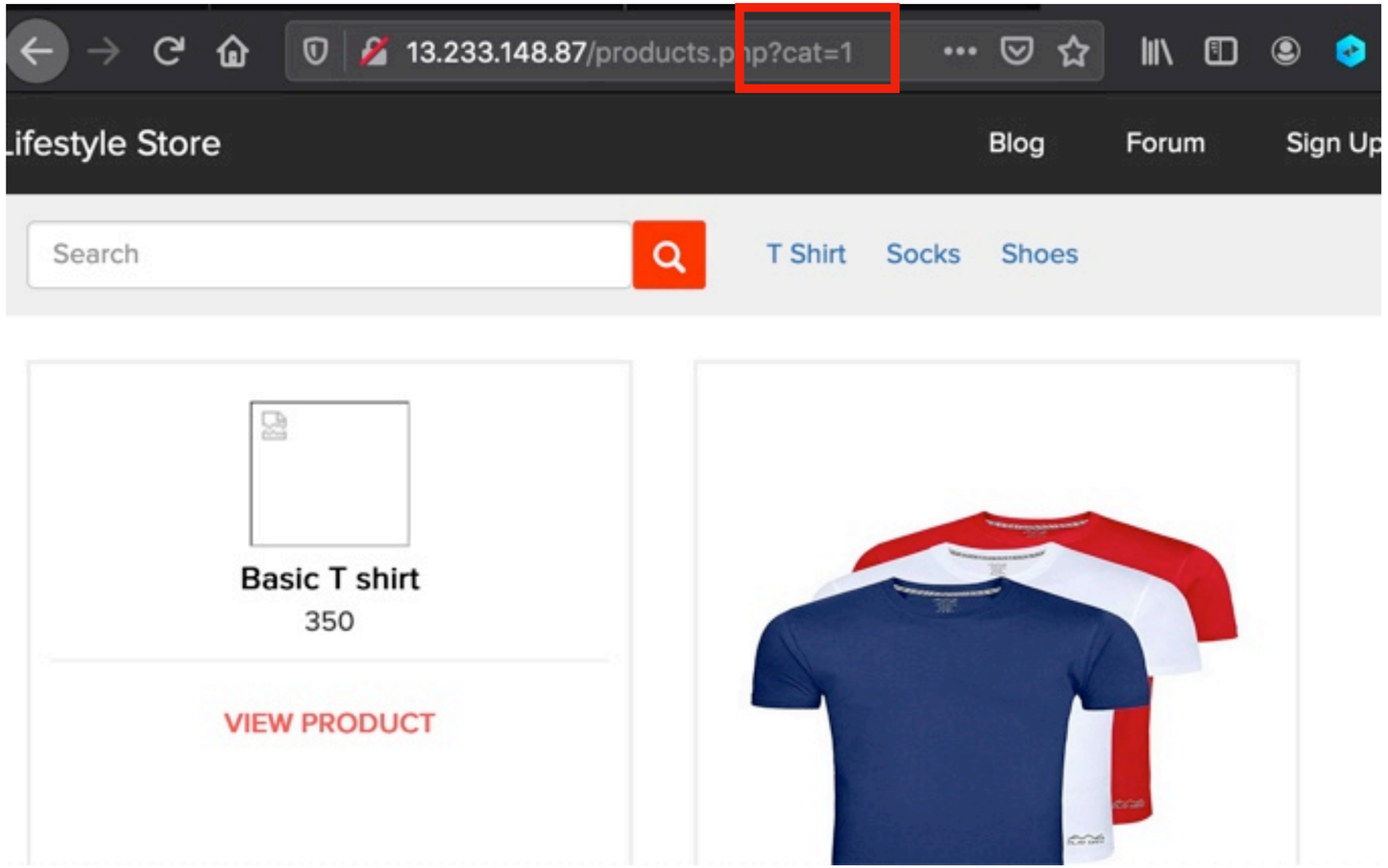
1.SQL Injection

It allows hacker to inject server side codes or commands. These are the flaws that allows a hacker to inject his own codes/commands into the web server that can provide illegal access to the data.

SQL Injection (Critical)	Below mentioned URL in the **Tshirt/socks/shoes module** is vulnerable to SQL injection attack **Affected URL :** • http://13.233.148.87/products.php?cat=1 **Affected Parameters :** • cat (GET parameter) **Payload:** • cat = 1’ **Affected URL :** • http://13.233.148.87/products.php?q=socks **Affected Parameters :** • q (GET parameter) **Payload:** • q=socks’

Observation

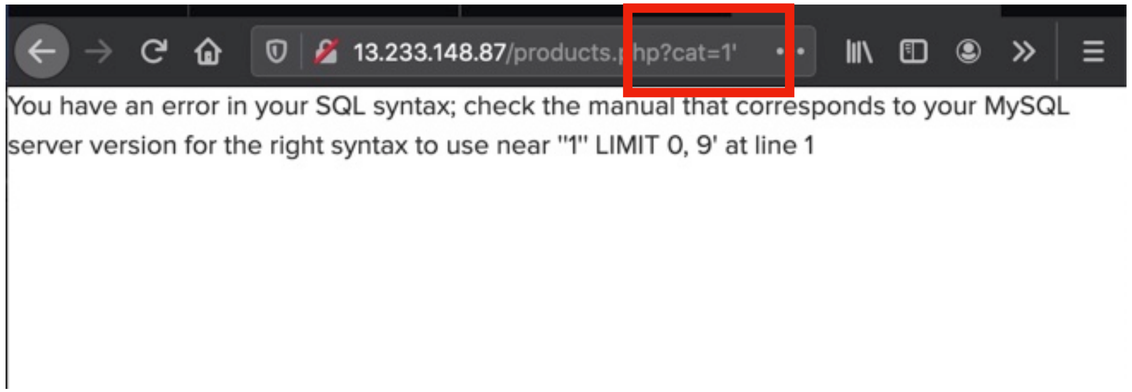
- After logging in as a customer, navigate to T shirts tab . Notice the GET parameter cat=1 in the URL:



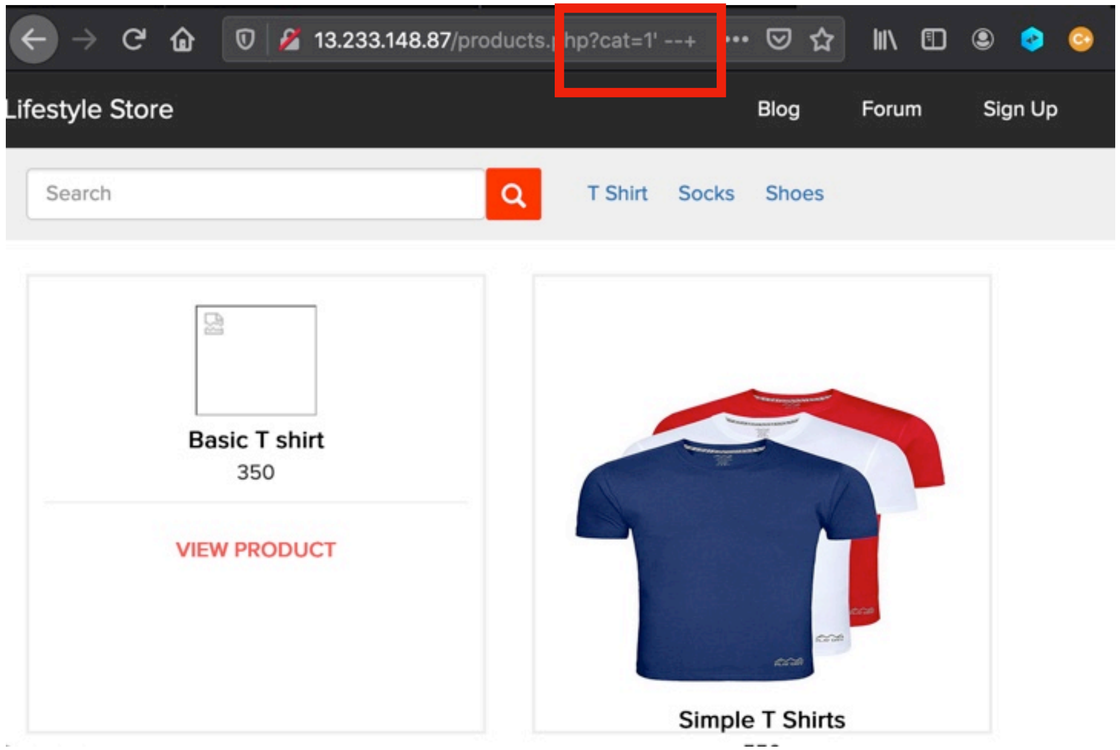
Observation

- We apply single quote in house parameter: **products.php?cat=1’ and we get complete MySQL error: (img 1)**
- We then put --+ : **products.php?cat=1’--+ and the error is removed confirming SQL injection:(img 2)**

img 1

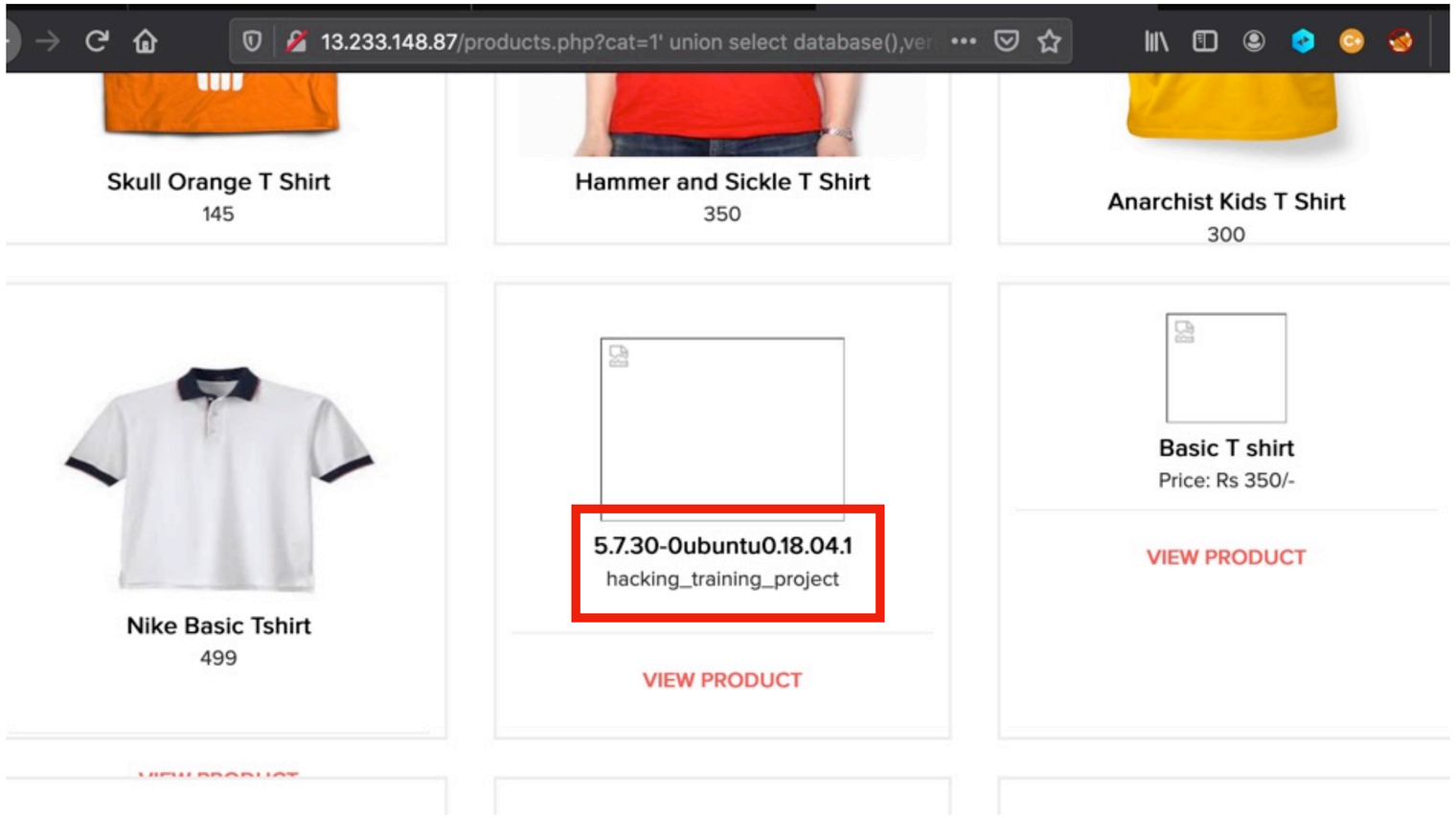


img 2



Proof of Concept (PoC)

- Attacker can execute SQL commands as shown below. Here we have used the payload below to extract the database name and MySQL version information:
  http://13.233.148.87/products.php?cat=1' union select database(),version(),database(),database(),version(),version(),version() --+



Proof of Concept (PoC)

- No of databases: 2
  - information_schema
  - hacking_training_project

- No of tables in SQL_Injection_V3: 10
  - brands
  - cart_items
  - categories
  - customers
  - order_items
  - orders
  - product_reviews
  - products
  - sellers
  - users

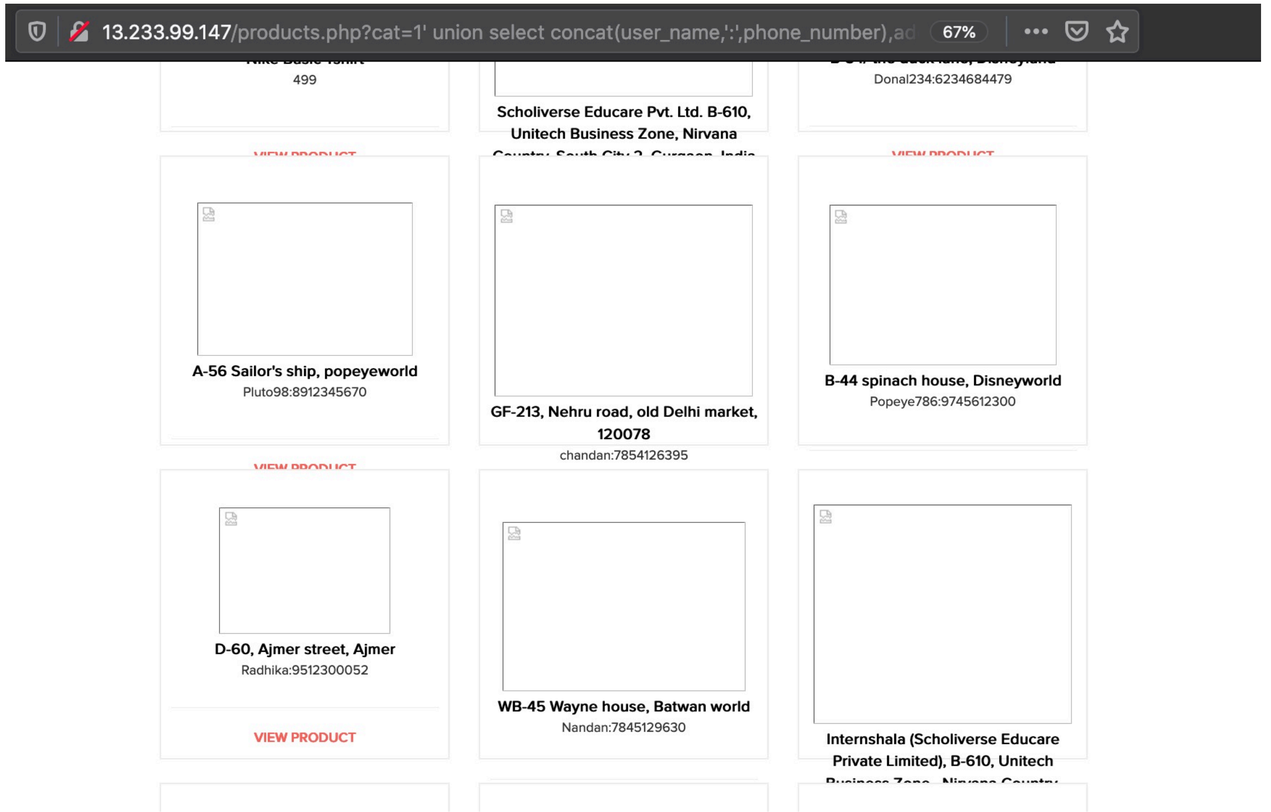
user_name	type	password	phone_number
admin	admin	$2y$10$Phrdr2F1sC9l2mG6jY5af.QbdJ7O6yasyHc/CZiNEchBPsWJiWuK2	8521479630
Donal234	customer	$2y$10$PM.7nBSP5FMaldXiM/S3s./p5xR6GTKvjry7ysJtxOkBqOJURAHsO	9489625136
Pluto98	customer	$2y$10$ba4bpp3nqfFRPB9.w.s4KeU36ecbRemyM6bj65FI/Q1Et0Qv1X9QK	8912345670
chandan	seller	$2y$10$4cZBEIrgthXdvT1hwUlivuFELe03rR.GIcdp03NjrlS0VeiOKLVDa	7854126395
Popeye786	customer	$2y$10$Fkv1RfwYTioW0w2CaZtAQuXVnhGAUjt/If/yTqkNPC5zTrsVm7EeC	9745612300
Radhika	seller	$2y$10$RYxNhOyV/G4g7OtFwpqYaexvHi8rF6XXui8kT1WtrfqhTutCA8JC.	9512300052
Nandan	seller	$2y$10$G.cRNLMEiG79ZFXElHg.R.o95334U0xmZu4.9MqzR5614ucwnk59K	7845129630
MurthyAdapa	customer	$2y$10$mzQGzD4sDSj2EunpCioe4eK18c1Abs0T2P1a1P6eV1DPR.11UubDG	8365738264
john	customer	$2y$10$GhDB8h1X6XjPMY12GZ1vDO7Y3en97u1/.oXTZLmYqB6F18FBgecvG	6598325015
bob	customer	$2y$10$kiUikn3HPFbuyTtK75lLNurxzqC0LX3eMGy0/Ux16JOoGJ7dCGKLq	8576308560
jack	customer	$2y$10$z/nyNlkRJ76m9ItMZ4N5lOeRxy6Gkqi9N/UBcJu5ZeO7eM7N4pTHu	9848478231
bulla	customer	$2y$10$HT5oiRMetqaZ7xGZPE9s2.MklyF4PnYDJHCWbm2w/xuKpjEEI/zjG	7645835473
hunter	customer	$2y$10$pB3U9iFxwBgSbl2AkBpiEeIBdhiYfWy9y.xV23q12gGbMCyn7N3g2	9788777777
asd	customer	$2y$10$At5pFZnRWpjCD/yNnJWDL.L3Cc4Cv0W8Q/WEHmWzBFqVIkBQFpCF2	9876543210
acdc	customer	$2y$10$J50B78.gpucuLTwpHwbcPedYcain.Yi.tsTLyQtK17FzdSpmIRRbi	9999999999
FindMe	customer	$2y$10$ieLZsBhtXY0N92Wyo3o5y.BQJO4zd7tpcF18Xv61F/FhyBT6.zfNa	9999999999

Business Impact - Extremely High

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.

Below is the screenshot of some information extracted from users table which shows user credentials being leaked .Since the passwords are hashed ,the risk is comparatively low .

Attacker can use this information to attack the users and 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.



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

References

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

https://en.wikipedia.org/wiki/SQL_injection

2.Insecure /Arbitrary File Uploads

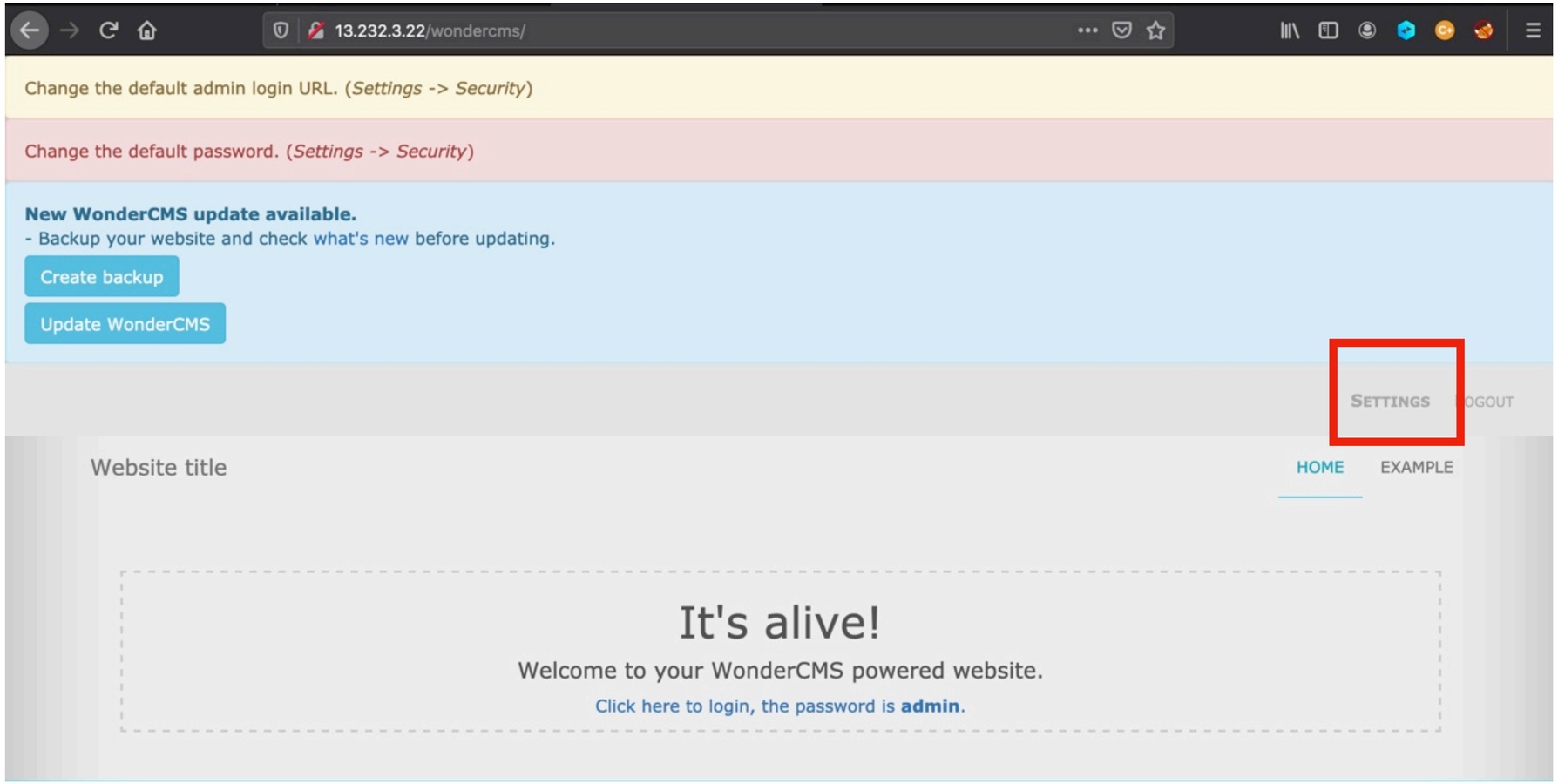
This happens when applications do not implement proper file type checking and allow uploading of files of different file formats. For example, a PHP file instead of a jpeg profile picture.

Insecure /Arbitrary File Uploads (Critical)	The attacker can upload insecure shells and files and gain access over the entire database and login as the admin and the vesion is known to have vulnerabilities . **Affected URL :** • http://13.232.3.22/wondercms/ **Affected Parameters :** • File Upload (POST parameter) The attacker can upload files with extension other than .jpeg . **Affected URL :** • http://13.232.3.22/profile/2/edit/ **Affected Parameters :** • Upload Profile Photo (POST parameter)

Observation

- Login using any of the user's details on the **Lifestyle store** and navigate to **Blog** tab . Now click on **Login** and put the password - **admin.**

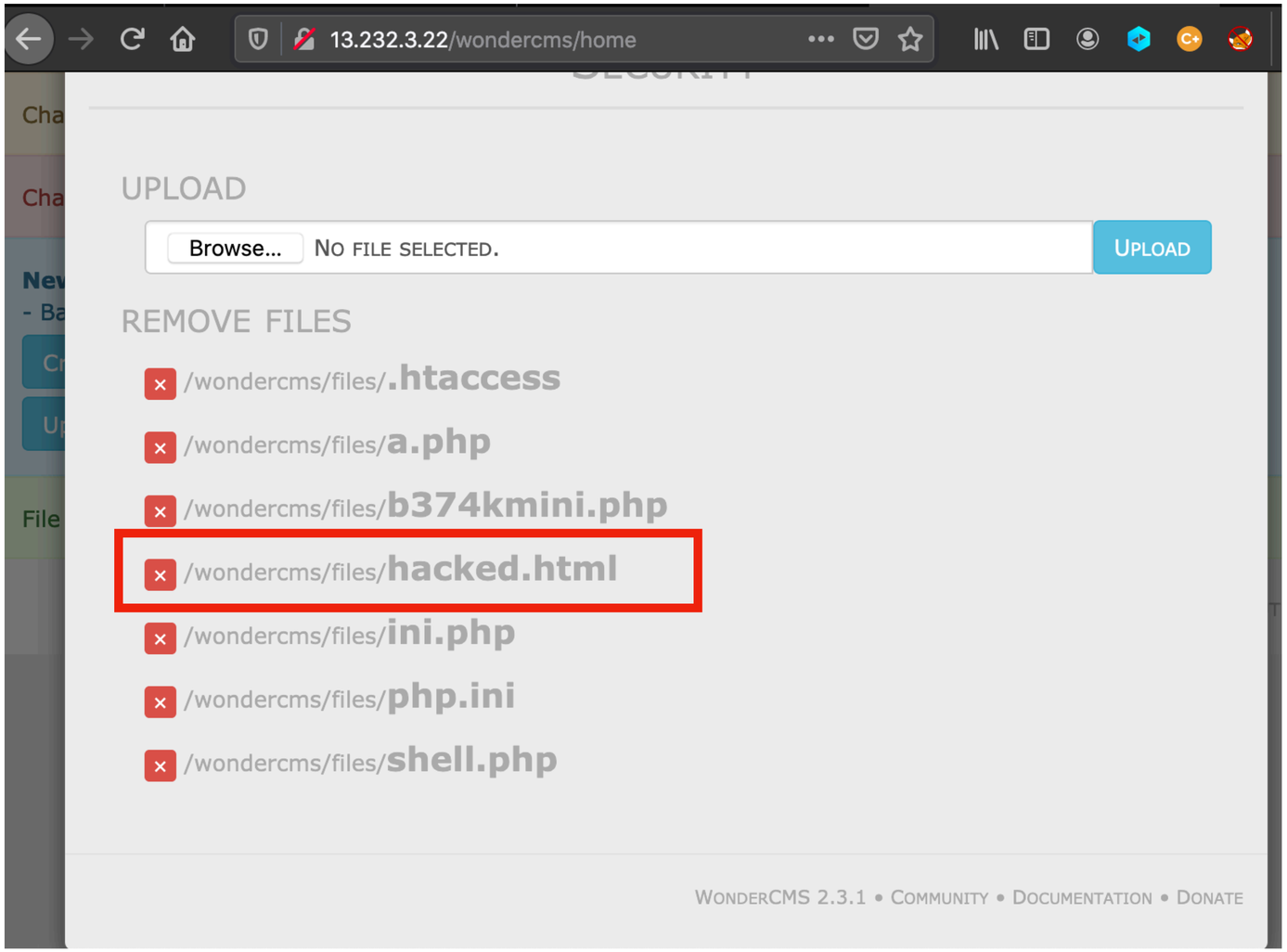
- You will see the following page and then click on **Settings** tab.



Observation

- Click on **Files** tab . Here hacker can upload the file like shown **.**

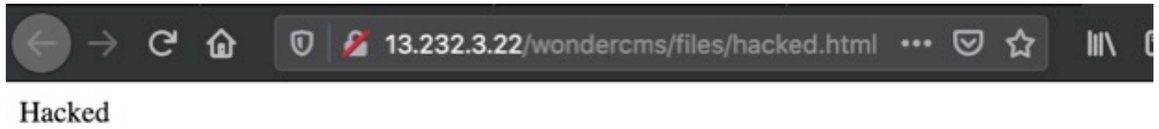
- Click on the uploaded file **hacked.html** and it will be opened.



Proof of Concept (PoC)

- Weak password - **admin**.
- Arbitrary File Inclusion.

Below is the result of the uploded file in the previous slide likewise some malicious shell can be uploaded as well.



Business Impact – Extremely High

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

- Change the **Admin password** to something strong and not guessable.
- The application code should be configured in such a way, that it should block uploading of malicious files extensions such as exe/ php and other extensions with a thorough server as well as client validation. CVE ID allocated: **CVE-2017-14521.**
- Rename the files using a code, so that the attacker cannot play around with file names.
- Use static file hosting servers like CDNs and file clouds to store files instead of storing them on the application server itself.

References

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

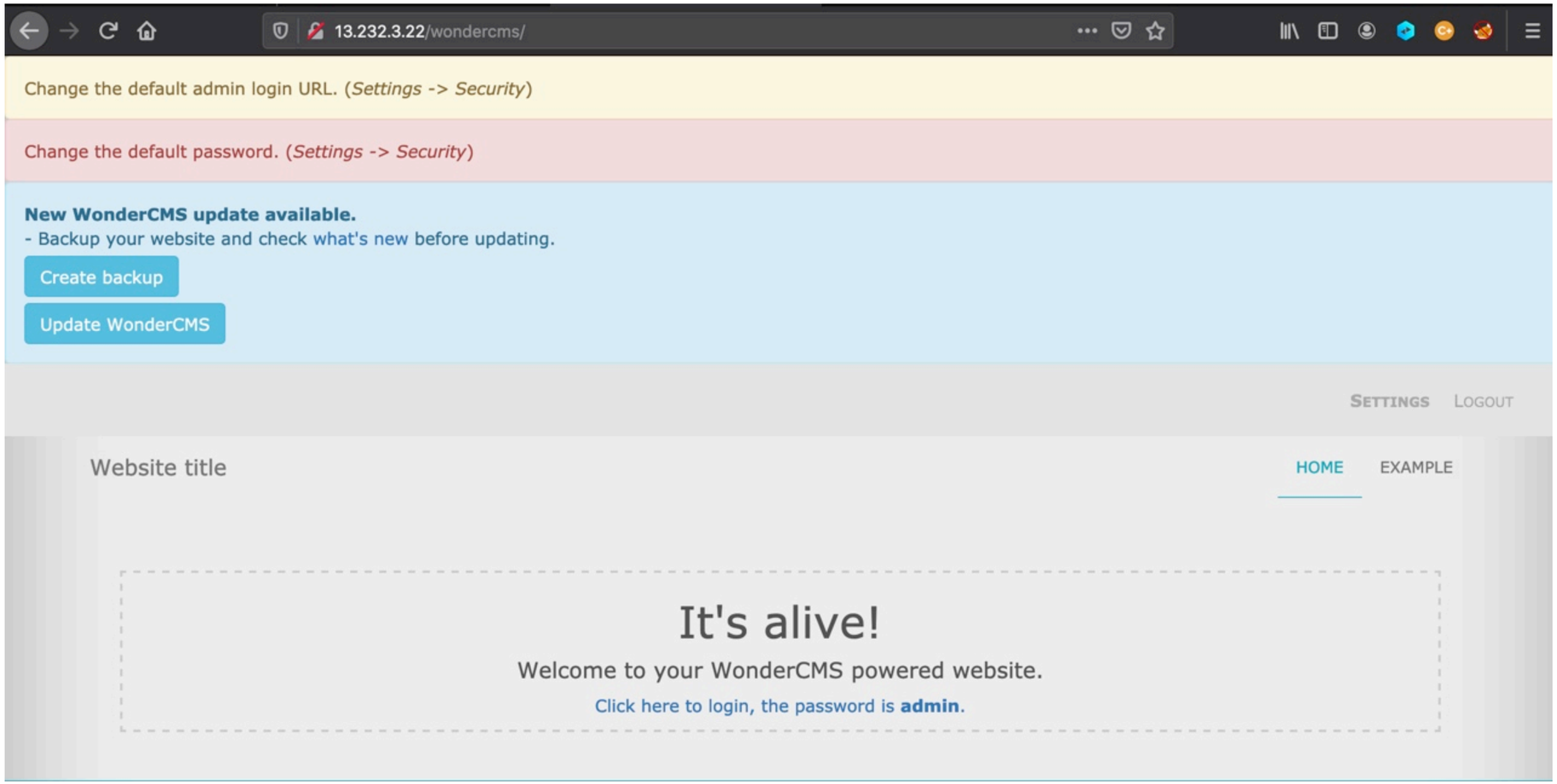
https://www.opswat.com/blog/file-upload-protection-best-practices

3.Access to admin panel

Access to admin panel (Critical)	Below mentioned URL is vulnerable to Arbitrary File Upload and making other admin level changes. **Affected URL :** • http://13.232.3.22/wondercms/loginURL

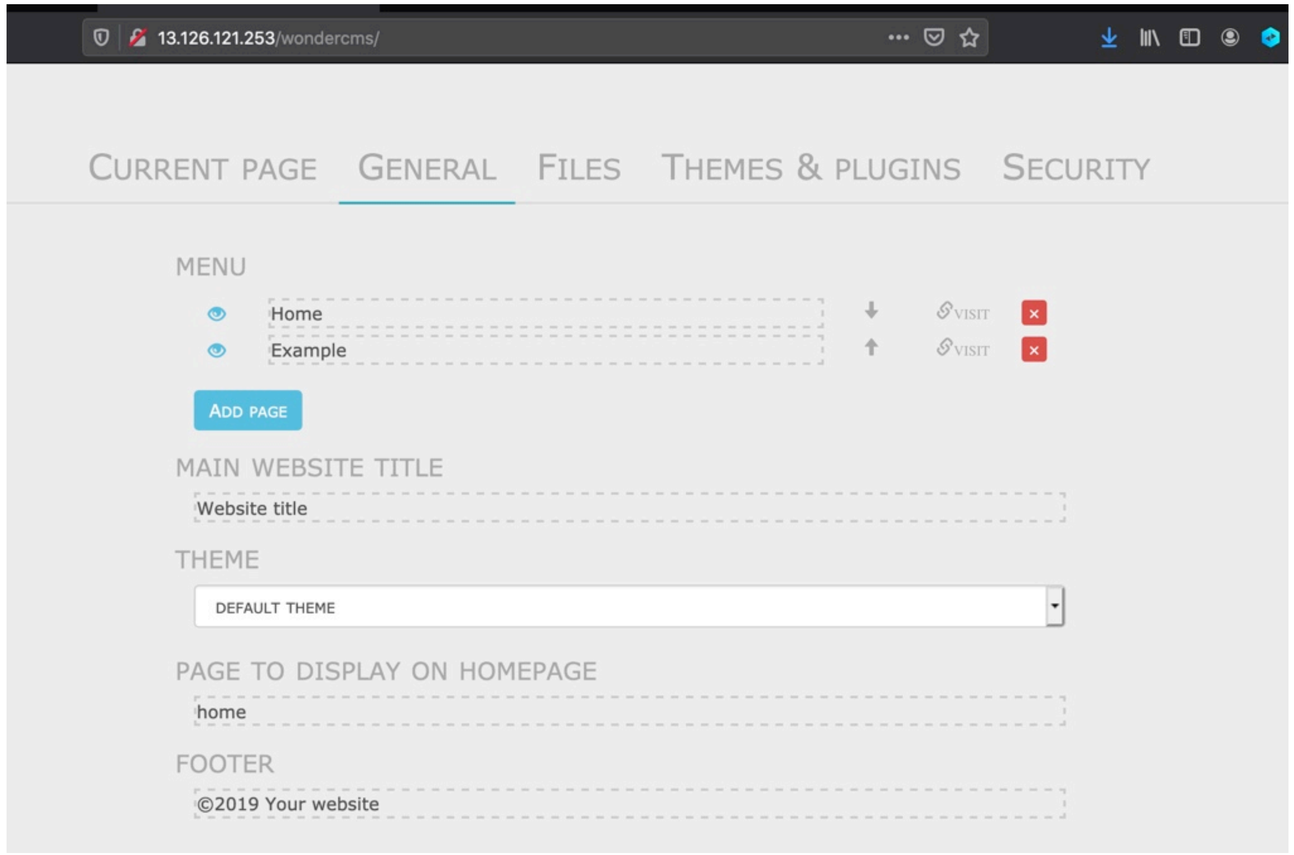
Observation

When we navigate to http://13.232.3.22/wondercms/ url ,we get the **password on the page and login as : admin** in the url http://13.232.3.22/wondercms/loginURL .



Proof of Concept (PoC)

Hacker can change the admin login password making the actual admin unable to login the next time .
Hacker can also add and delete pages.



Business impact - Extremely High

- Using this vulnerability ,the attacker can get complete access to the blog of the website.
- The attacker can change the password or even change the url of the admin panel and restrict the admin to access it.
- Even pages can be created and deleted along with editing.
- Files can be added (without verification) and hence can be dangerous to the entire website,as the control of the entire website can be taken.

Recommendation

- The default password should be changed and a strong password must be setup.
- The admin url must also be such that its not accessible to normal users.
- Password changing option must be done with 2 to 3 step verification.
- Password must be at least 8 characters long containing numbers,alphanumerics,etc.
- All the default accounts should be removed.
- Password should not be reused.

References

https://www.owasp.org/index.php/Testing_for_weak_password_change_or_reset_functionalities_(OTG-AUTHN-009)

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

https://www.us-cert.gov/ncas/alerts/TA13-175A

4.Access via OTP Bypass

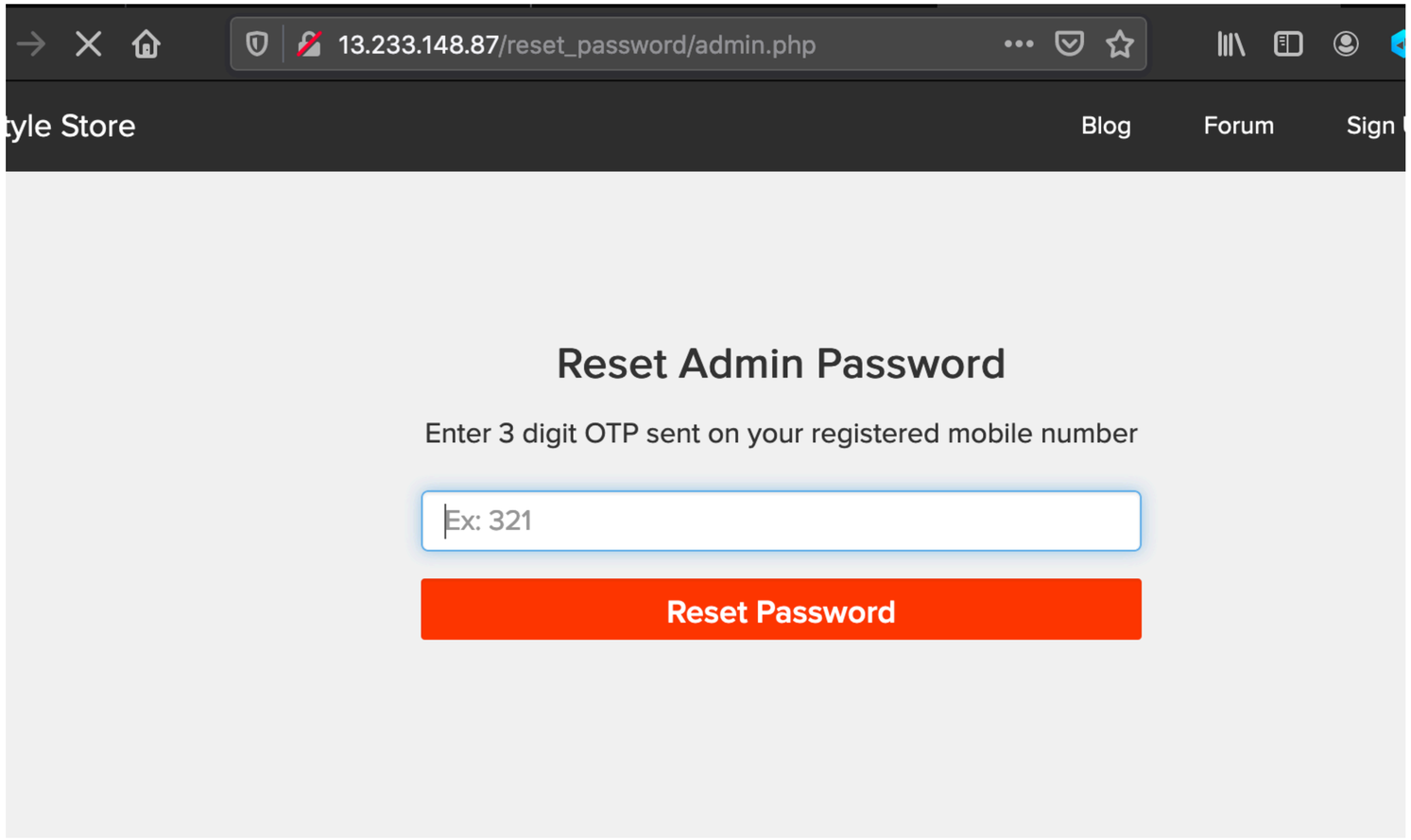
Access via OTP Bypass(Critical)	The admin dashboard at the below mentioned URL has 3 digit otp allowing brute forcing the otp and reset the password and gaining access. **Affected URL :** • http://13.233.148.87/reset_password/admin.php **Affected Parameters :** • OTP (GET parameters) **Payload used :** • otp=227

4.Access via OTP Bypass

Access via OTP Bypass(Critical)	The coupon code at the below mentioned URL can be bruteforced. **Affected URL :** • http://15.206.125.83/cart/cart.php **Affected Parameters :** • apply_coupon(POST parameters) **Payload used :** • UL-1056

Observation

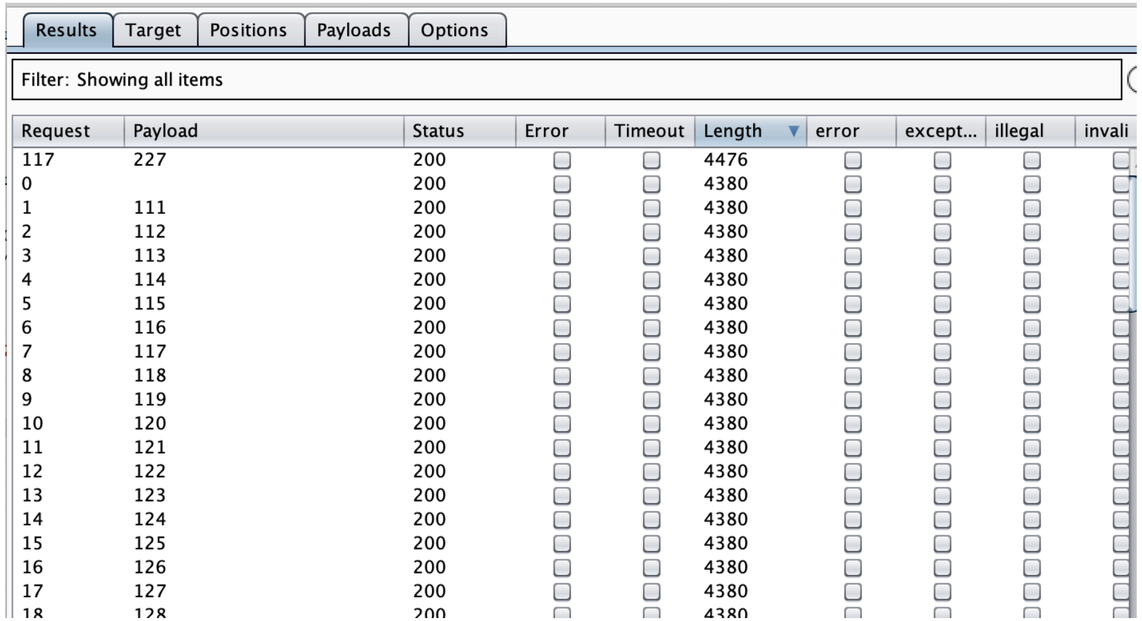
- Navigate to http://13.233.148.87/reset_password/admin.php You will see reset password page via OTP. Enter random otp while capturing requests in a local proxy .



Observation

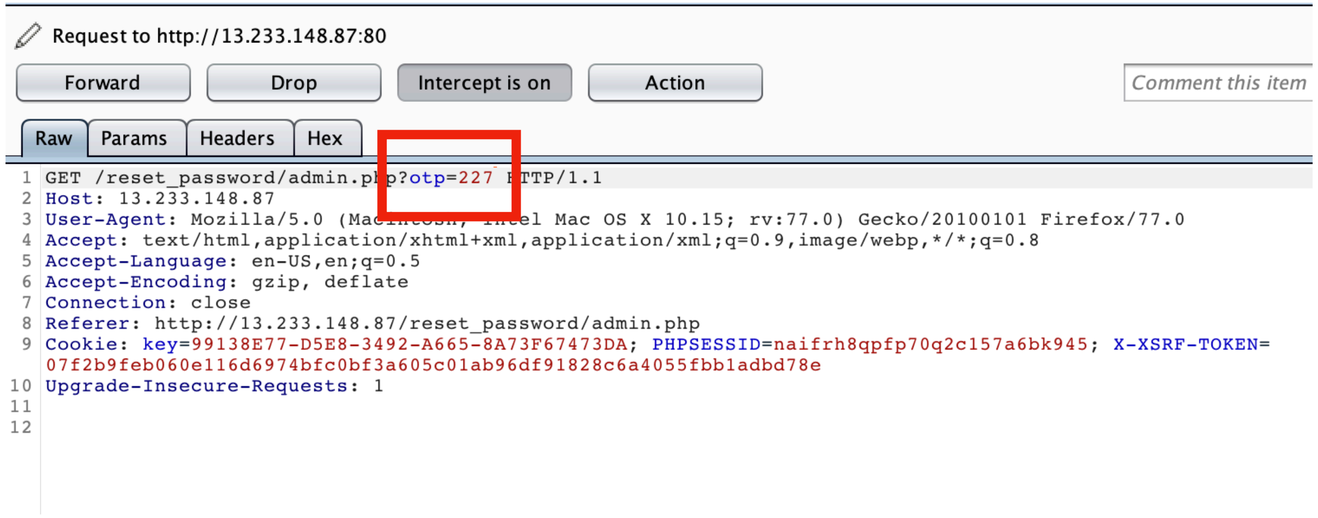
- On brute forcing the 3 digit otp , under the length column the value which is distinct from others yields the correct **otp - 227 (img 1)**
- Enter this otp in the captured request **(img 2)**

img 1



Request	Payload	Status	Error	Timeout	Length	error	except...	illegal	invali
117	227	200			4476				
0		200			4380				
1	111	200			4380				
2	112	200			4380				
3	113	200			4380				
4	114	200			4380				
5	115	200			4380				
6	116	200			4380				
7	117	200			4380				
8	118	200			4380				
9	119	200			4380				
10	120	200			4380				
11	121	200			4380				
12	122	200			4380				
13	123	200			4380				
14	124	200			4380				
15	125	200			4380				
16	126	200			4380				
17	127	200			4380				
18	128	200			4380				

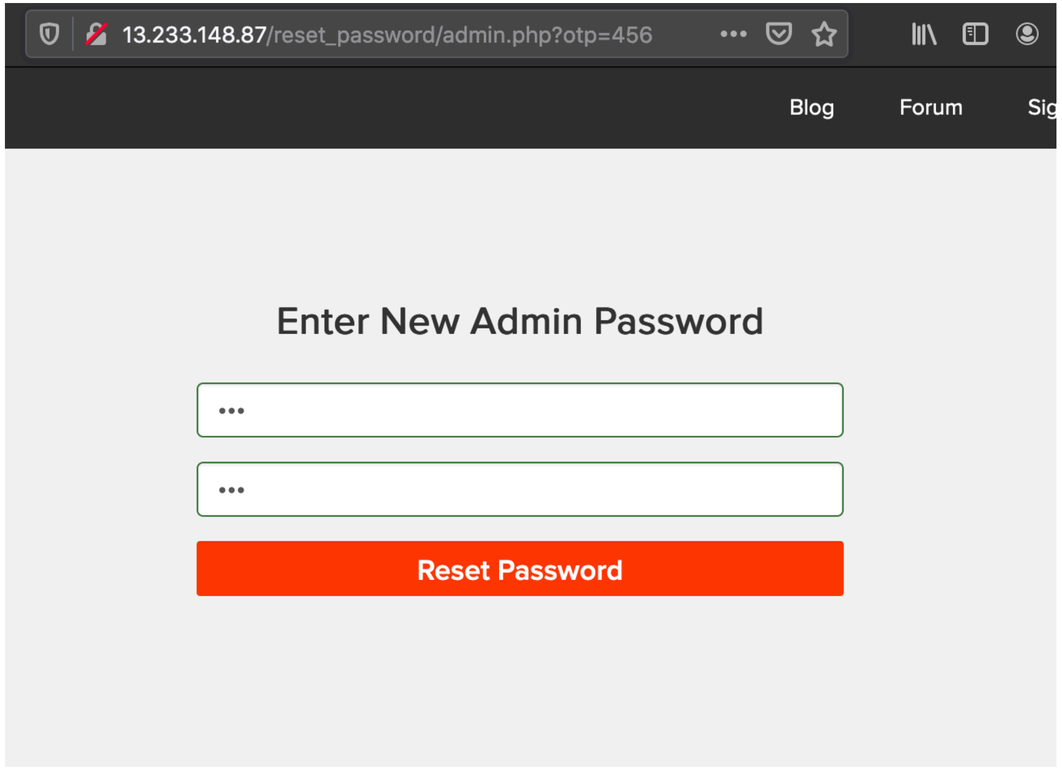
img 2



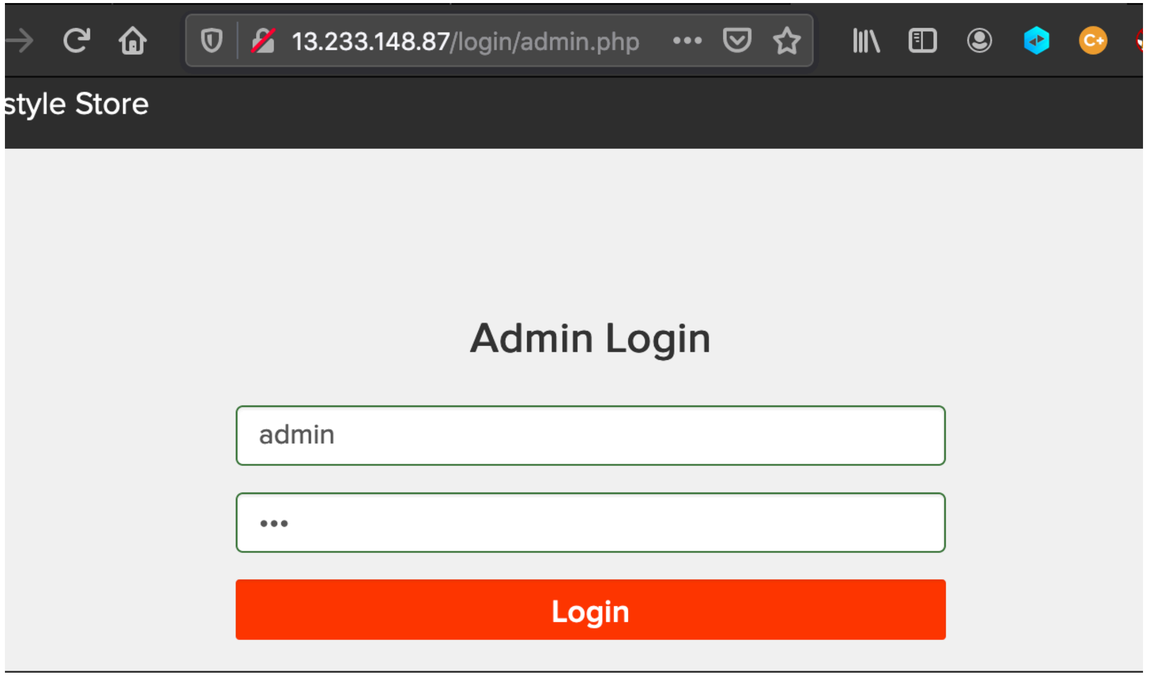
Observation

- You will be navigated to the reset password page .Here change the password (img 1).
- Navigate to http://13.233.148.87/login/admin.php. Enter **username-admin and password** (img 2).

img 1

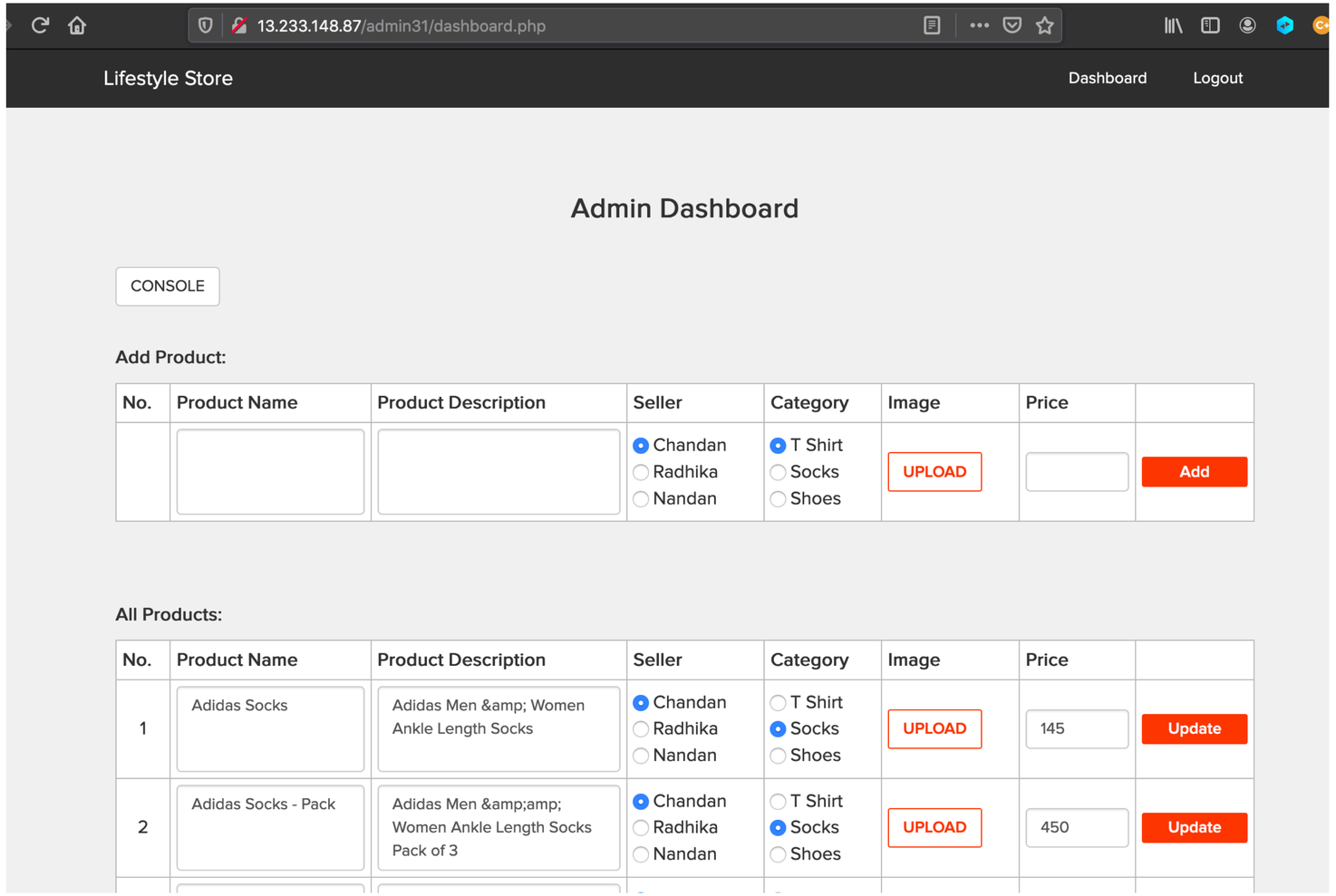


img 2



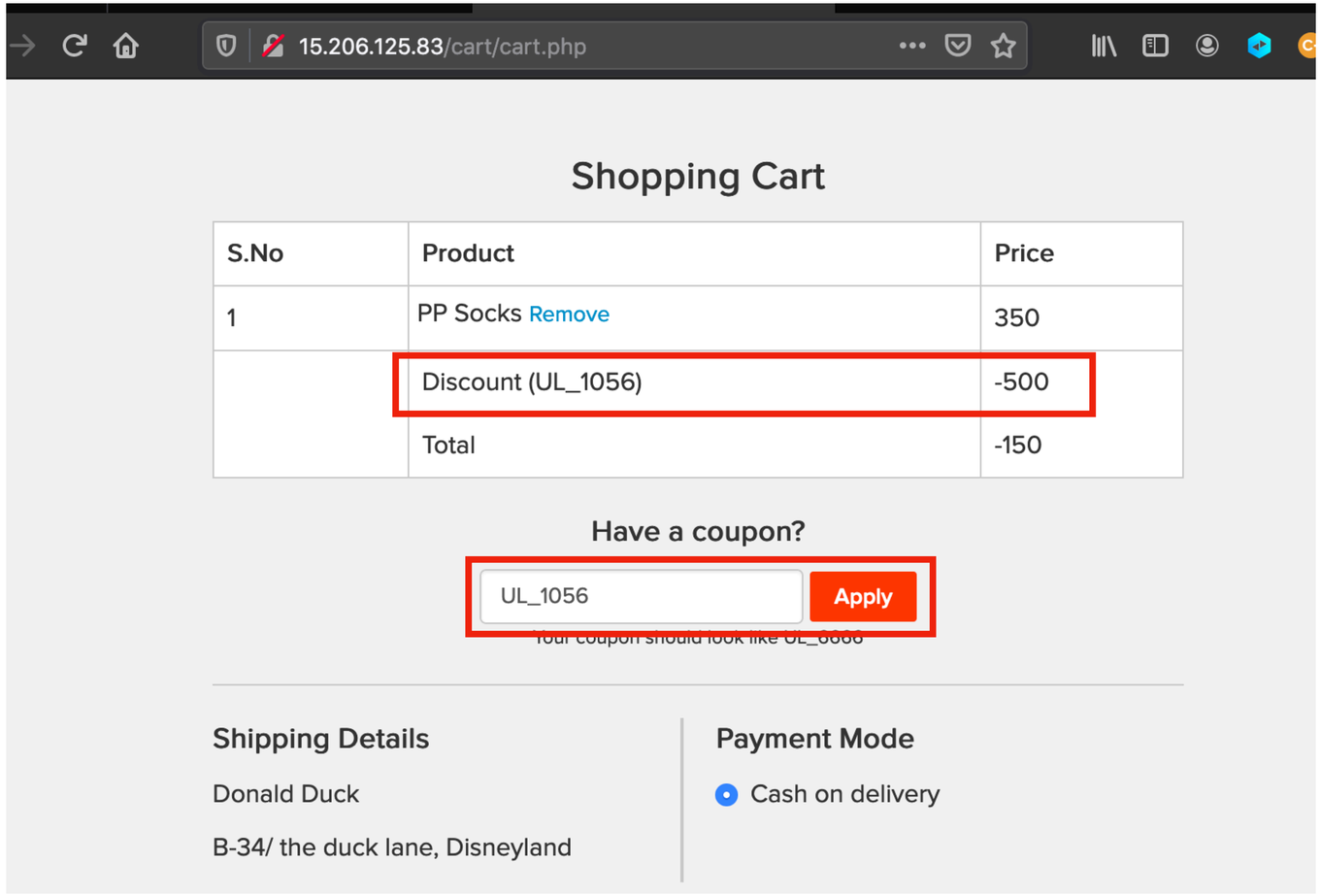
Observation

- You will be redirected to the admin dashboard where you can see the details of all the users/ sellers/customers.



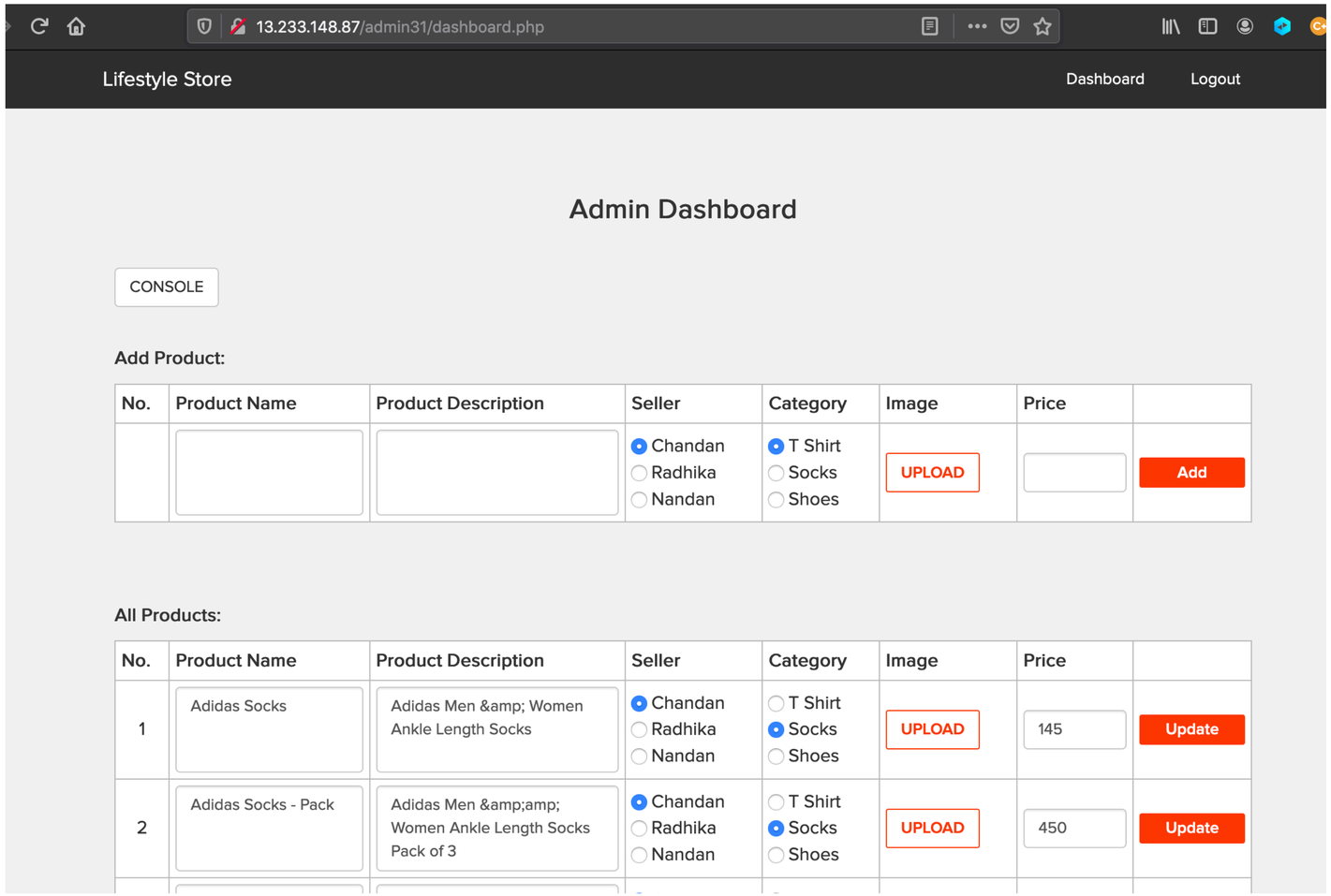
Proof of Concept (PoC)

At url http://15.206.125.83/cart/cart.php coupon code - **UL-1056** is applied.



Business Impact – Extremely High

A malicious hacker can gain access to any account and change the information about the products. This may lead to defamation of the seller and the website which the customer trusts.
Attacker once logs in can then carry out actions on behalf of the admin which could lead to serious loss to any user.



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

5.Unauthorised Access To Customer Details

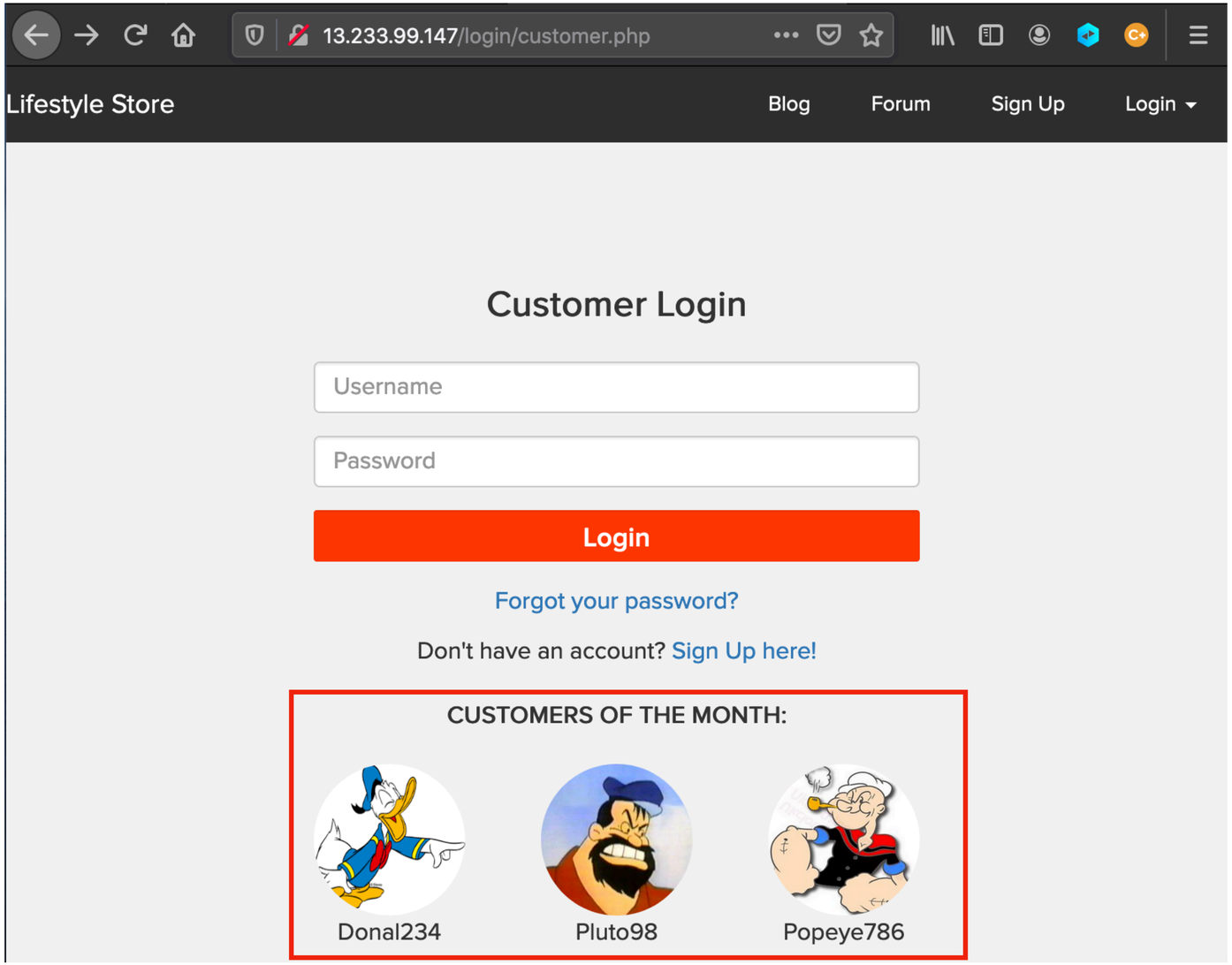
Unauthorised Access To Customer Details (Critical)	The below mentioned login page allows you to change password without verification and view details of other customers (CSRF). **Affected URL :** • http://13.233.99.147/reset_password/customer.php **Affected Parameters :** • Reset Password button (POST parameter) We can change the details of the user's account details. **Affected URL :** • http://13.232.3.22/profile/2/edit/ **Affected Parameters :** • Update button (POST parameter)

5.Unauthorised Access To Customer Details

Unauthorised Access To Customer Details (Critical)	Similarly after gaining access over the account , Hacker can remove/add/confirm items in the cart of the user (CSRF). **Affected URL :** • http://13.232.3.22/cart/cart.php **Affected Parameters :** • Remove option (POST parameter) **Affected URL :** • http://13.232.3.22/cart/cart.php **Affected Parameters :** • Confirm order option (POST parameter)

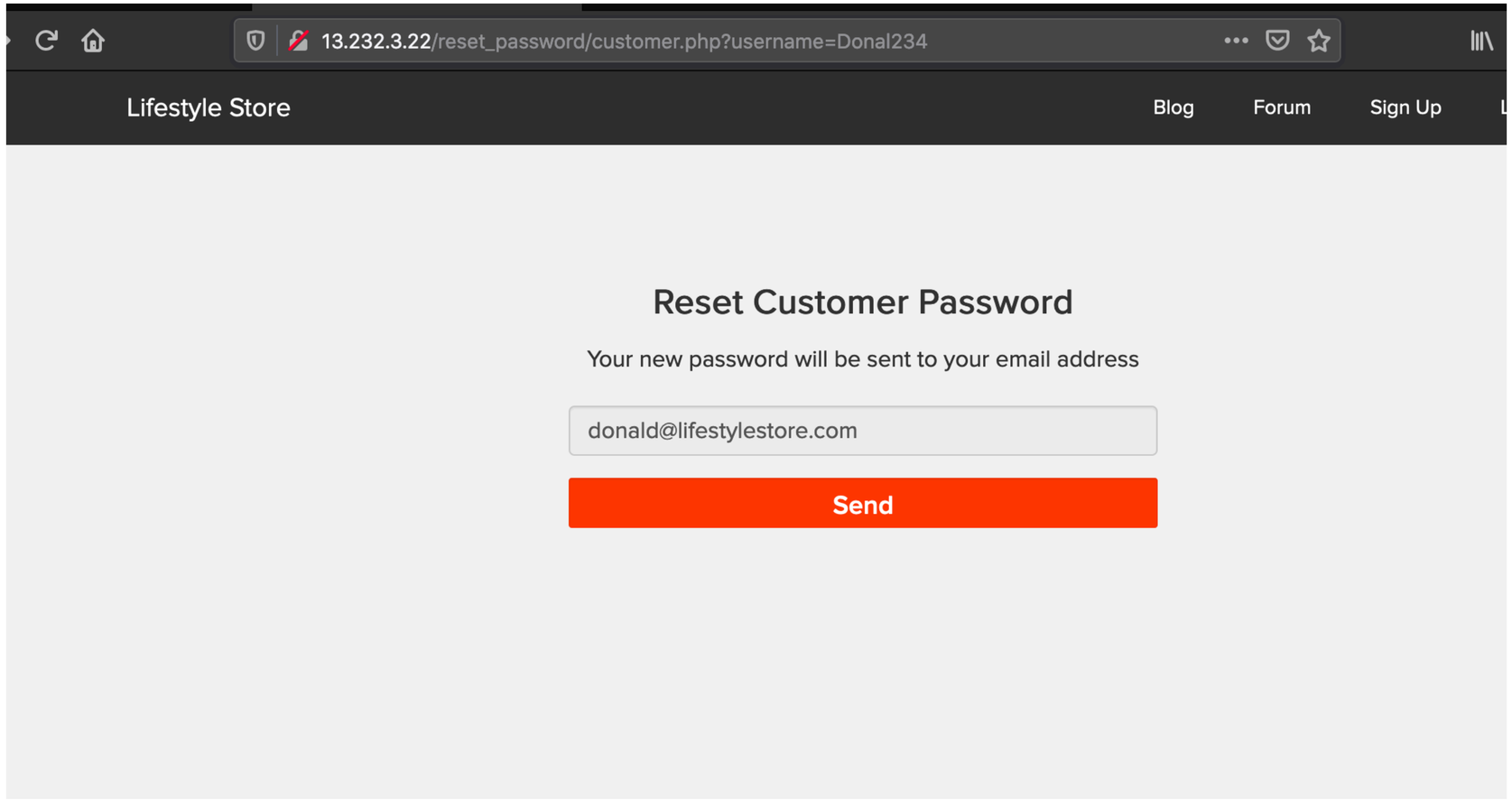
Observation

- Navigate to http://13.233.99.147/login/customer.php. Copy any of the **CUSTOMERS OF THE MONTH's** username.



Observation

- Navigate to http://13.233.99.147/reset_password/customer.php .
- Paste the copied username and click on **Reset Password** button. You will be redirected to the following page **.** Hit send.

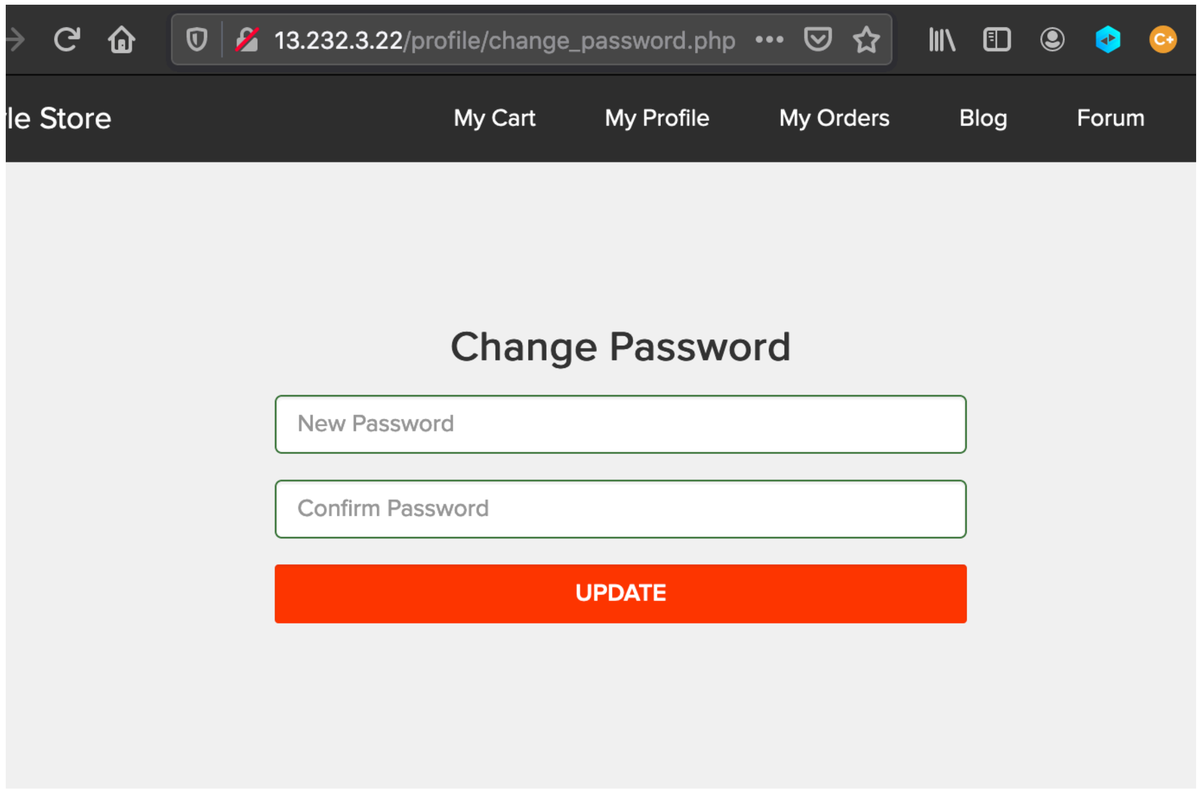


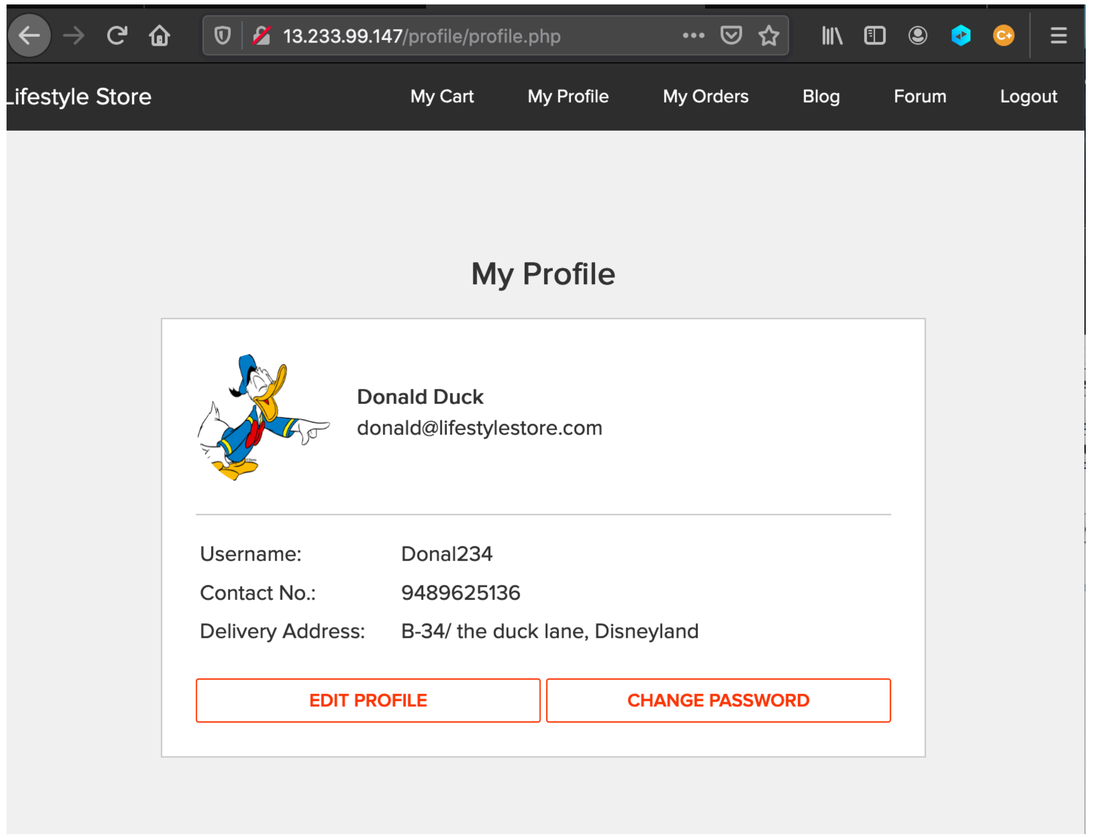
Observation

- You will be redirected to the following page. Then click on **click here.**
- Then you can change the password **.**

Proof of Concept (PoC)

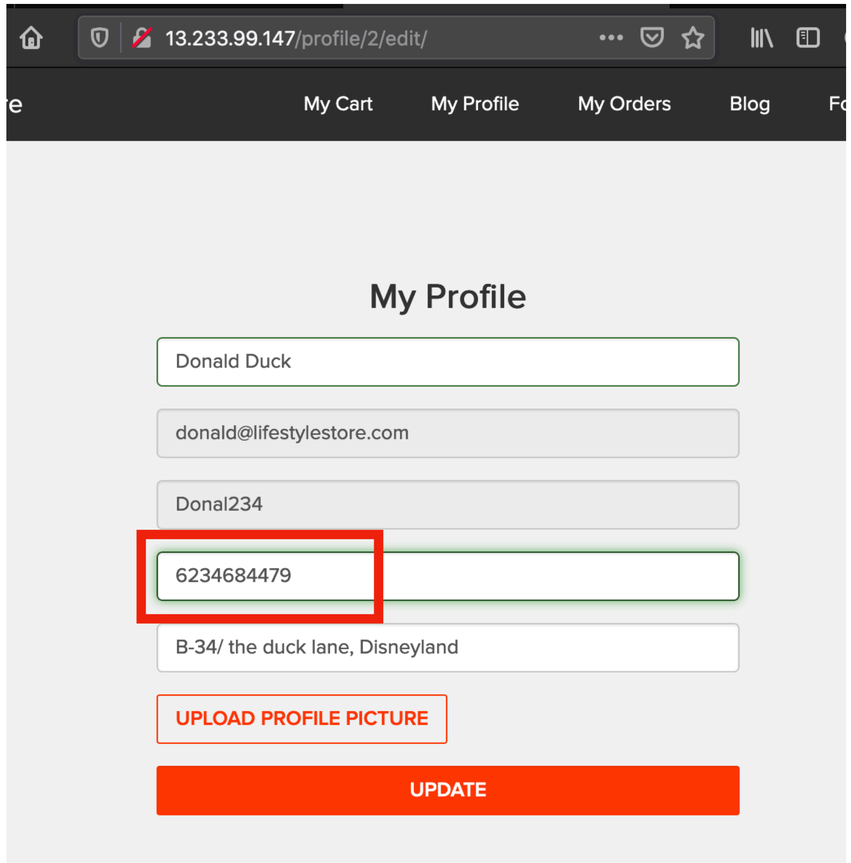
- Attacker can change the details and password of the customer easily and can place orders on user's behalf.

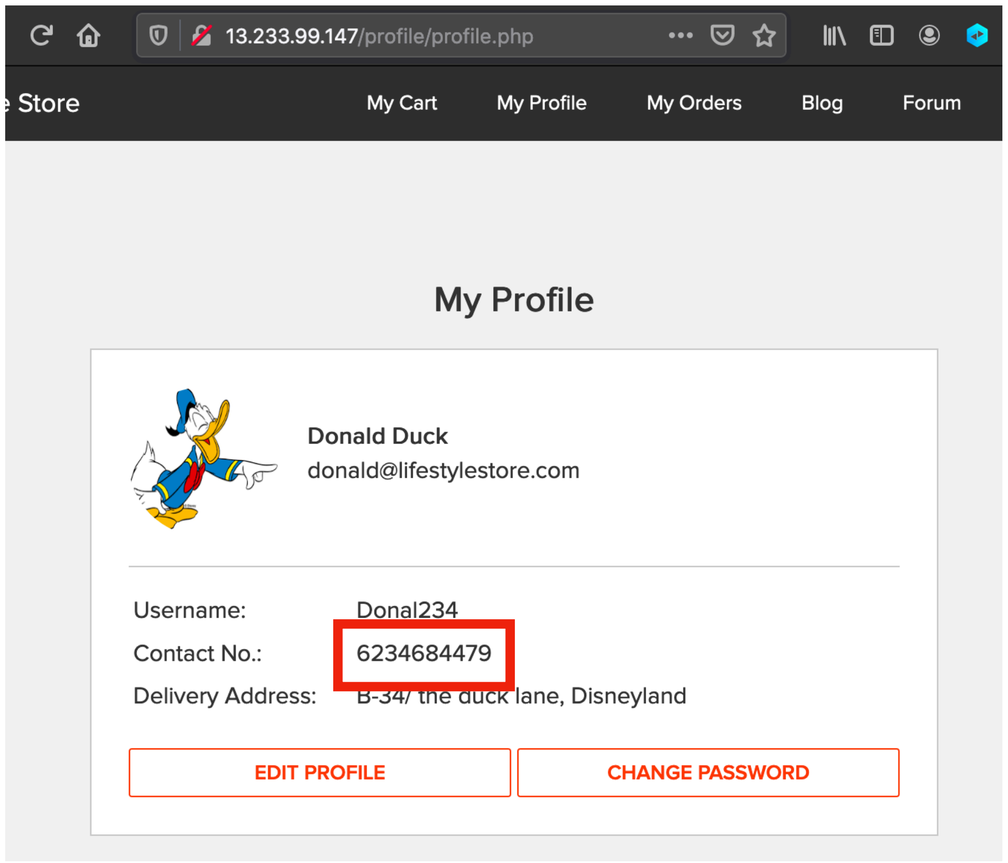




Business Impact - Very High

- A malicious hacker can gain complete access to customers's account just by clicking on **forgot password**. This leads to complete compromise of personal user data of the 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. Below are the screenshots of changing the phone number of the attacked user.





Recommendation

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

References

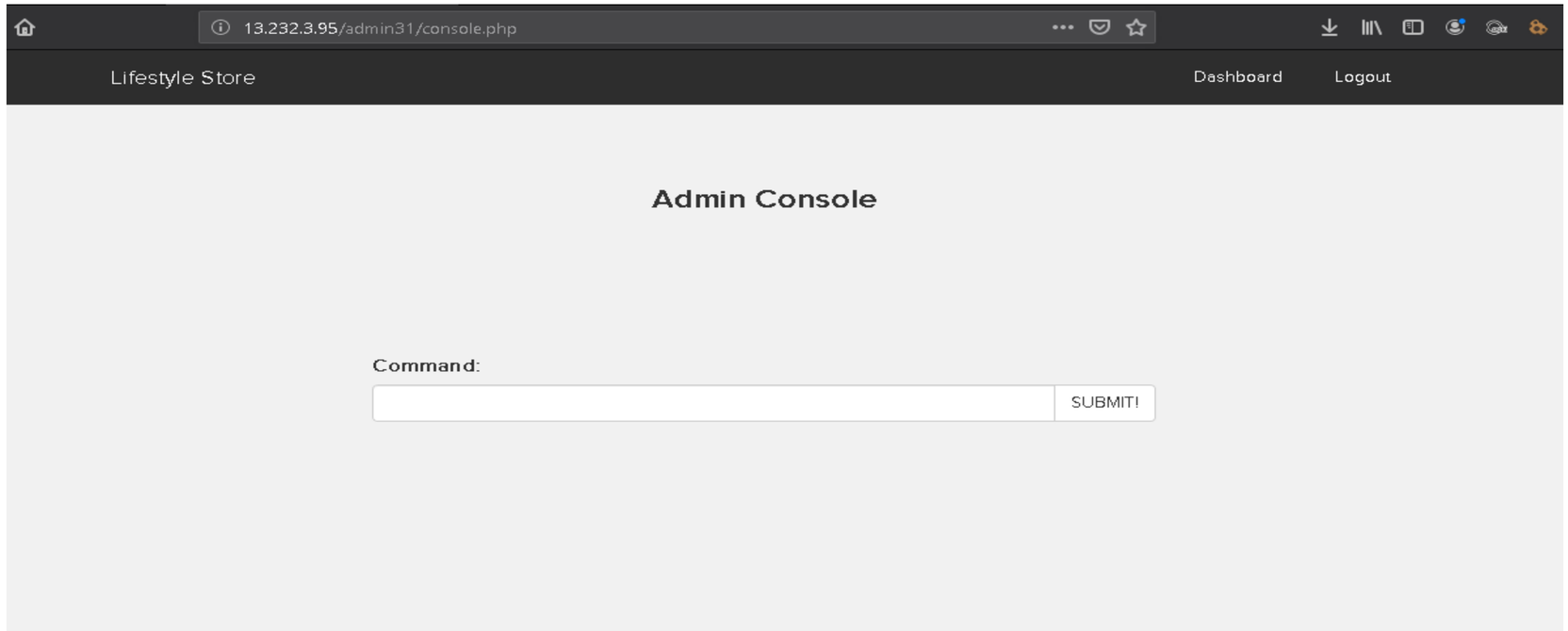
https://www.netsparker.com/blog/web-security/csrf-cross-site-request-forgery/
https://digitalguardian.com/blog/how-secure-personally-identifiable-information-against-loss-or-compromise

6.Command Execution Vulnerability

Command Execution Vulnerability (Critical)	Below mentioned URLs is vulnerable to Command Execution. **Affected URL :** • http://13.233.148.87/admin31/console.php • Shell can be uploaded at files tab to access the server details at http://13.232.3.22/wondercms/ **Affected Parameters :** • Command (POST parameter)

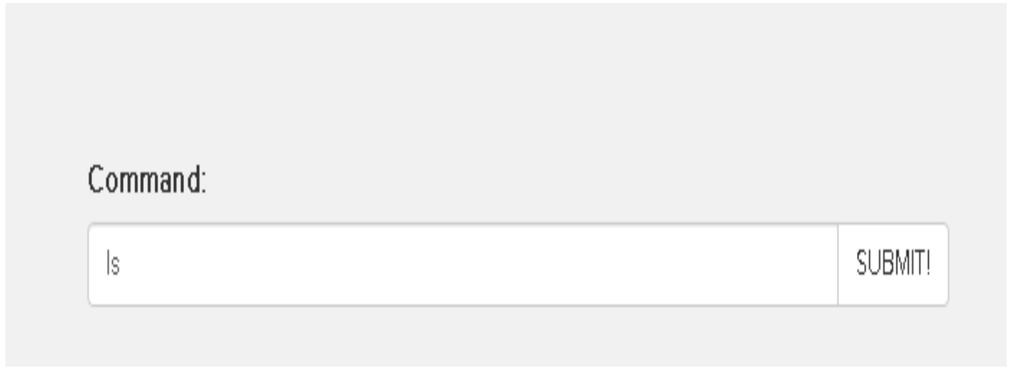
Observation

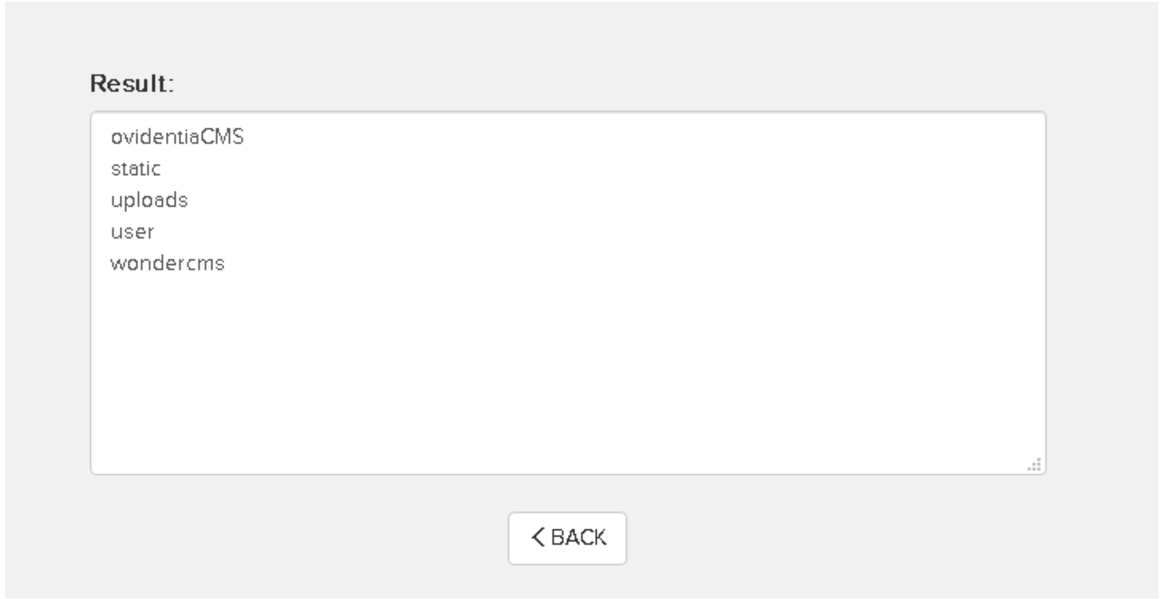
- Navigate to http://13.233.148.87/admin31/console.php after logging in as the **admin** and you will see the following page.



Proof of Concept (PoC)

When command **ls** is entered the following output is visible.





Business Impact – High

- If the attacker enters into the admin account and finally to the console url ,the he can put in any malicious code to extract or even edit data ,as he the has the admin privileges.
- Other than entering malicious code , the attacker can even get the details of the websites and its components like its version and hence find vulnerabilities to exploit them.
- If successfully exploited, impact could cover loss of confidentiality, loss of integrity, loss of availability, and/or loss of accountability.

Recommendation

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

References

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

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

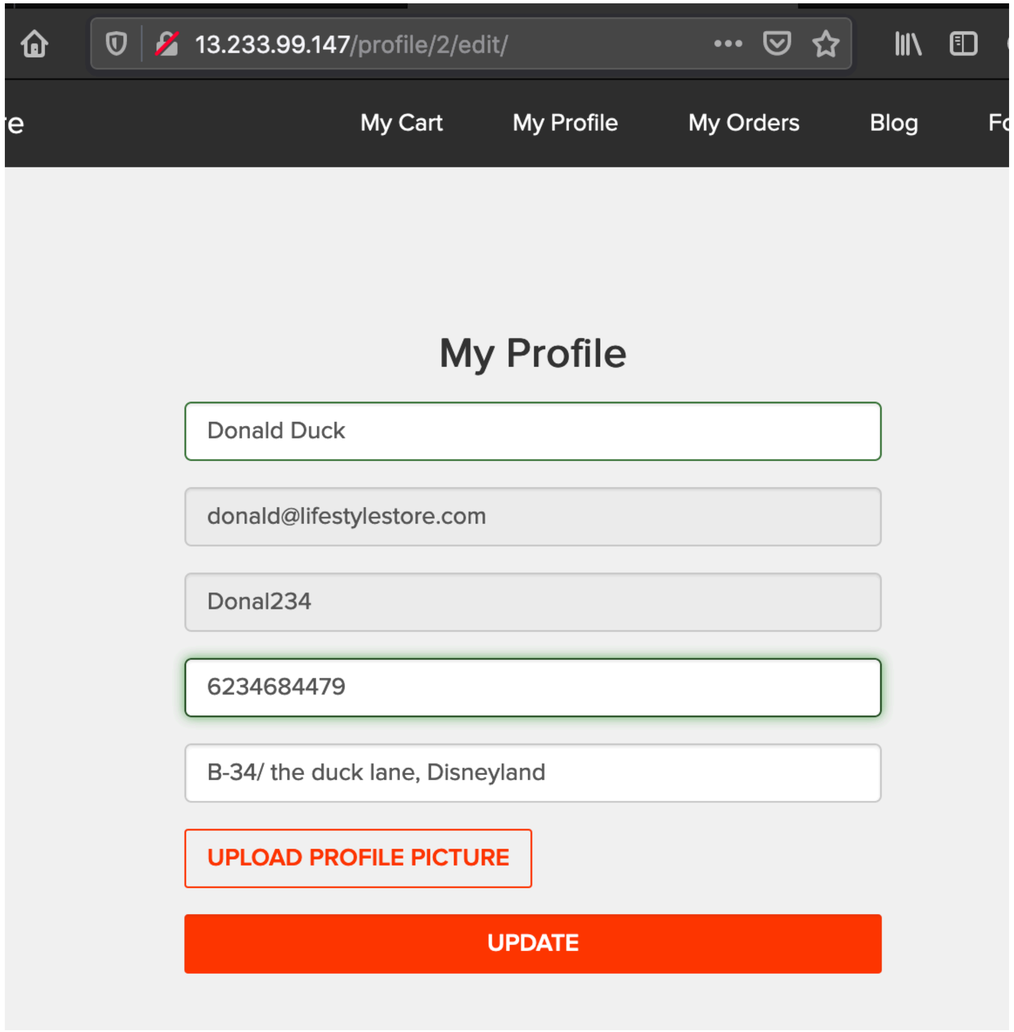
7. Cross Site Scripting

This happens when a user controlled input is reflected somewhere else in an HTML page and is not encoded/ sanitised properly. This leads to an attacker being able to inject HTML code in the affected page.

Cross Site Scripting (Severe)	Below mentioned parameters are vulnerable to reflected XSS **Affected URL :** • http://13.232.3.22/products/details.php?p_id=2 **Affected Parameters :** • POST button under Customer Review (POST parameters) **Payload:** • <script>alert(1)</script> **Affected URL :** • http://13.232.3.22/profile/2/edit/ **Affected Parameters :** • Address (POST parameters) **Payload:** • <script>alert(0)</script>

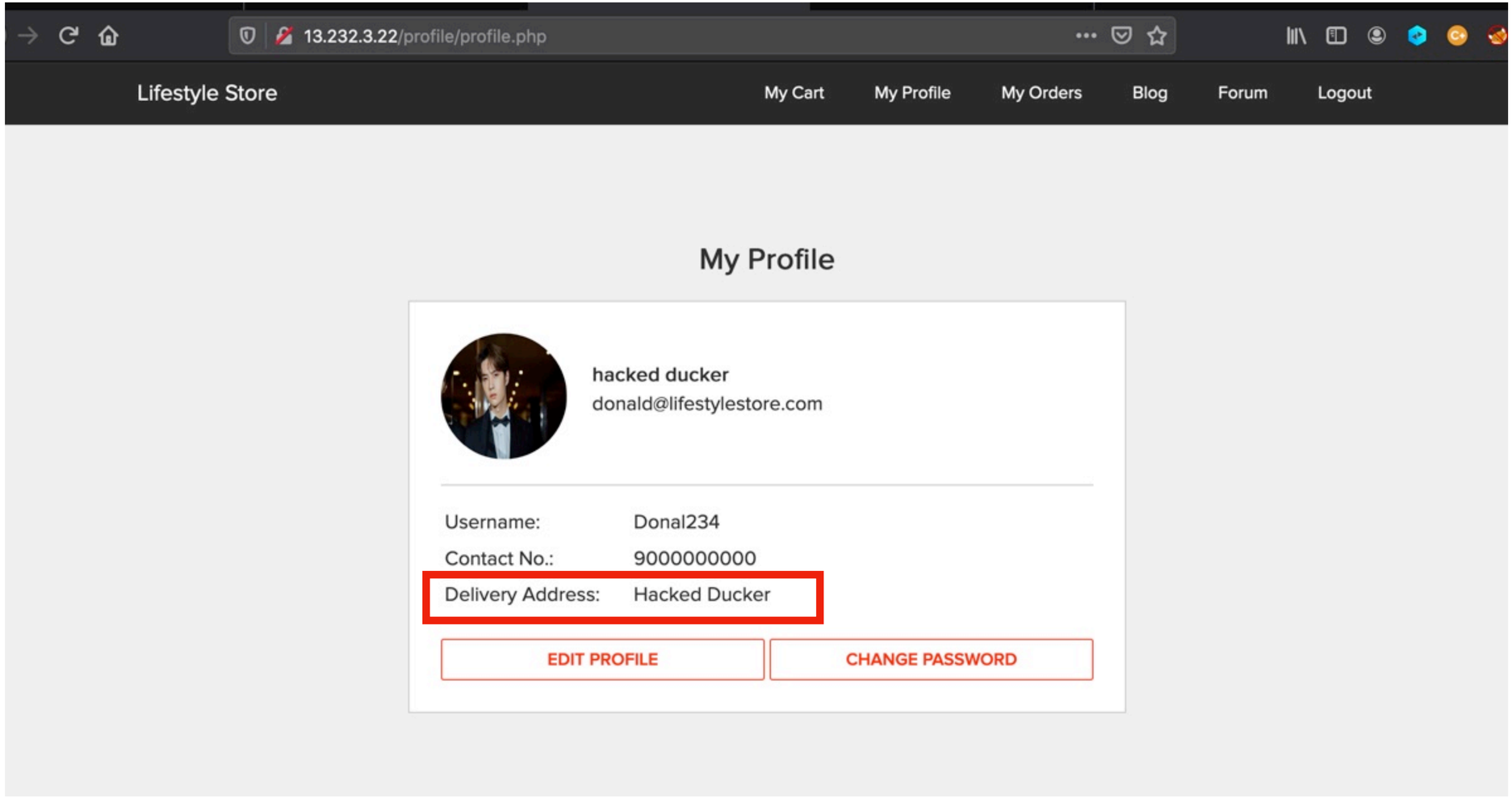
Observation

- Navigate to http://13.232.3.22/profile/2/edit/ .You will see user's details.



Observation

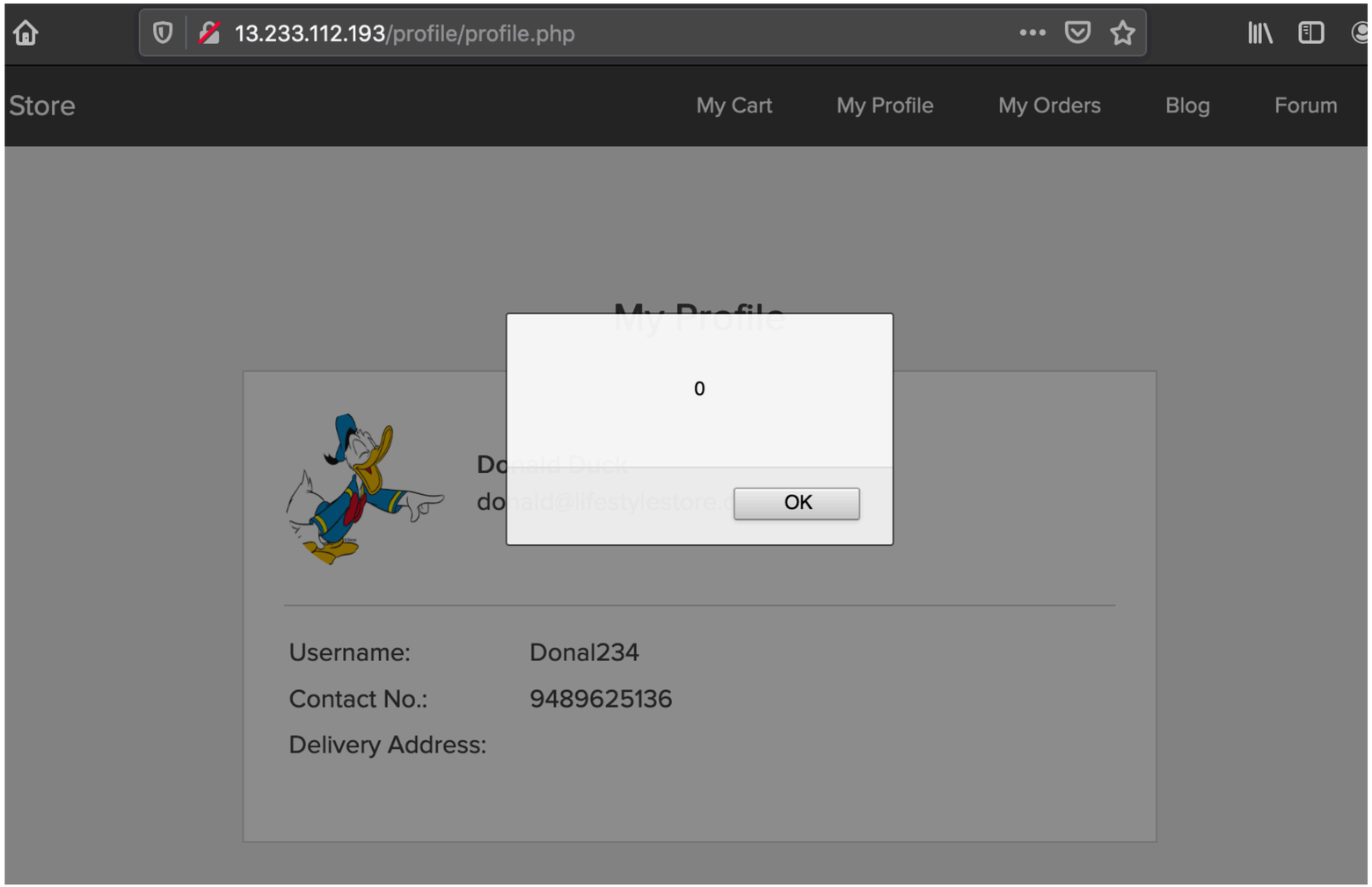
- Enter any text and click on **Update** , you will see it reflected in the next page and value will be in POST parameter in **Address** field.



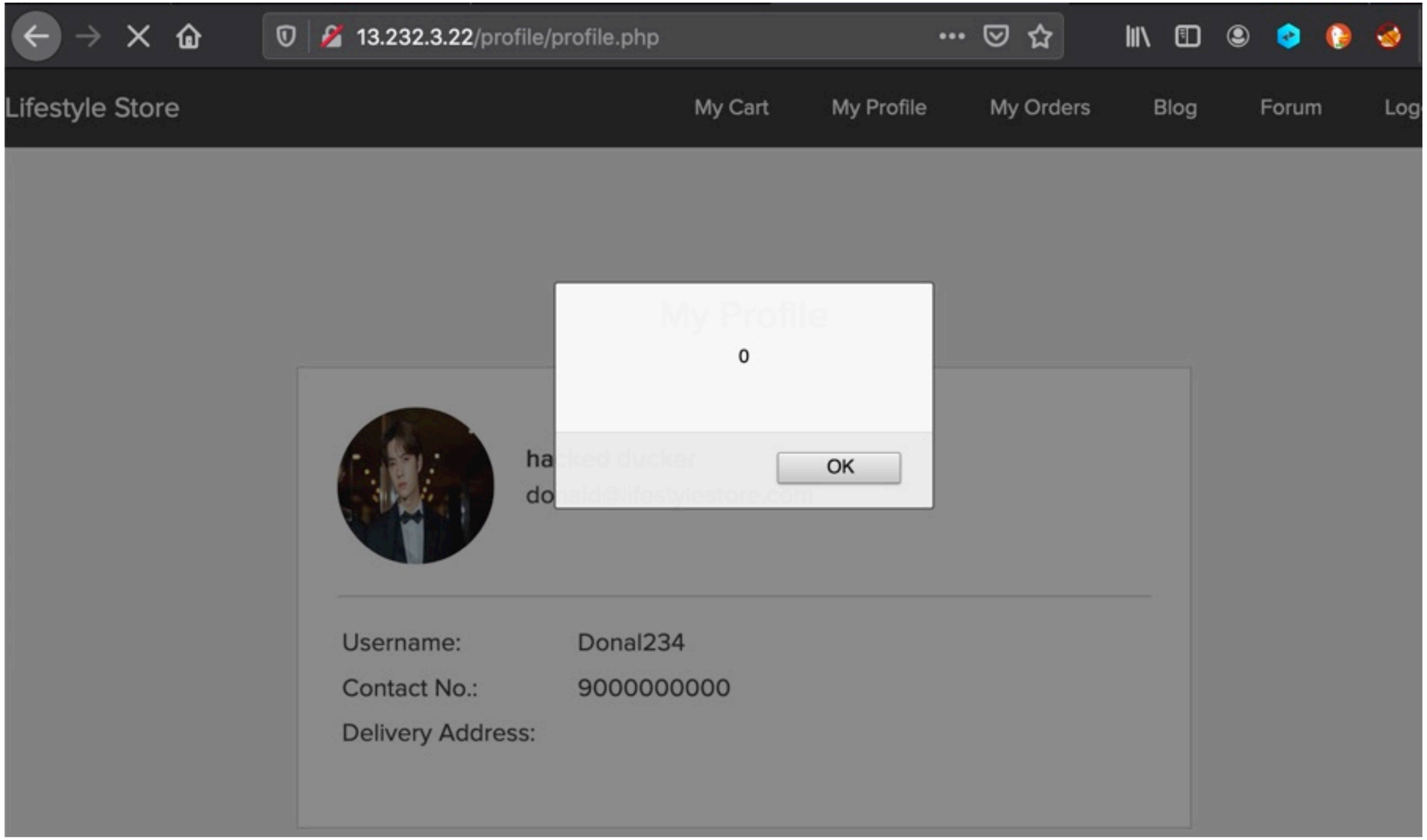
Observation

Put this payload instead of hacked ducker: **<script>alert(0)</script>**

As you can see we executed custom JS causing popup



Proof of Concept (PoC)



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 organisation

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.

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.
- Apply Client Side Filters to prevent client side filters bypass.

.

References

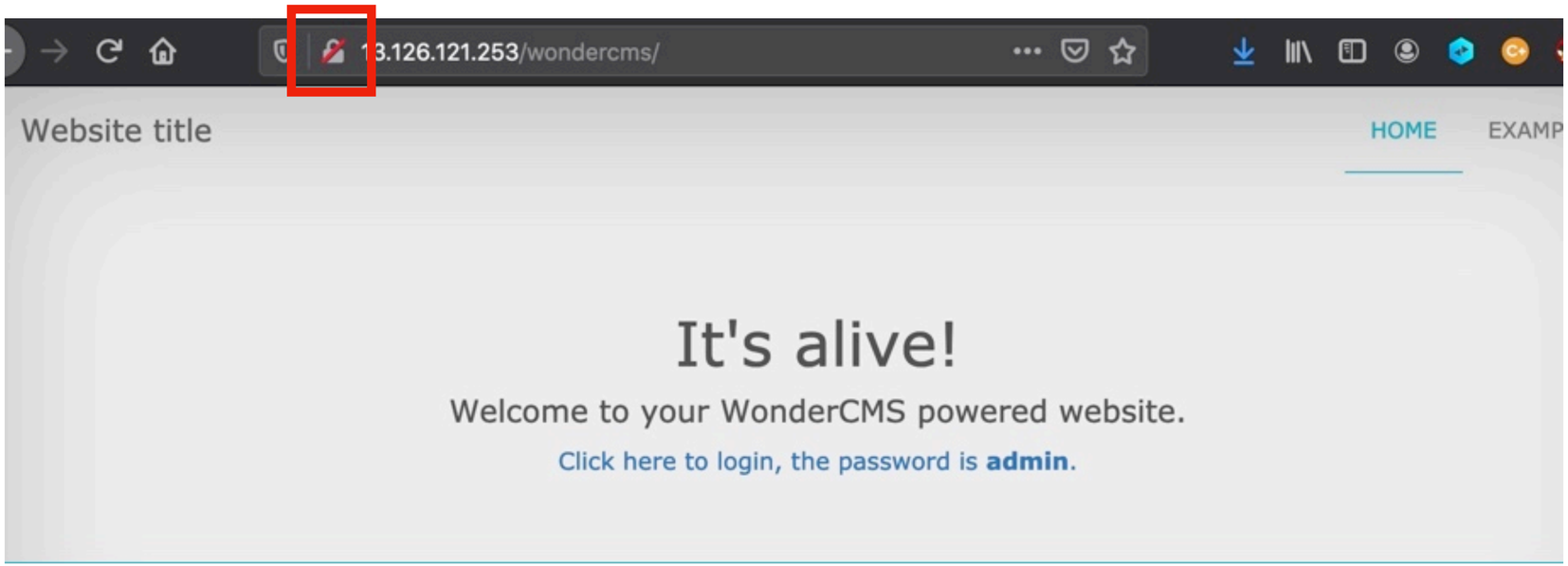
https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
https://en.wikipedia.org/wiki/Cross-site_scripting
https://www.w3schools.com/html/html_entities.asp

8.Crypto Configuration Flaws

Crypto Configuration Flaw((Severe)	Crypto Configuration Flaws are found in the modules below. **Affected URL :** • http://13.126.121.253/ (All the webpages ,blogs ,forum)

Observation

Clearly ,all the webpages use 'http' and not 'https' which is far less secure and not encrypted.



Business Impact - High

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

Recommendation

- Use https instead of http as the protocol.

References

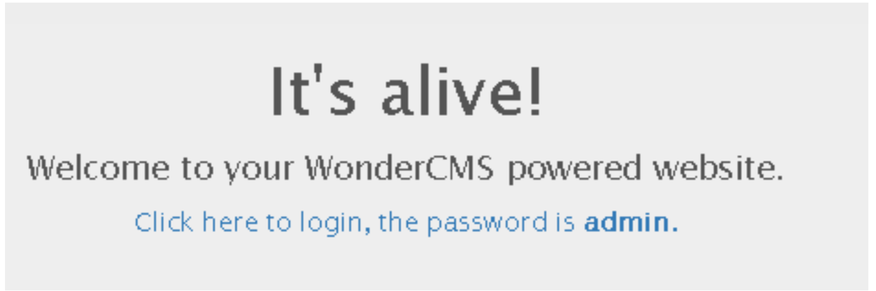
https://www.owasp.org/index.php/Category:Cryptographic_Vulnerability
https://www.w3.org/Protocols/rfc2616/rfc2616-sec15.html

9.Common Passwords

Weak Password Flaw (Severe)	Below given urls have weak passwords. **Affected URL :** • http://13.126.121.253/login/seller.php • http://52.66.198.61/wondercms/

Observation

The passwords of sellers and ,admin of blog ,is very common and easily predictable.



Business Impact - High

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

Recommendation

- There should be password strength check at every creation of an account.
- There must be a minimum of 8 characters long password with a mixture of numbers ,alphanumerics ,special characters ,etc.
- There should be no repetition of password ,neither on change nor reset.
- The password should not be stored on the web, rather should be hashed and stored.

References

https://www.acunetix.com/blog/articles/weak-password-vulnerability-common-think/

https://www.owasp.org/index.php/Testing_for_Weak_password_policy_(OTG-AUTHN-007)

10. Unauthorised availability of Details

Unauthorised Access to Customer Details (Severe)	The Show My Orders module is vulnerable from an Insecure Direct Object Reference (IDOR) that allows attacker see to anyones user details. **Affected URL :** • http://13.232.3.22/orders/orders.php?customer=2 **Affected Parameters :** • customer (GET parameters) **Payload used:** • http://13.232.3.22/orders/orders.php?customer=3

10. Unauthorised availability of Details

Unauthorised Access to Customer Details (Severe)	Similarly other vulnerable urls are given below. **Affected URL :** • http://13.232.3.22/orders/orders.php?customer=2 • http://13.232.3.22/orders/orders.php?customer=13 • http://13.232.3.22/orders/orders.php?customer=5 • http://13.232.3.22/orders/orders.php?customer=8 • http://13.232.3.22/orders/orders.php?customer=14

10. Unauthorised availability of Details

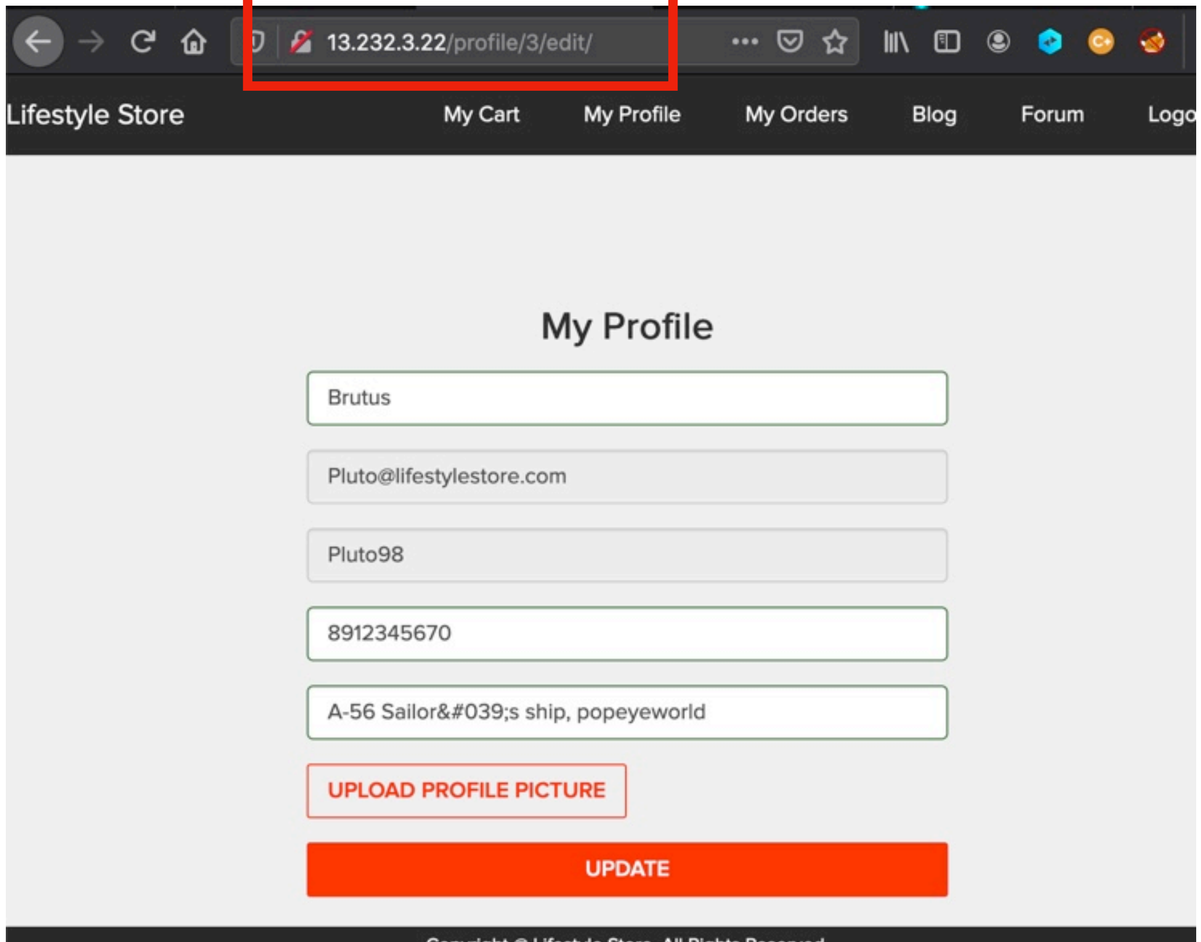
Unauthorised Access to Customer Details (Severe)	The Show My Orders module is vulnerable from an Insecure Direct Object Reference (IDOR) that allows attacker see to anyones Bill details **Affected URL :** • http://13.232.3.22/profile/2/edit/ **Affected Parameters :** • user_id (GET parameters) **Payload used:** •http://13.232.3.22/profile/3/edit/

Observation

- Login using any customer of the month’s details. Then navigate to the below link.

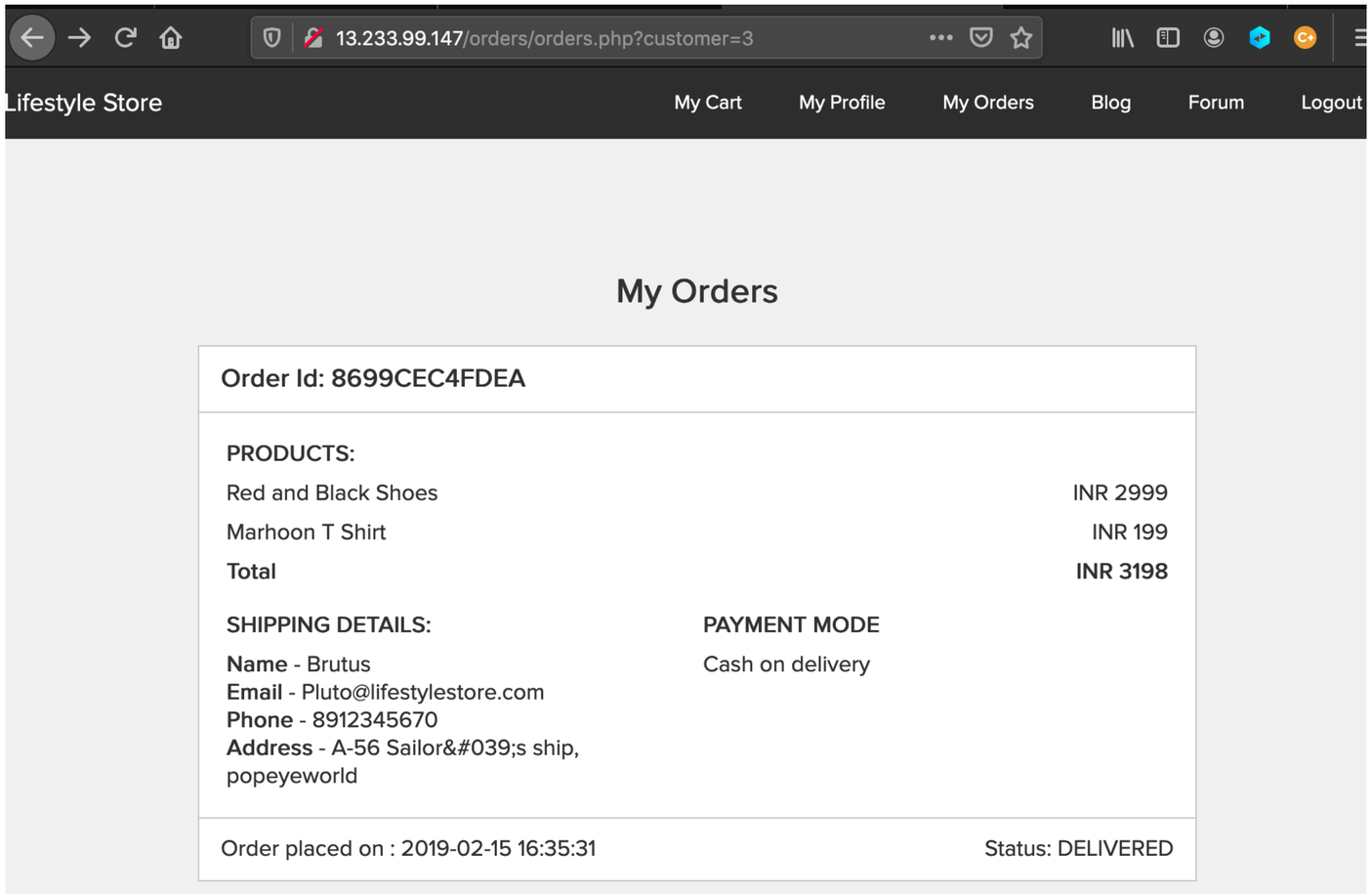
  http://13.232.3.22/profile/2/edit/

- Now remove **2** and insert **3** in the url like shown in the given screenshot and you will see the details of another user .



Proof of Concept (PoC)

- Below is the screenshot of the bill details of another user accessed from attacked user's account.



Business Impact – Extremely High

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

- Mobile Number
- Bill Number
- Billing Period
- Bill Amount and Breakdown
- Phone no. and email address
- Address

This can be used by malicious hackers to carry out targeted phishing attacks on the users and the information can also be sold to competitors/blackmarket.

More over, 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

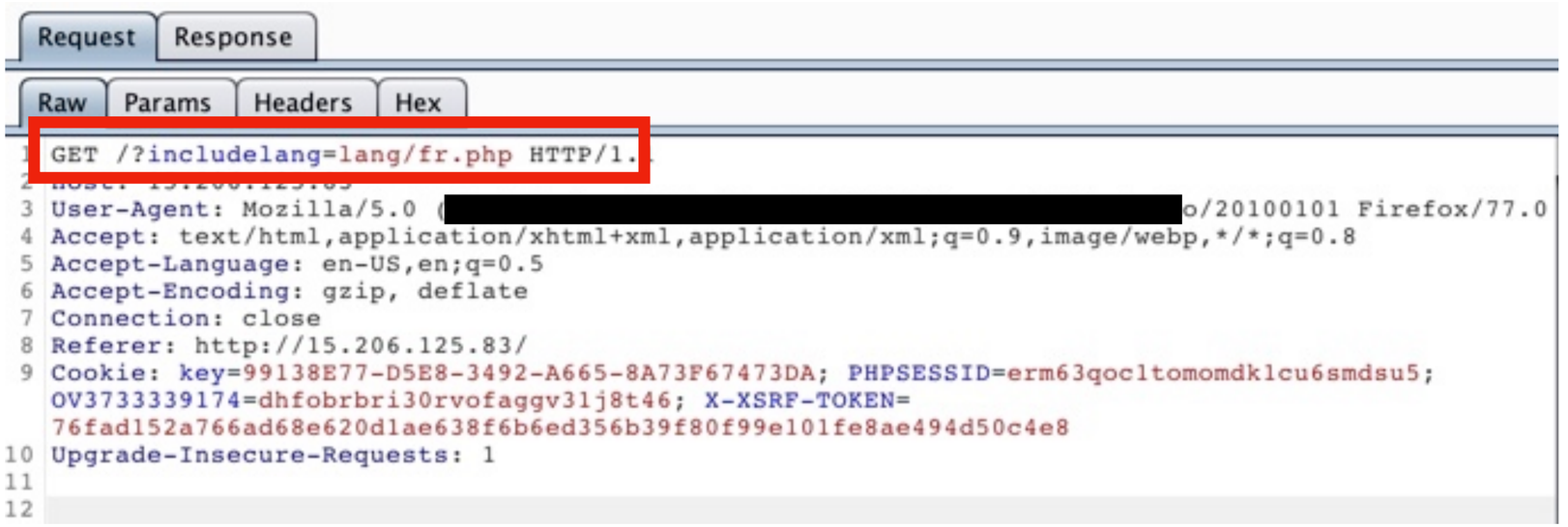
https://www.owasp.org/index.php/Insecure_Configuration_Management
https://www.owasp.org/index.php/Top_10_2013-A4-Insecure_Direct_Object_References

11.Open Redirection

Open Redirection (Severe)	The **Lang** module is vulnerable to open redirection. **Affected URL :** • http://15.206.125.83/?includelang=lang/en.php • http://15.206.125.83/?includelang=lang/fr.php **Affected Parameters :** • lang (GET parameters) **Payload used:** •http://15.206.125.83/?includelang=https://google.com/?lang/en.php **Other Affected URL :** • http://15.206.125.83/products/details.php?p_id=5

Observation

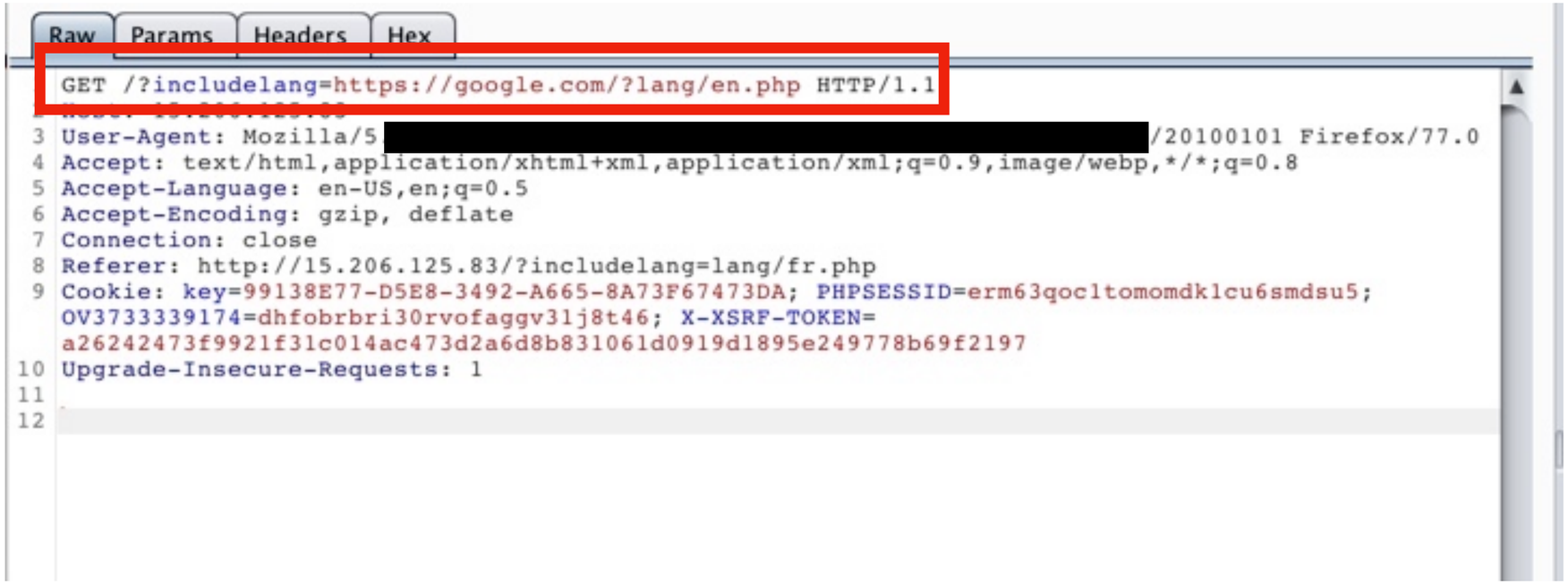
- Navigate to http://15.206.125.83/ and under the **Lang** tab click on **French**.
- Capture this request in local proxy .



Observation

- Now edit the request like this : GET /?includelang=**https://google.com/?**lang/en.php HTTP/1.1

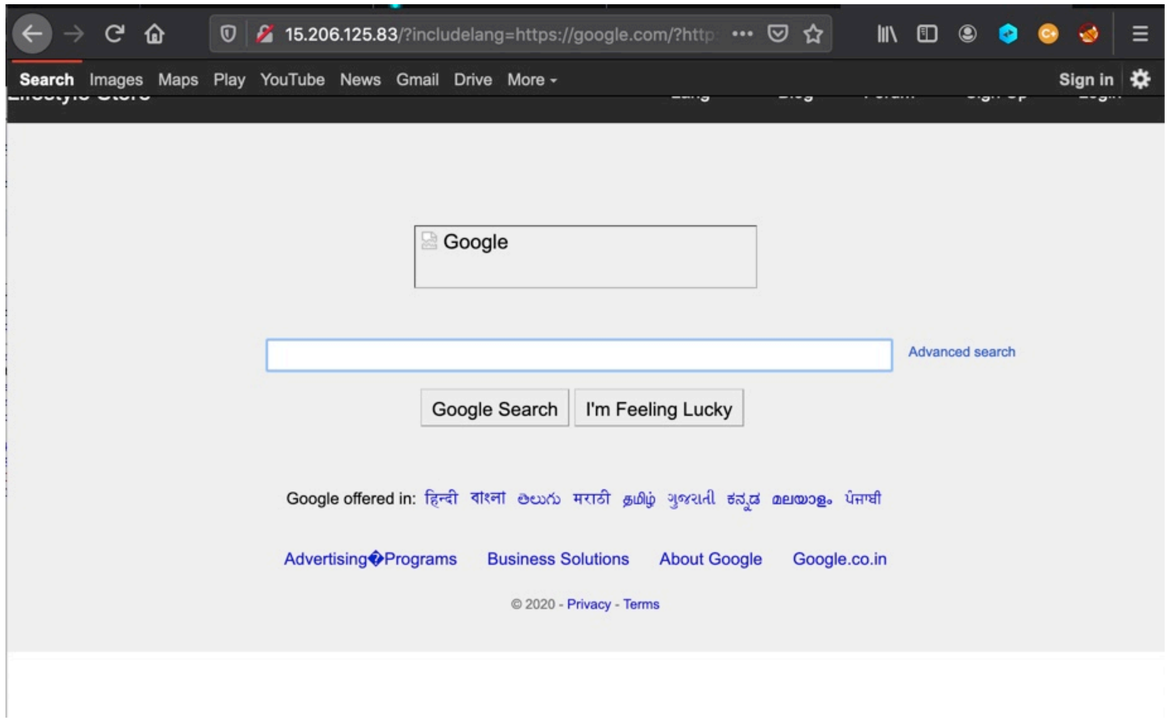
- Then pass this request in the browser. You will see the google.com .



Proof of Concept (PoC)

Raw Params Headers Hex

```
1  GET /?includelang=https://google.com/?lang/fr.php HTTP/1.1
2  Host: 15.206.125.83
3  User-Agent: [redacted].15;
   rv:77.0) Gec
4  Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,imag
   e/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Connection: close
8  Referer: http://15.206.125.83/
9  Cookie: key=99138E77-D5E8-3492-A665-8A73F67473DA;
   PHPSESSID=erm63qoc1tomomdk1cu6smdsu5; OV3733339174=
   dhfobrbri30rvofaggv31j8t46; X-XSRF-TOKEN=
   76fad152a766ad68e620d1ae638f6b6ed356b39f80f99e101fe8ae494d
   50c4e8
10 Upgrade-Insecure-Requests: 1
11
12
```



Business Impact – Extremely High

An http parameter may contain a URL value and could cause the web application to redirect the request to the specified URL. By modifying the URL value to a malicious site, an attacker may successfully launch a phishing scam and steal user credentials. Because the server name in the modified link is identical to the original site, phishing attempts have a more trustworthy appearance.

Recommendation

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

References

https://cwe.mitre.org/data/definitions/601.html

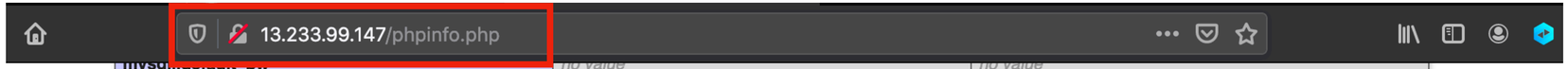
https://www.hacksplaining.com/prevention/open-redirects

12.Information disclosure due to Default Pages

Directory Listing (Moderate)	Below mentioned urls disclose server information. **Affected URL :** • http://13.233.99.47/phpinfo.php • https://15.206.125.83/robots.txt • http://35.154.142.220/composer.lock • http://35.154.142.220/composer.json • http://13.126.121.253/userlist.txt

Observation

- Navigate to http://13.233.99.47/phpinfo.php and you will see the below page.



mysqli.default_socket	*no value*	*no value*
mysqli.default_user	*no value*	*no value*
mysqli.max_links	Unlimited	Unlimited
mysqli.max_persistent	Unlimited	Unlimited
mysqli.reconnect	Off	Off
mysqli.rollback_on_cached_plink	Off	Off

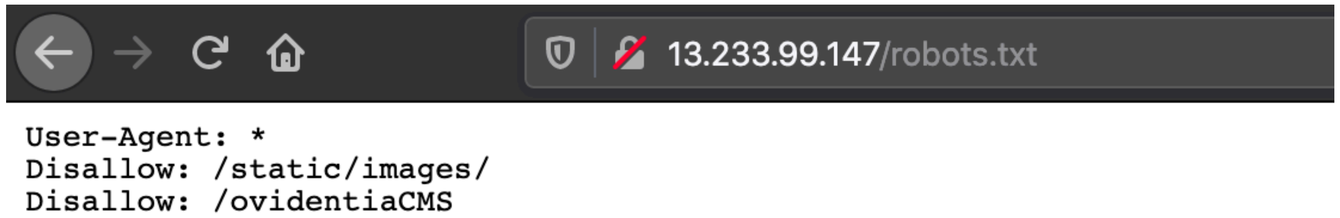
mysqlnd

mysqlnd	enabled
Version	mysqlnd 5.0.11-dev - 20120503 - $Id: 76b08b24596e12d4553bd41fc93cccd5bac2fe7a $
Compression	supported
core SSL	supported
extended SSL	supported
Command buffer size	4096
Read buffer size	32768
Read timeout	31536000
Collecting statistics	Yes
Collecting memory statistics	No
Tracing	n/a
Loaded plugins	mysqlnd,debug_trace,auth_plugin_mysql_native_password,auth_plugin_mysql_clear_password,auth_plugin_sha256_password
API Extensions	mysql,mysqli,pdo_mysql

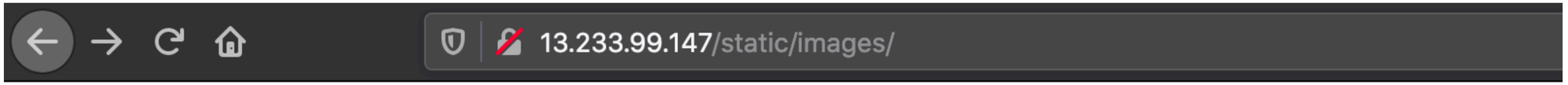
mysqlnd statistics	
bytes_sent	86853
bytes_received	317866
packets_sent	1646
packets_received	5888
protocol_overhead_in	23552
protocol_overhead_out	6584
bytes_received_ok_packet	0
bytes_received_eof_packet	0
bytes_received_rset_header_packet	3708
bytes_received_rset_field_meta_packet	0

Observation

- Navigate to https://15.206.125.83/robots.txt and you will se the following page.
- Next you can navigate to any of the listed files.



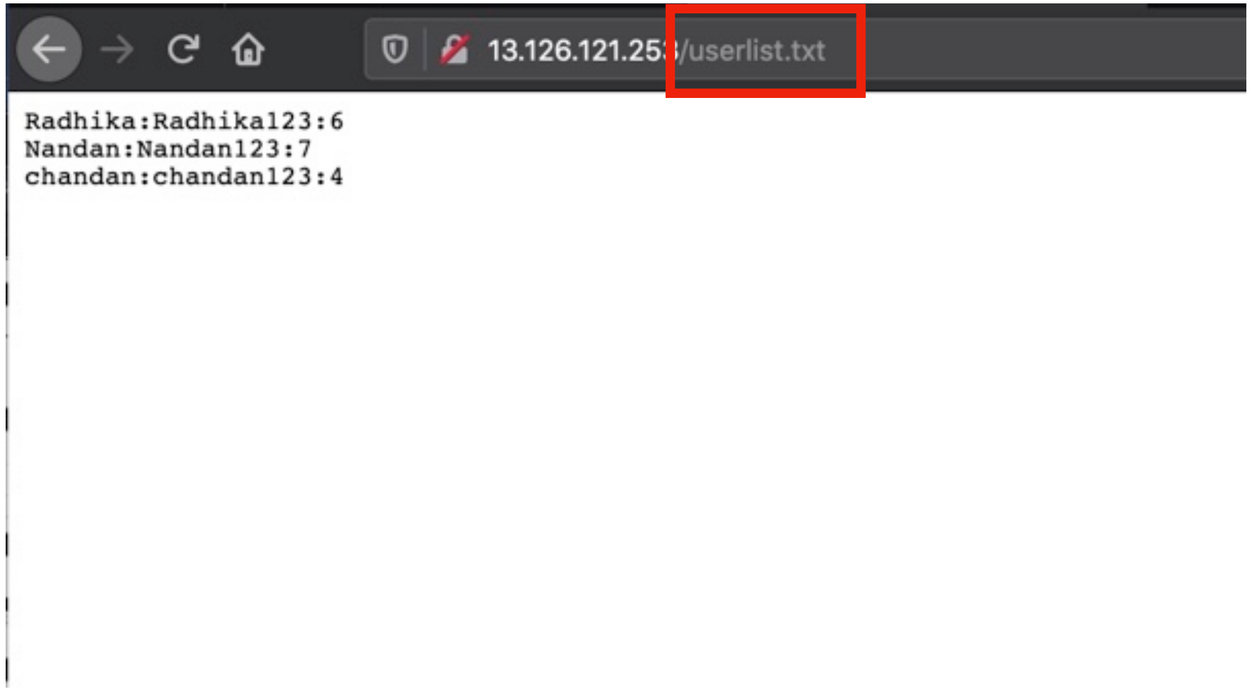
Proof of Concept (PoC)



Index of /static/images/

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

Proof of Concept (PoC)



Business Impact – Moderate

Although this vulnerability does not have a direct impact to users or the server, though it can aid the attacker with information about the server and the users.
Information Disclosure due to default pages are not exploitable in most cases, but are considered as web application security issues because they allows malicious hackers to gather relevant information which can be used later in the attack lifecycle, in order to achieve more than they could if they didn’t get access to such information.

Recommendation

Take the following precautions:

- Disable all default pages and folders including server-status and server-info.
- Multiple security checks enabled on important directories.

References

https://www.netsparker.com/blog/web-security/information-disclosure-issues-attacks/

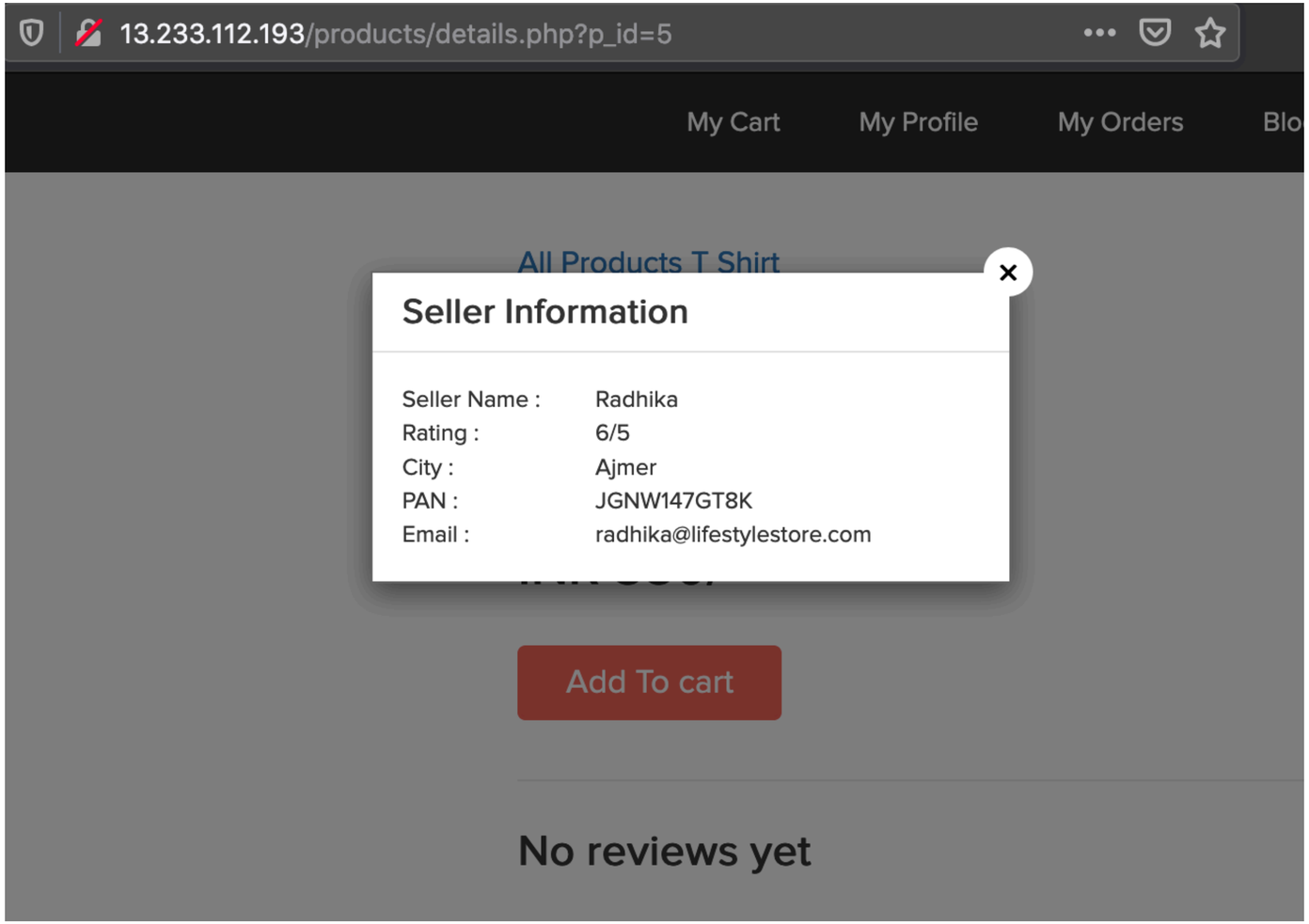
https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/information-disclosure-phpinfo/

13.Unnecessary Details about Sellers

Unnecessary Details about Sellers (Moderate)	Below mentioned URL gives the unnecessary details about the seller (PII). **Affected URL :** • http://13.126.121.253/products/details.php?p_id=2

Observation

- When we click on the **Seller Info** option ,we get the details of the seller ,even those which are not required like the pan card number ,etc.



Business Impact – Moderate

- There is no direct business impact in this case ,but this amount of information can definitely lead to social engineering attacks on the seller and can indirectly harm the business.
- The information could be sold to rival business companies .
- Sellers can be unnecessarily be pranked.

Recommendation

- Only name and email is sufficient as far as the query or help is concerned.

References

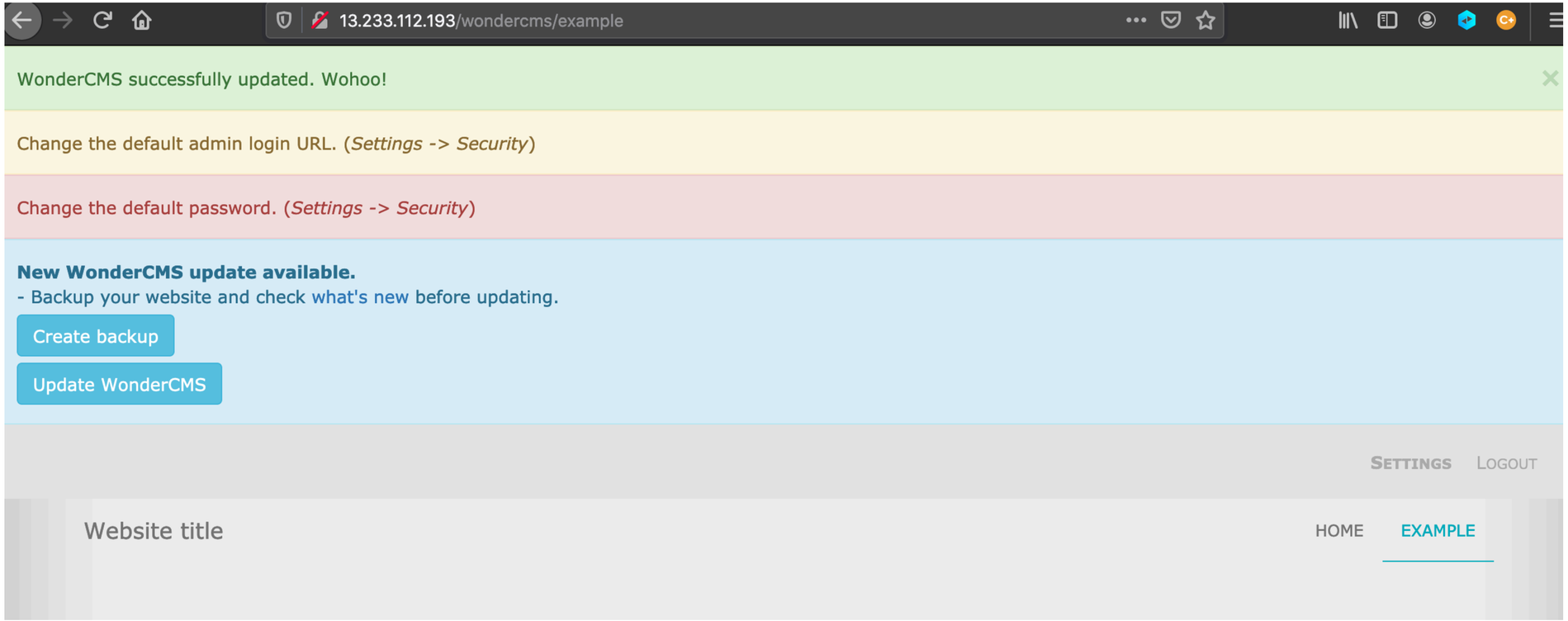
https://digitalguardian.com/blog/how-secure-personally-identifiable-information-against-loss-or-compromise

14.Components with known vulnerabilities

Components with known Vulnerabilities (Critical)	• Server used is nginx/1.14.0 appears to be outdated (current is at least 1.17.3) i.e it is known to have exploitable vulnerabilities. • WonderCMS

Observation

The PHP version installed is not the latest one and has multiple vulnerabilities that can be exploited.Also, wondercms is also outdated and highly vulnerable.



Business Impact – High

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

Recommendation

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

References

https://usn.ubuntu.com/4099-1/

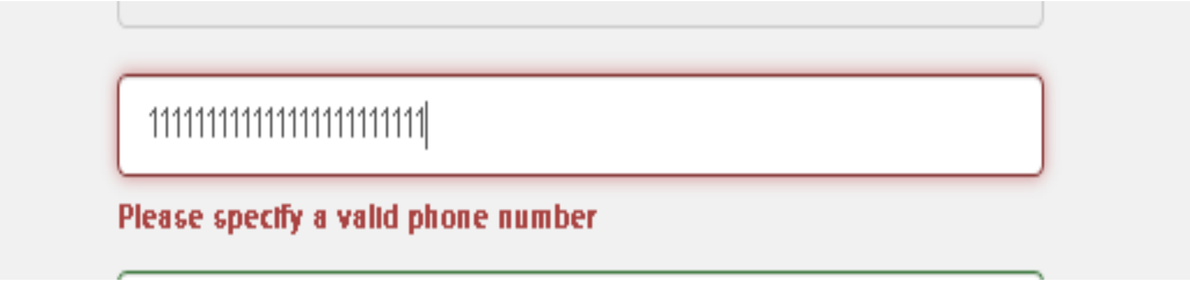
http://securitywarrior9.blogspot.com/2018/01/vulnerability-in-wonder-cms-leading-to.html

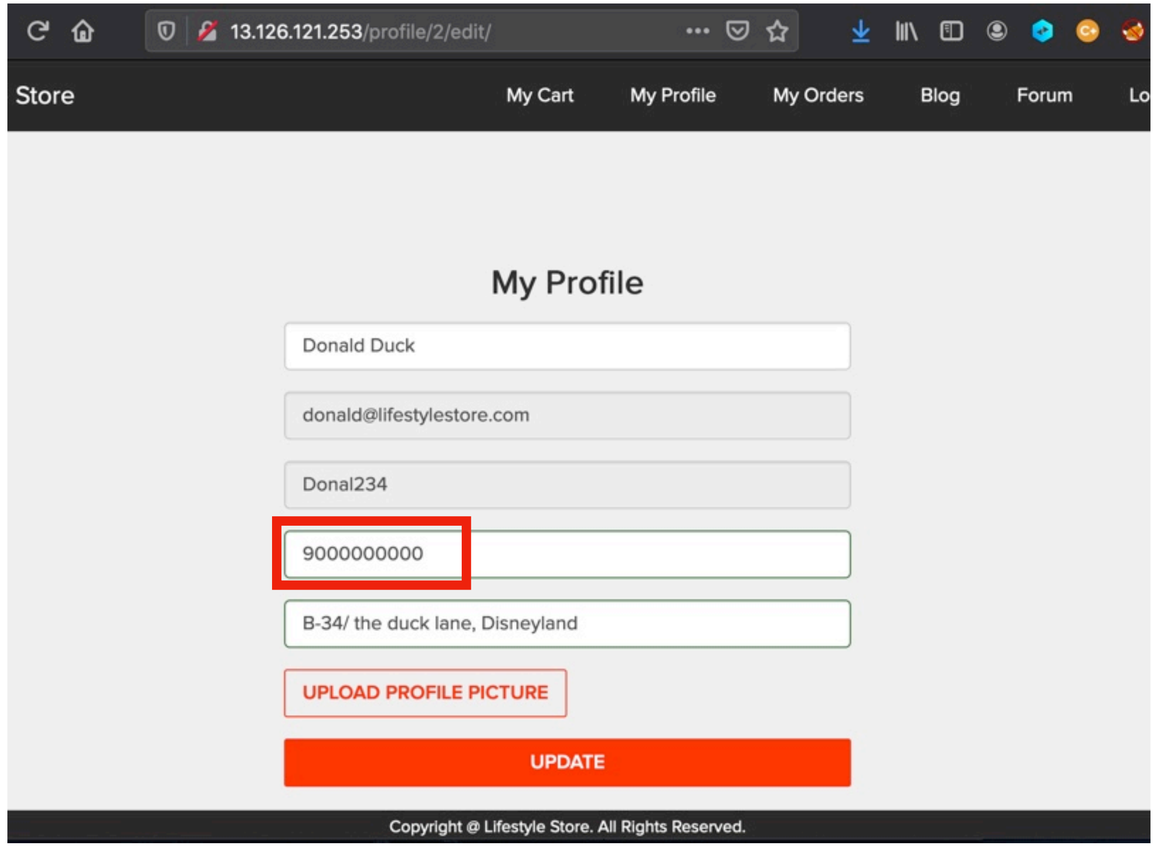
15.Improper Server Side and Client Side Filters

Improper Server Side and client side Filter (Low)	Below mentioned urls have improper server side filter **Affected URL :** • http://13.126.121.253/profile/16/edit/ **Affected parameter:** • Contact Number (POST Parameter) **Payload used:** • 9000000000 **Other Affected URL :** http://13.126.121.253/forum/index.php?u=/user/register

Observation

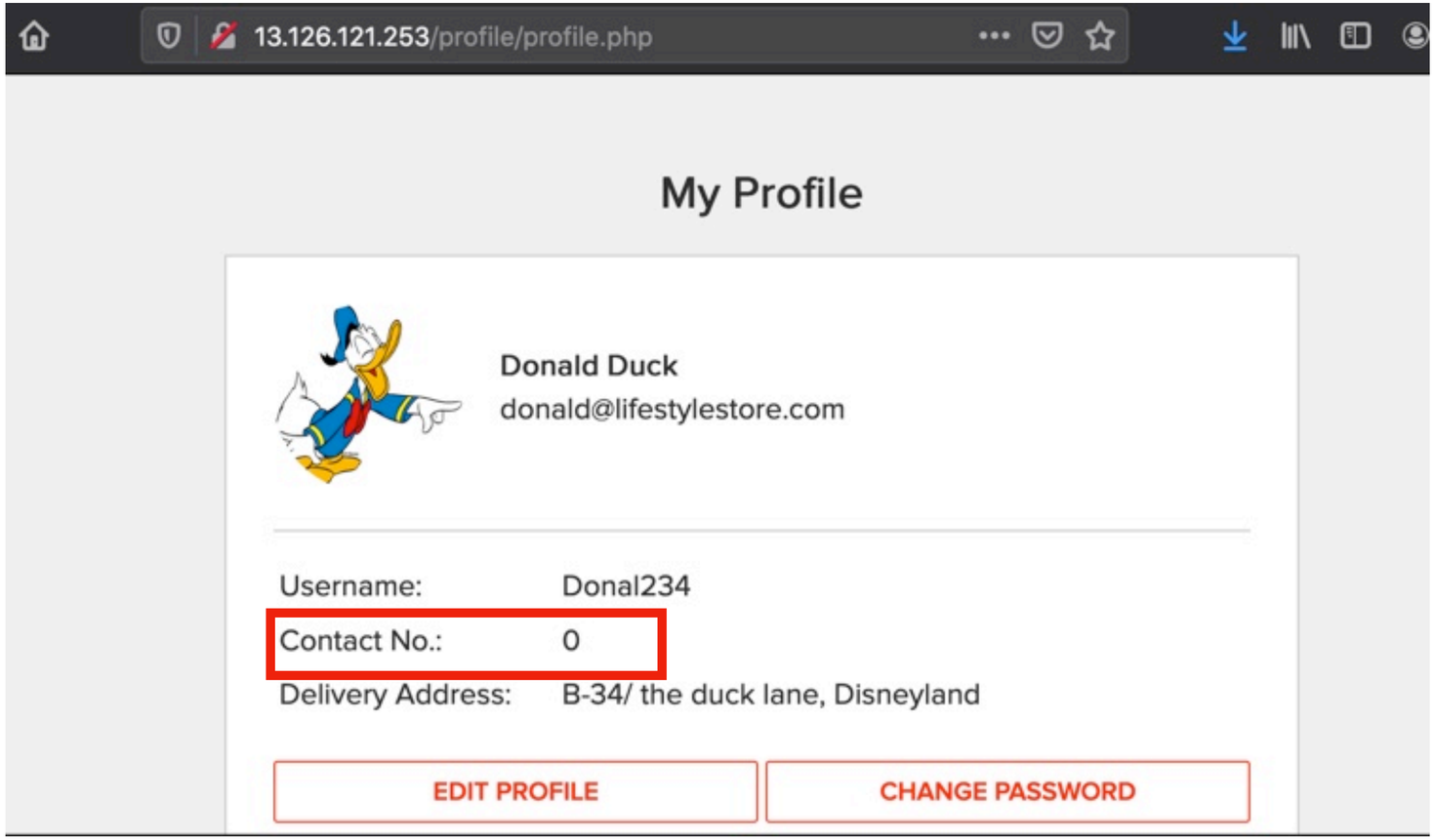
- After logging in as customer, when we try to edit the phone number to some invalid one , the error is as shown.
- Also if phone no. with correct length is entered but actually doesn’t exist, it is validated.





Proof of Concept (PoC)

- But when we give a valid phone number on the client side, but intercept it through burpsuite and again give invalid number ,it gets accepted.



Business Impact - Low

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

Recommendation

- Implement all critical checks on server side code only.
- Client-side checks must be treated as decoratives only.
- All business logic must be implemented and checked on the server code. This includes user input, the flow of applications and even the URL/Modules a user is supposed to access or not.

References

http://projects.webappsec.org/w/page/13246933/Improper%20Input%20Handling

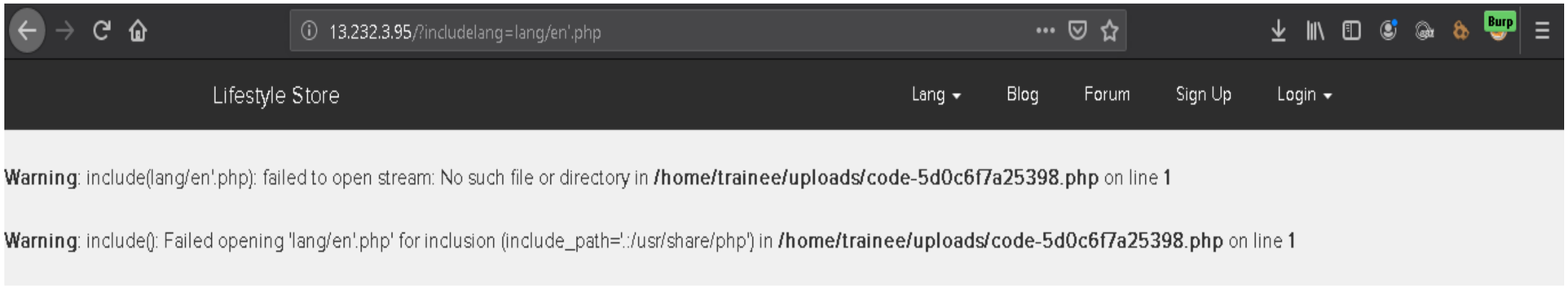
https://www.owasp.org/index.php/Unvalidated_Input

16.Default Error Display

Default Error Display (Low)	Below mentioned urls have default error displaying on fuzzing: **Affected URL :** • http://35.154.118.58/?includelang=lang/en.php **Payload** • en’.php (GET Parameter) **Affected URL:** • http://35.154.118.58/search/search.php **Parameter:** • q (GET Parameter) **Payload:** • q='

Observation

The default error with the path is displayed as:



Proof of Concept (PoC)

When we give **socks'** in the search option of the home page ,we get the error as:

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

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

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

References

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

THANK YOU

For any further clarifications/patch assistance, please contact:
92772xxxxx