🕵️ 🚾 Wordpress: Manual Enumeration 🫴

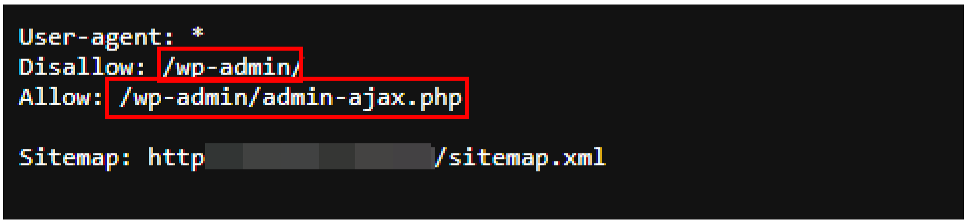
Intro

In this section, we will learn how to **manually enumerate a *WordPress*website**. Manual enumeration is a valuable skill for penetration testers and security researchers, as it allows for a deeper understanding of the target environment without relying solely on automated tools. In future sessions we will see the use of automated tools. But it is very important that we understand how to enumerate manually because depending on the scenario, we will use both automated tools and manual techniques.

🫴 Wordpress: Manual Enumeration

The very first think that we must to do is identity if our target machine is running a ***WordPress***, Manually we can do it in two ways

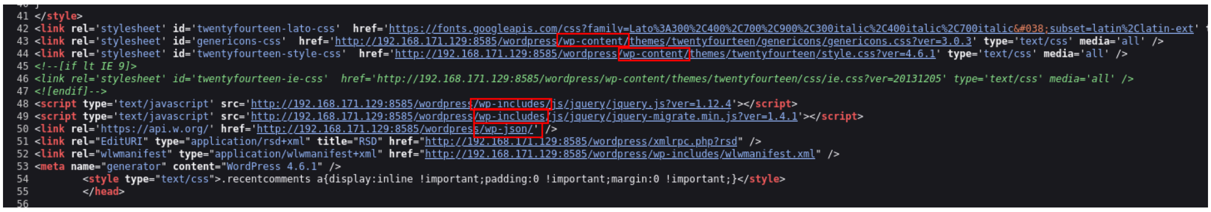
1. Going to the **robots.txt** directory



```
User-agent: *
Disallow: /wp-admin/
Allow: /wp-admin/admin-ajax.php

Sitemap: http[redacted]/sitemap.xml
```

2. **View the source code (Ctrl+u)**



```
</style>
<link rel='stylesheet' id='twentyfourteen-lato-css'  href='https://fonts.googleapis.com/css?family=Lato%3A300%2C400%2C700%2C900%2C300italic%2C400italic%2C700italic&#038;subset=latin%2Clatin-ext' t
<link rel='stylesheet' id='genericons-css'  href='http://192.168.171.129:8585/wordpress/wp-content/themes/twentyfourteen/genericons/genericons.css?ver=3.0.3' type='text/css' media='all' />
<link rel='stylesheet' id='twentyfourteen-style-css'  href='http://192.168.171.129:8585/wordpress/wp-content/themes/twentyfourteen/style.css?ver=4.6.1' type='text/css' media='all' />
<!--[if lt IE 9]>
<link rel='stylesheet' id='twentyfourteen-ie-css'  href='http://192.168.171.129:8585/wordpress/wp-content/themes/twentyfourteen/css/ie.css?ver=20131205' type='text/css' media='all' />
<![endif]-->
<script type='text/javascript' src='http://192.168.171.129:8585/wordpress/wp-includes/js/jquery/jquery.js?ver=1.12.4'></script>
<script type='text/javascript' src='http://192.168.171.129:8585/wordpress/wp-includes/js/jquery/jquery-migrate.min.js?ver=1.4.1'></script>
<link rel='https://api.w.org/' href='http://192.168.171.129:8585/wordpress/wp-json/' />
<link rel="EditURI" type="application/rsd+xml" title="RSD" href="http://192.168.171.129:8585/wordpress/xmlrpc.php?rsd" />
<link rel="wlwmanifest" type="application/wlwmanifest+xml" href="http://192.168.171.129:8585/wordpress/wp-includes/wlwmanifest.xml" />
<meta name="generator" content="WordPress 4.6.1" />
        <style type="text/css">.recentcomments a{display:inline !important;padding:0 !important;margin:0 !important;}</style>
        </head>

```

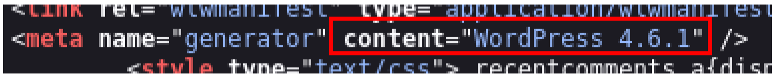
When doing this, whether viewing the **robots.txt** file or viewing the source code, we have to pay attention to the "keyword" which is ***wp-*** . If we find this, **Eureka!** The website that we are auditing is WordPress .

Once we have identified that our target website is a wordpress. What things are we going to look for when enumerating Wordpress? Among these things are: **1. Identifying the *WordPress* Version 2. User Enumeration, 3. Directory Browsing and File Exposure, 4. Plugin and Theme Detection 5. Login and Admin Pages and Other Public Files.**

And this is what we are going to do.

🕵️ Identifying the WordPress Version

To identify manually a *WordPress* version, we can go to the source page and find the line where is ***<meta name="generator"***

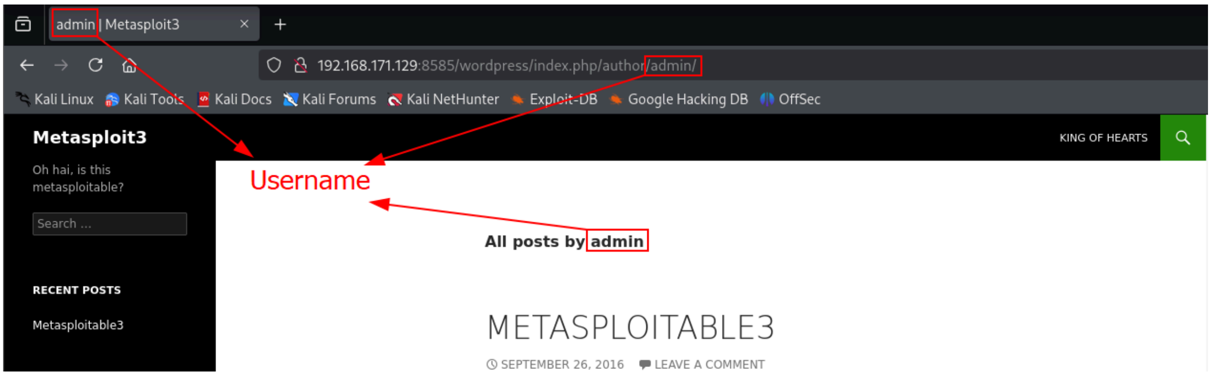


```
<link rel="wlwmanifest" type="application/wlwmanifest
<meta name="generator" content="WordPress 4.6.1" />
    <style type="text/css">.recentcomments a{disp
```

And look! With this information, we can now search for possible ***exploits*** for this version in case it's outdated.

👥 User Enumeration

To enumerate Users manually on a *WordPress* we just have to go to the **url** of our target website and add to **/?author=** aparat followed by the number 1 onwards and let see the result:



admin | Metasploit3

192.168.171.129:8585/wordpress/index.php/author/admin/

Metasploit3

Oh hai, is this metasploitable?

Search ...

RECENT POSTS

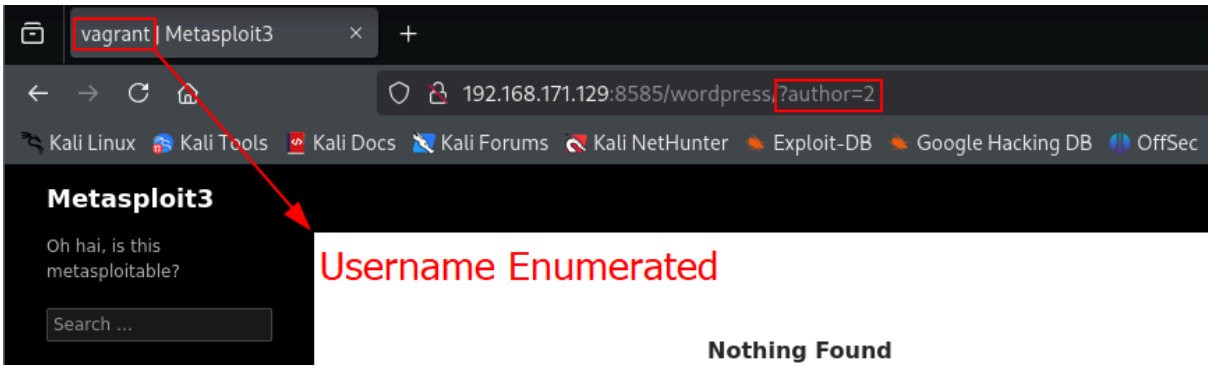
Metasploitable3

All posts by admin

METASPLOITABLE3

SEPTEMBER 26, 2016 LEAVE A COMMENT

We have successfully enumerated a user, concretely the admin user by adding **/?author=1** to the url . But, as we said, we can continue iterating over this attack Let’s do **/?author=2** to see what happens



vagrant | Metasploit3

192.168.171.129:8585/wordpress/?author=2

Metasploit3

Oh hai, is this metasploitable?

Search ...

Nothing Found

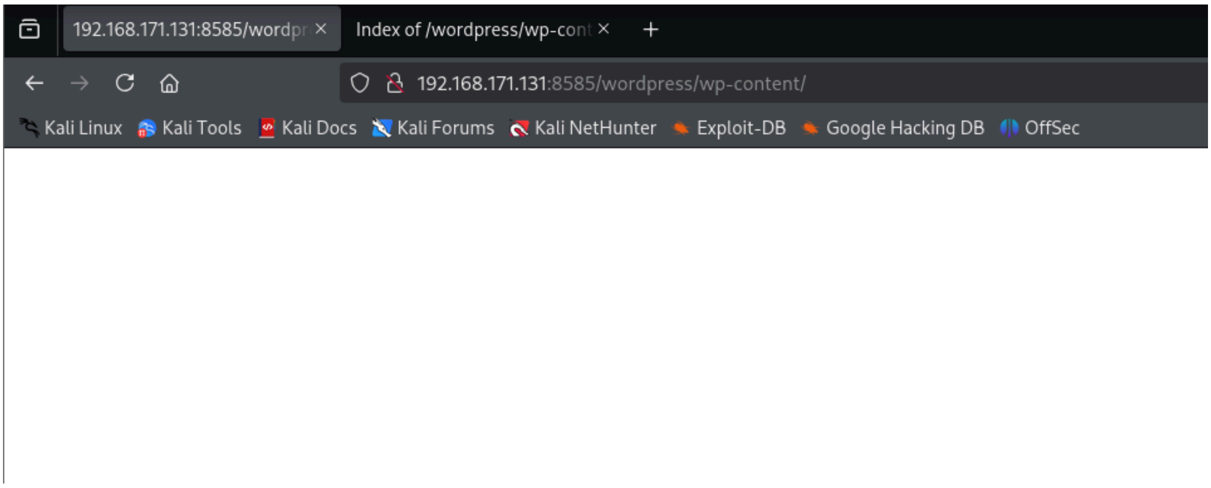
Look, we have enumerated other users just adding **/?author=1 and by moving on to the next number** until we have enumerated all users that have access to the wordpress login.

📂 Directory Browsing and File Exposure

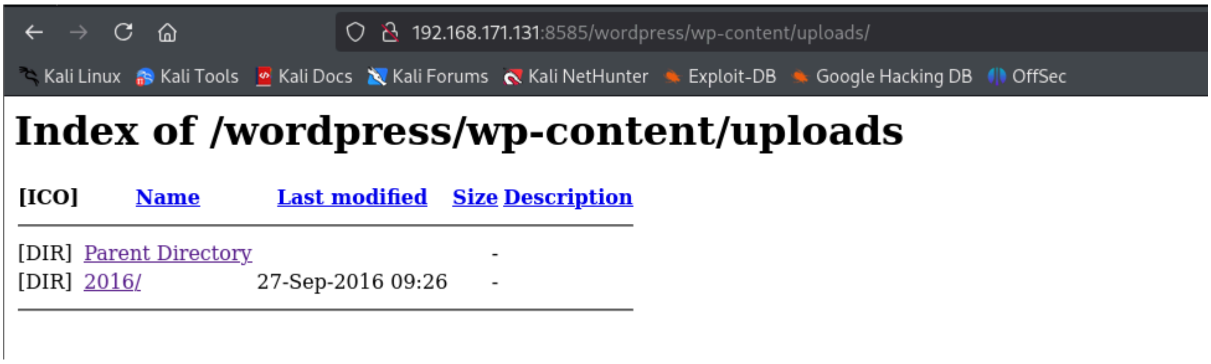
Something very important when enumerating a ***WordPress*** is to verify if directory listing is enabled for ***sensitive paths*** such as:

/wp-content/, /wp-content/plugins/, /wp-content/themes/ or **/wp-content/uploads**

And we realize that they are accessible when placing these directories the website goes blank:


192.168.171.131:8585/wordpr
Index of /wordpress/wp-cont
192.168.171.131:8585/wordpress/wp-content/
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec

If they are accessible, these directories **may expose files, plugin names, media uploads, or configuration remnants** that provide valuable insights. For example, if the directory **/uploads/** is accessible we enumerate those **expose files** and **media upload:**



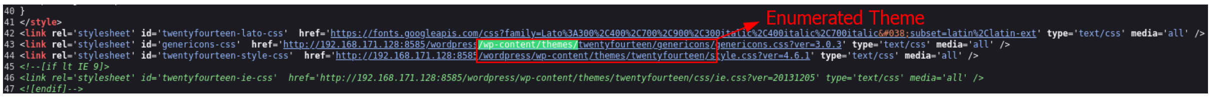
Index of /wordpress/wp-content/uploads

```
[ICO]  Name              Last modified      Size Description
[DIR]  Parent Directory                       -
[DIR]  2016/             27-Sep-2016 09:26    -
```

(For more information consult the section **2.5**)

🧩 Plugin and Theme Detection

We can often detect **active plugins** and themes by analyzing the HTML source or trying to directly access plugin/theme directories like **/wp-content/plugins/plugin-name/** or **/wp-content/themes/theme-name/**

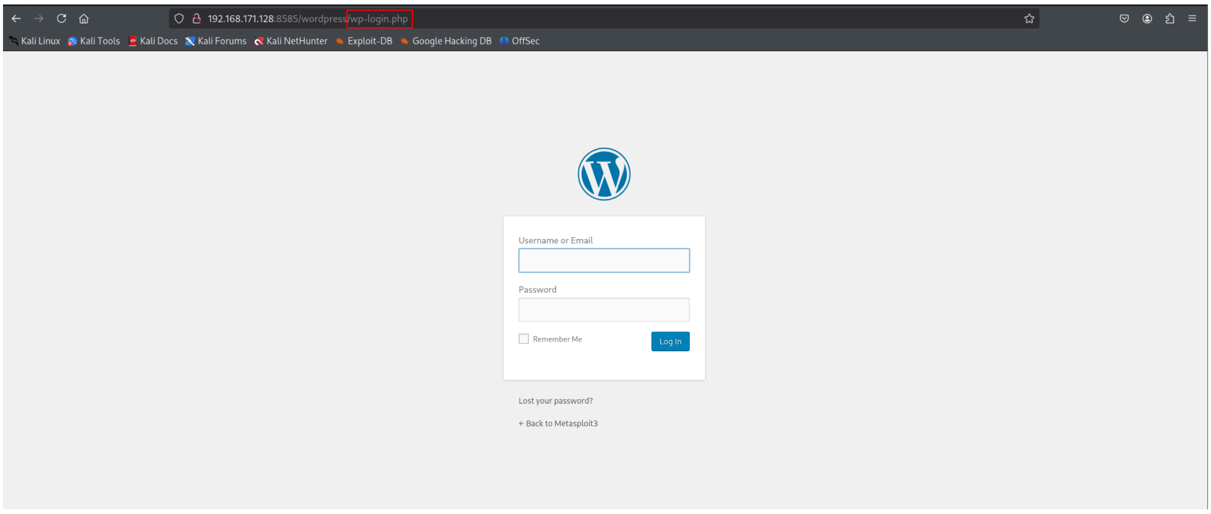


```
}
</style>
<link rel='stylesheet' id='twentyfourteen-lato-css'  href='https://fonts.googleapis.com/css?family=Lato%3A300%2C400%2C700%2C900%2C300italic%2C400italic%2C700italic&#038;subset=latin%2Clatin-ext' type='text/css' media='all' />
<link rel='stylesheet' id='genericons-css'  href='http://192.168.171.128:8585/wordpress/wp-content/themes/twentyfourteen/genericons/genericons.css?ver=3.0.3' type='text/css' media='all' />
<link rel='stylesheet' id='twentyfourteen-style-css'  href='http://192.168.171.128:8585/wordpress/wp-content/themes/twentyfourteen/style.css?ver=4.6.1' type='text/css' media='all' />
<!--[if lt IE 9]>
<link rel='stylesheet' id='twentyfourteen-ie-css'  href='http://192.168.171.128:8585/wordpress/wp-content/themes/twentyfourteen/css/ie.css?ver=20131205' type='text/css' media='all' />
<![endif]-->
```

Enumerated Theme

👤Login and Admin Pages

Standard login interfaces may be reachable at **/wp-login.php** or **/wp-admin/** panel



192.168.171.128:8585/wordpress/wp-login.php

These pages may offer fingerprinting opportunities or serve as brute-force targets (obviously only in authorized testing environments).

Conclusions

Look at everything we can do by manually enumerating a ***WordPress*** site to find bad configurations, usernames, exposed files, etc.

Once we understand this, let's move on to the next section where we'll look at the first **automated tool** for auditing ***WordPress*** websites.