# Potato:1

*Corso di Penetration Testing and Ethical Hacking*


DOCENTE
ARCANGELO CASTIGLIONE

CANDIDATO
Eduardo Autore
Matr. 052250/1549

# Scopo del Progetto

Gli obiettivi del progetto sono stati i seguenti:

- Effettuare il penetration testing su una macchina target vulnerabile by design, chiamata <<**potato:1**>>, reperita su Vulnhub.

- Produrre la documentazione necessaria, che comprende narrative e report sulle vulnerabilità trovate.

# Macchine e strumenti utilizzati


MACCHINA ATTACCANTE

VIRTUALIZZATORE
RETE CORSO: 10.0.2.0/24


ASSET VULNERABILE
(**Potato**)





# Metodologia

Le tipiche fasi di un penetration testing, come stabilito dal Framework Generale per il Penetration Testing (FGPT), che sono state eseguite, sono nel seguente ordine:

1) Information Gathering
2) Target Discovery
3) Enumerating Target & Port Scanning
4) Vulnerability Mapping
5) Target Exploitation
6) Post-Exploitation

# 1. Information Gathering

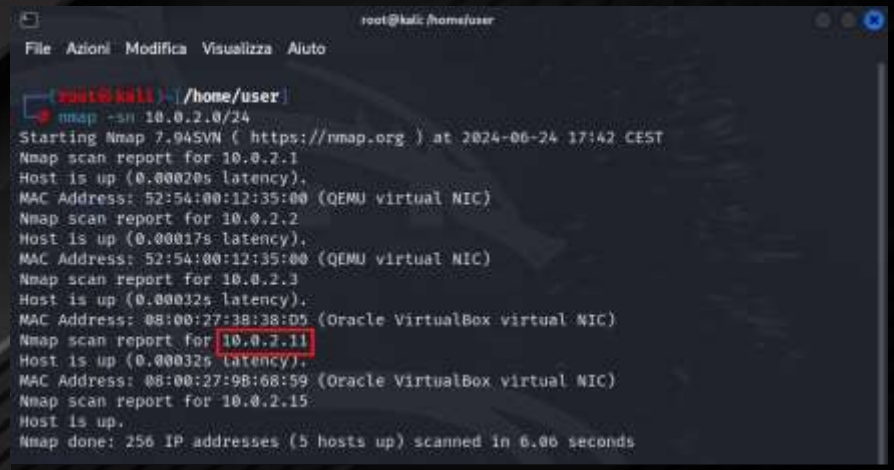
Fase volta a raccogliere il maggior numero possibile di informazioni sull'asset da attaccare.

- Alcune di queste informazioni sono disponibili su VulnHub.
- Altre visualizzando la schermata della macchina da virtualbox come la **versione di Ubuntu.**


# 2. Target Discovery

Fase in cui l'obiettivo è raccogliere dati dettagliati sulle macchine target del penetration testing.

- Utilizzo di varie opzioni di scansione fornite da Nmap.




IDENTIFICAZIONE HOST SULLA RETE


OS FINGERPRINT ATTIVO


VERIFICA DISPONIBILITA' HOST

# 3. Enumerating Target e Port Scanning

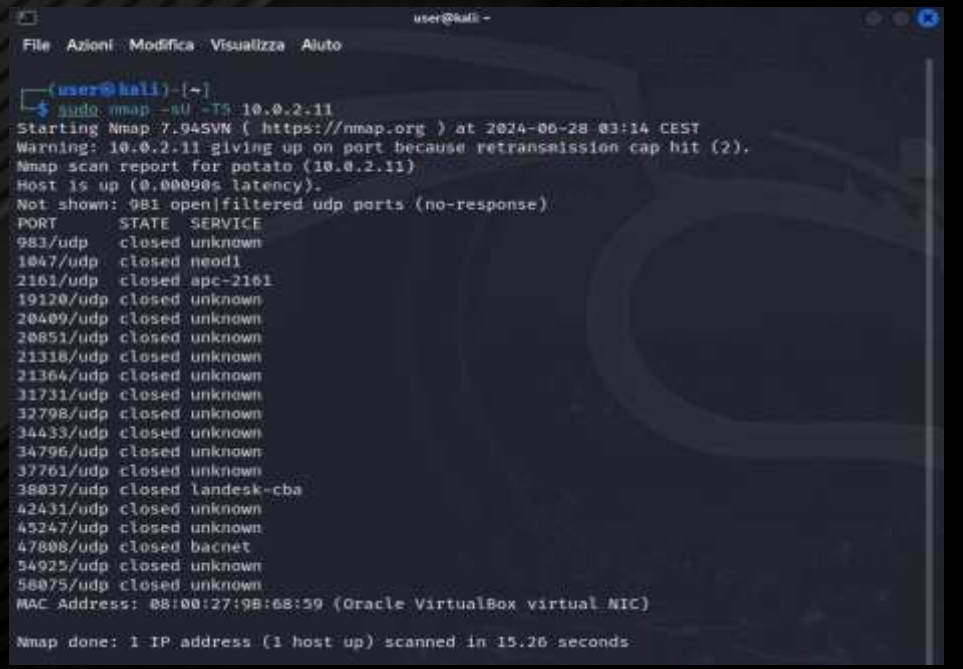
## INTRODUZIONE E SCANSIONI NMAP

Durante questa fase, vengono raccolte informazioni dettagliate e cruciali, tra cui:

- Stato di apertura delle porte
- Tipologia e natura dei servizi attivi
- Versioni specifiche dei servizi in esecuzione
- Sistemi operativi identificati
- Eventuali configurazioni di rete rilevanti

Principali strumenti utilizzati: **Nmap** e **Unicornscan**.


NMAP TCP (-Sv)



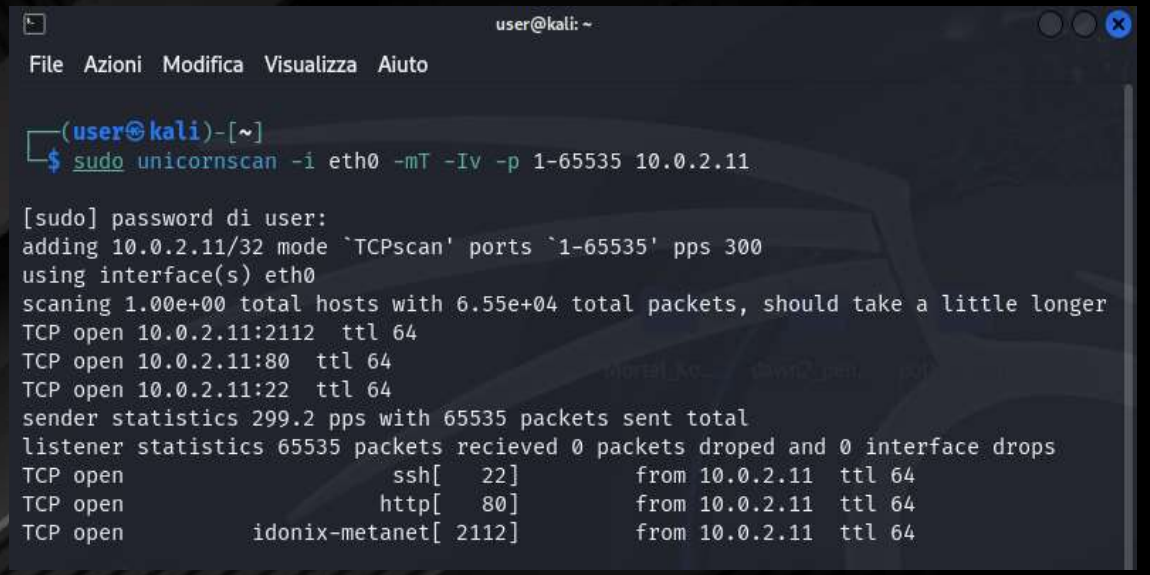

NMAP UDP (-sU)

# 3. Enumerating Target e Port Scanning

## SCANSIONI UNICORNSCAN

- Ulteriori scansioni sono state effettuate con strumenti come **Unicornscan** (TCP e UDP) per altri riscontri.

- Non sono stati ottenuti ulteriori risultati di rilievo rispetto alle precedenti scansioni.




UNICORN SCAN TCP

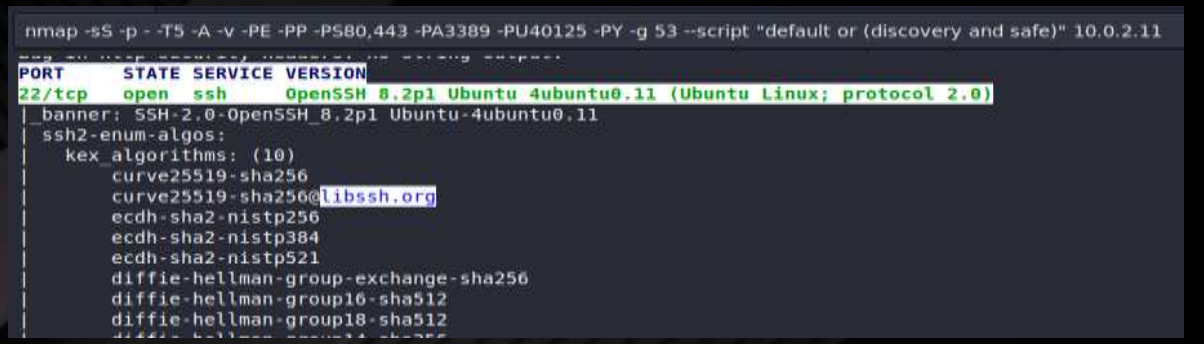

UNICORN SCAN UDP

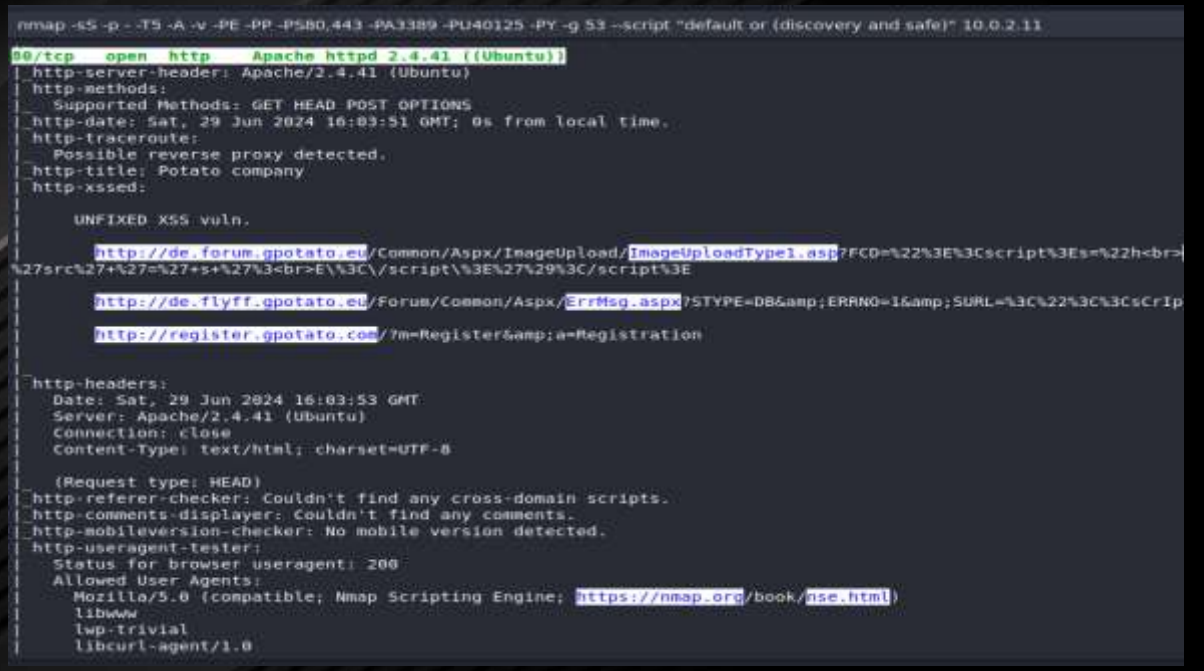
# 3. Enumerating Target e Port Scanning

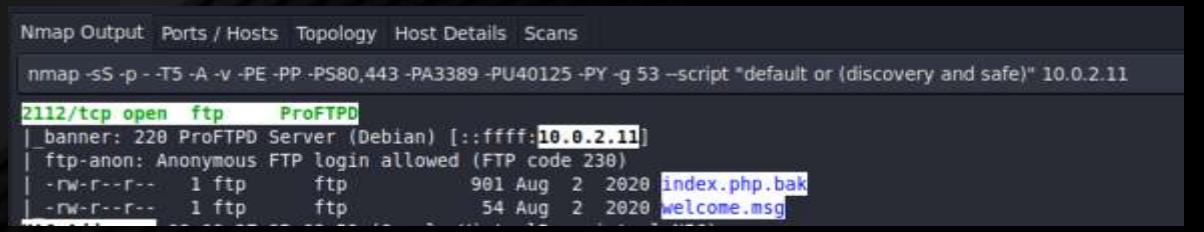
## SCANSIONI AVANZATE

- Altre scansioni avanzate sono state effettuate con **Nmap** e **Zenmap** per cercare  di ottenere risultati più accurati.

- Ulteriori informazioni su protocolli crittografici di OpenSSH, metodi supportati dal servizio Apache e informazioni preliminari su <<**FTP anonymous Login**>>.



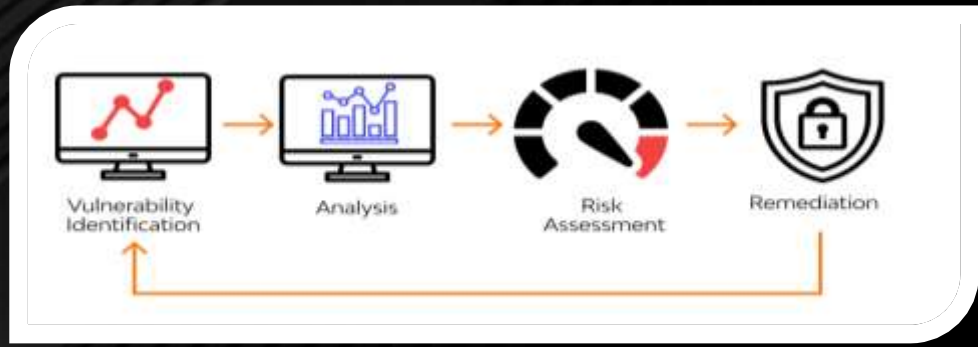



# 4. Vulnerability mapping

Le vulnerabilità presenti nella macchina vengono identificate e analizzate con precisione. Sono state adottate due tecniche principali per questa attività:

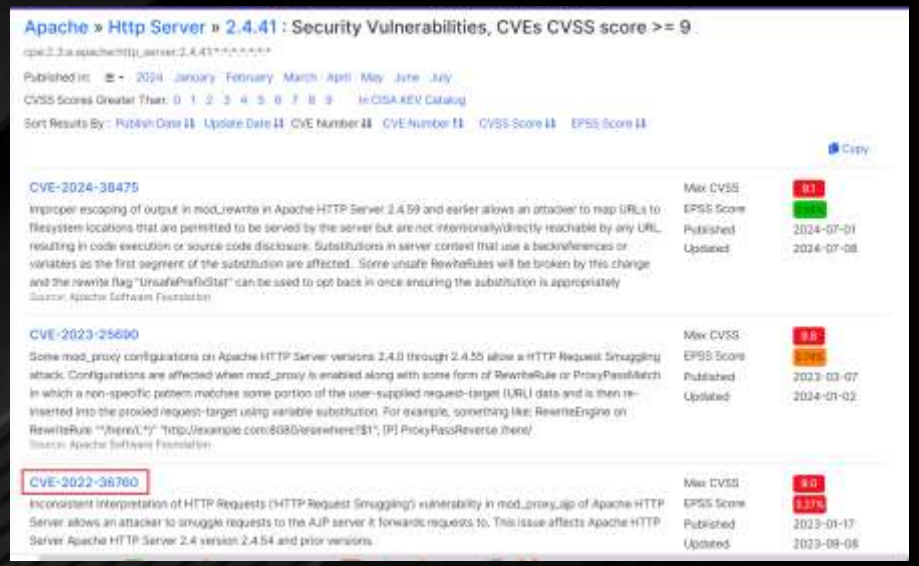
1. **Tecniche Manuali**
   - Utilizzo di database di vulnerabilità come **CVE Details** ed **Exploit-DB** per la ricerca di exploit specifici e dettagliati.

2. **Tecniche Automatiche**
   - Utilizzo di strumenti di scansione automatizzata come **Nessus**, **Nmap**, **OpenVAS** e **Nikto** per rilevare e valutare le vulnerabilità in modo sistematico e approfondito.



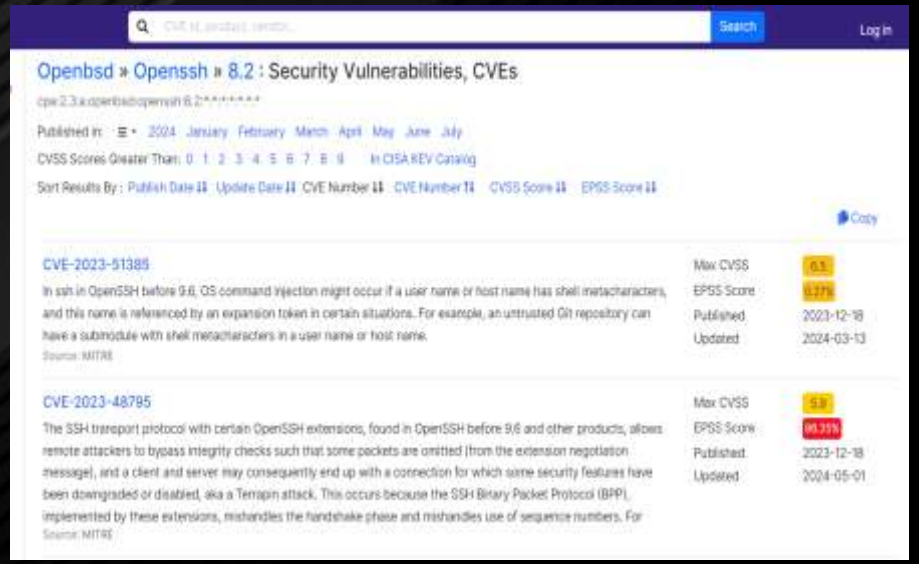
# 4. Vulnerability mapping

## TECNICHE MANUALI




APACHE 2.4.41 - CVE-DETAILS


APACHE 2.4.x - EXPLOIT-DB

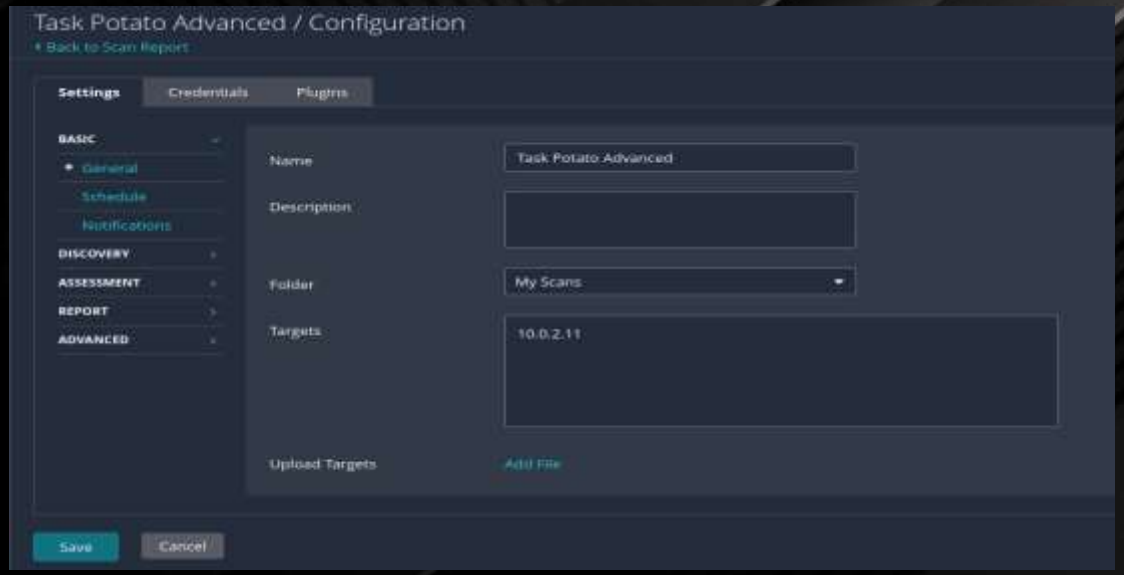



OPENSSH 8.2 - CVE-DETAILS

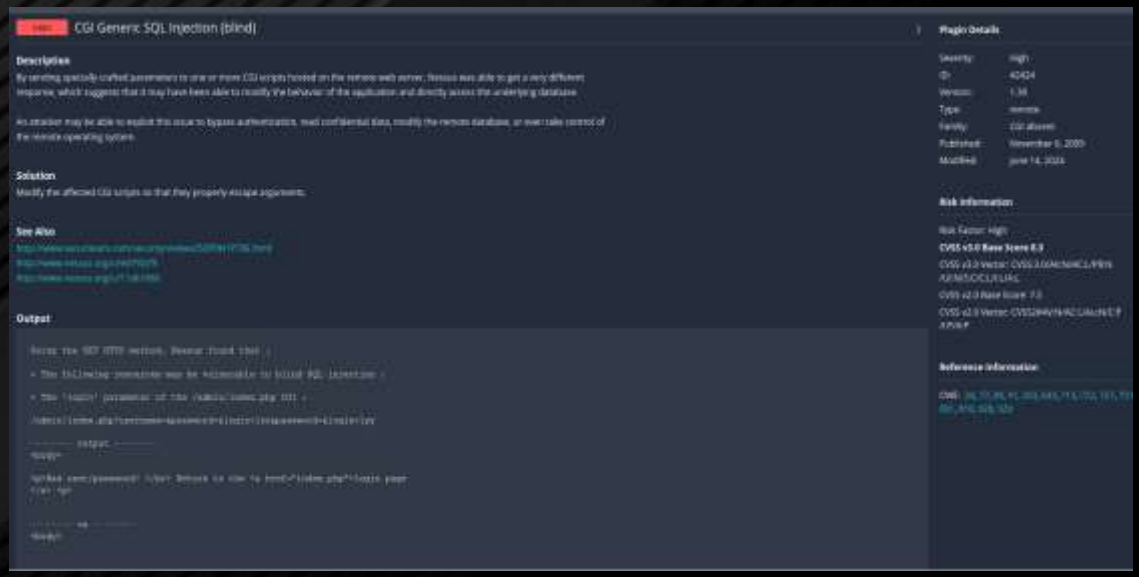
# 4. Vulnerability mapping

## TECNICHE AUTOMATICHE (NESSUS)


SCAN NESSUS



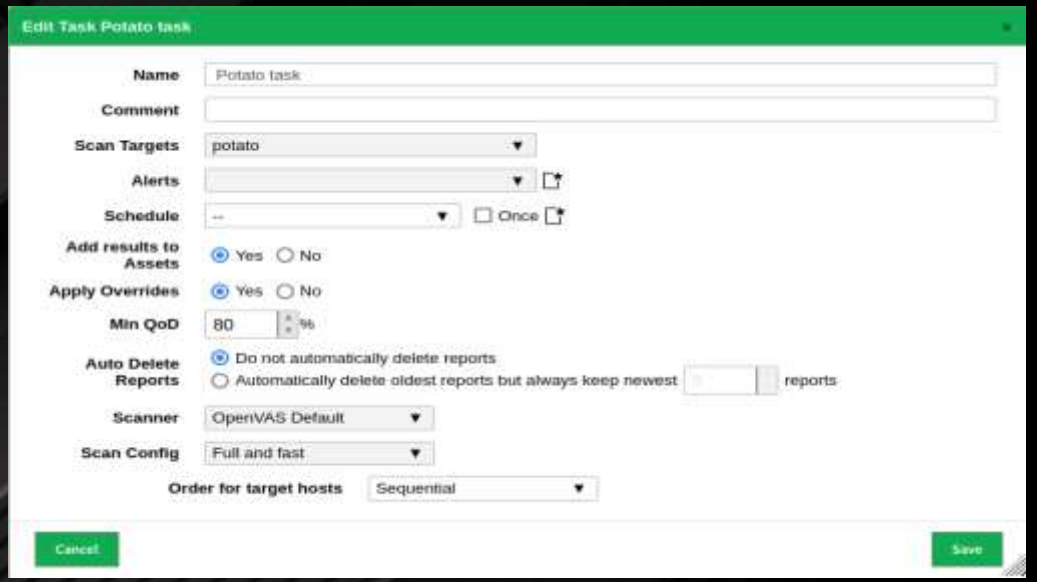

CONFIGURAZIONE SCAN NESSUS




ESEMPIO DI VULNERABILITA’ TROVATA

# 4. Vulnerability mapping

## TECNICHE AUTOMATICHE (OPENVAS)




CONFIGURAZIONE SCAN OPENVAS


SCAN OPENVAS

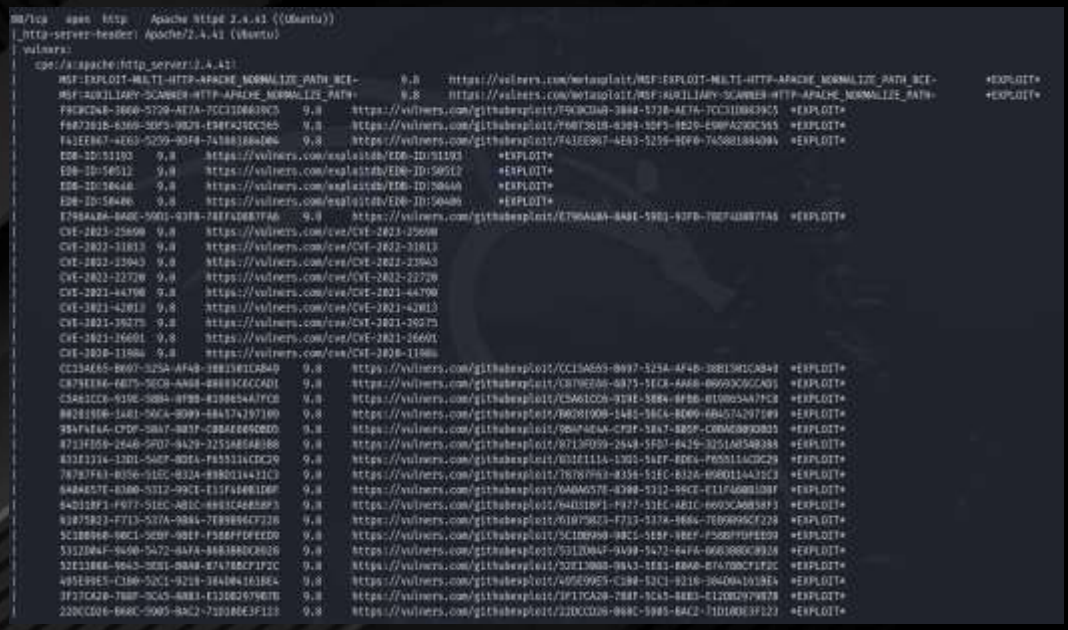

ESEMPIO DI VULNERABILITA' TROVATA

# 4. Vulnerability mapping

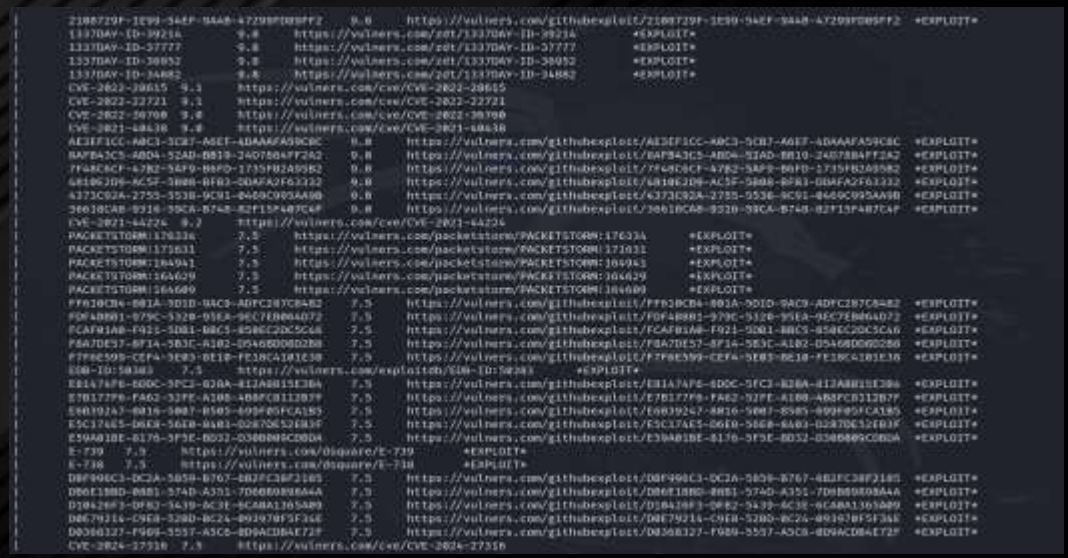
## ALTRE TECNICHE AUTOMATICHE

```
(kali㉿kali)-[~]
$ nikto -h 10.0.2.11 -C all -p 80
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          10.0.2.11
+ Target Hostname:    10.0.2.11
+ Target Port:        80
+ Start Time:         2024-07-02 12:33:34 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.4.41 (Ubuntu)
+ /: 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/we
b-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ Apache/2.4.41 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the
 EOL for the 2.x branch.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+ /admin/: This might be interesting.
+ /admin/index.php: This might be interesting: has been seen in web logs from an unknown scanner
.
+ 26640 requests: 0 error(s) and 6 item(s) reported on remote host
+ End Time:           2024-07-02 12:35:09 (GMT-4) (95 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

SCAN CON NIKTO



SCAN CON NMAP VULNERS PARTE 1



SCAN CON NMAP VULNERS PARTE 2

# 5. Target exploitation

- Nella fase di Target Exploitation, l'obiettivo principale è ottenere il controllo della macchina target, denominata "**potato**". Questo viene realizzato sfruttando le vulnerabilità rilevate nella fase precedente e utilizzando strumenti più invasivi per scoprirne altre nel caso in cui non dovessero bastare quelle scoperte.

- Quindi sono stati utilizzati i seguenti strumenti:

1. **Directory Busting (per scoprire directory e file nascosti)**:
   - Dirb
   - Gobuster
2. **Analisi e Manipolazione del Traffico**:
   - Burp Suite (utilizzato per l'analisi e la manipolazione del traffico del web server Apache)

# 5. Target exploitation

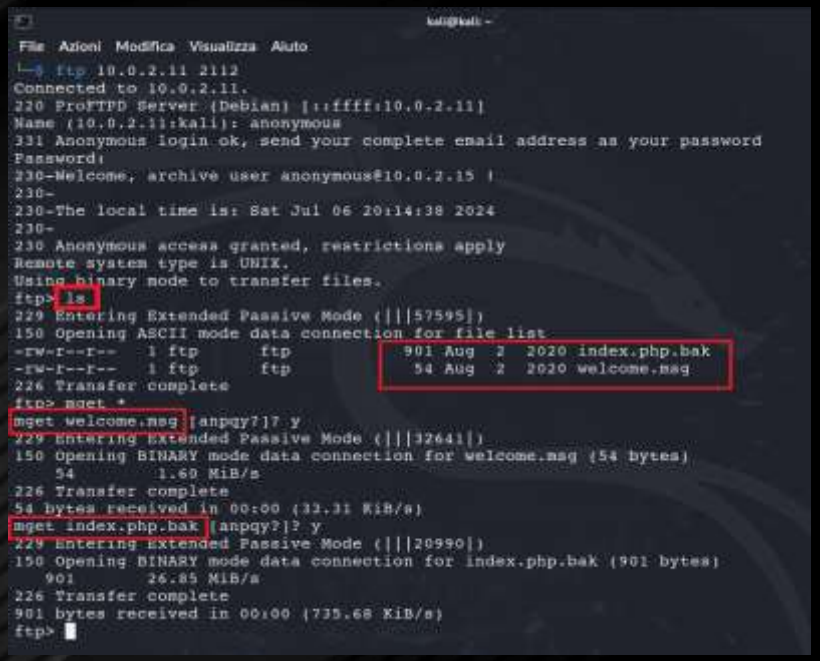
## SFRUTTAMENTO FTP ANONYMOUS

- Innanzitutto è stata sfruttata la vulnerabilità trovata da OpenVAS e Nmap per accedere ai file presenti sul servizio FTP per ottenere maggiori informazioni.

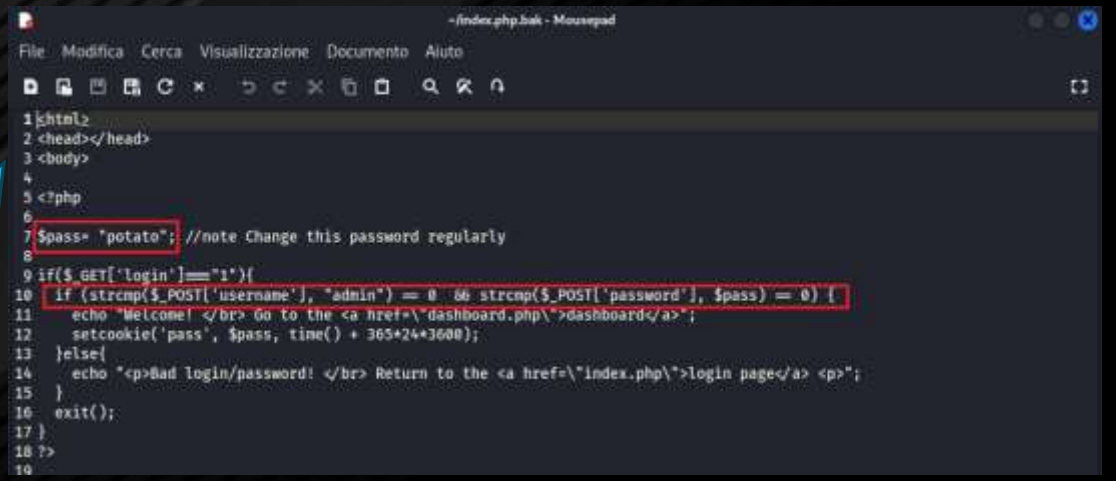
- Scopriamo l'esistenza di una variabile «**$pass**», potenzialmente manipolabile, oltre ad una vulnerabilità di **type juggling PHP**:

strcmp("foo", array()) => NULL + PHP Warning

Che per **NULL == 0** restituisce **true** , consentendoci l'accesso.




FTP ANONYMOUS LOGIN E OTTENIMENTO FILE



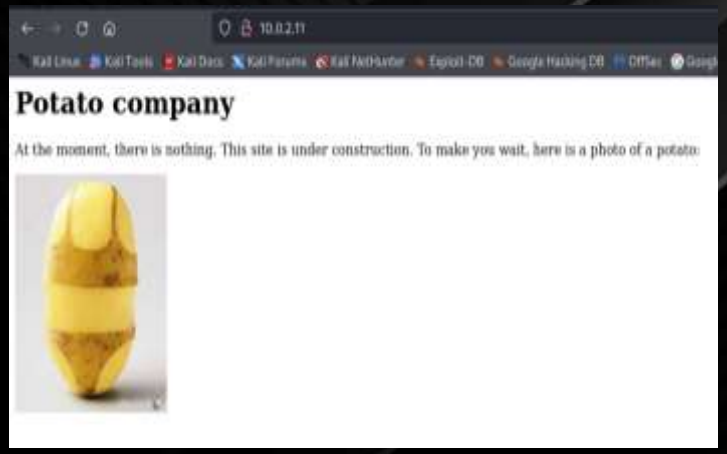

INDEX.PHP.BAK

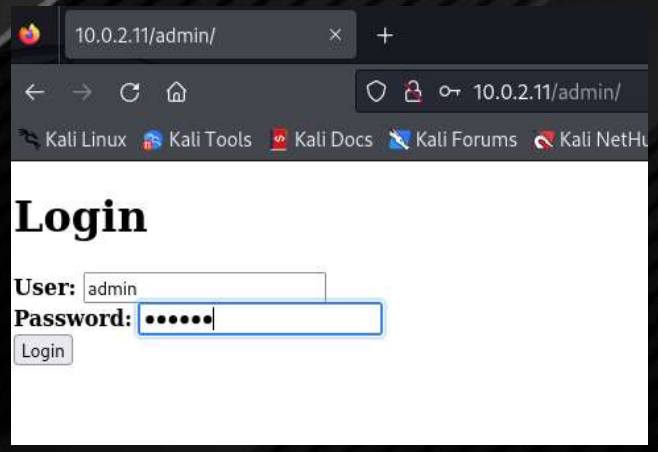
# 5. Target exploitation

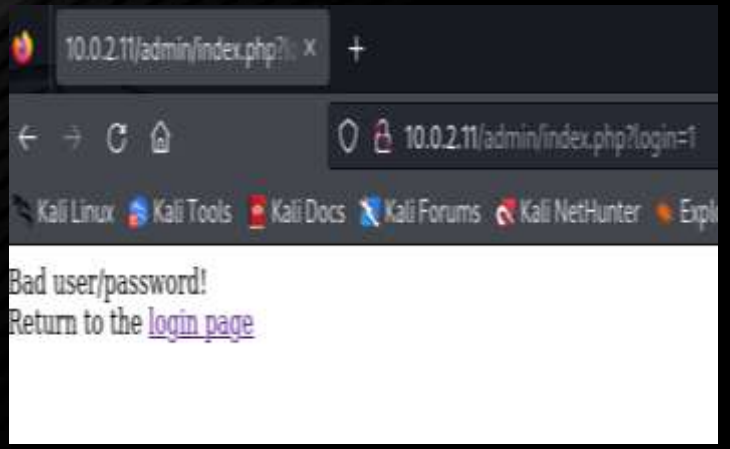
## ESPLORAZIONE WEB APPLICATION

Esploriamo la web application superficialmente e tentiamo il login (senza successo) con:

- username: «**admin**>
- password: «**potato**»



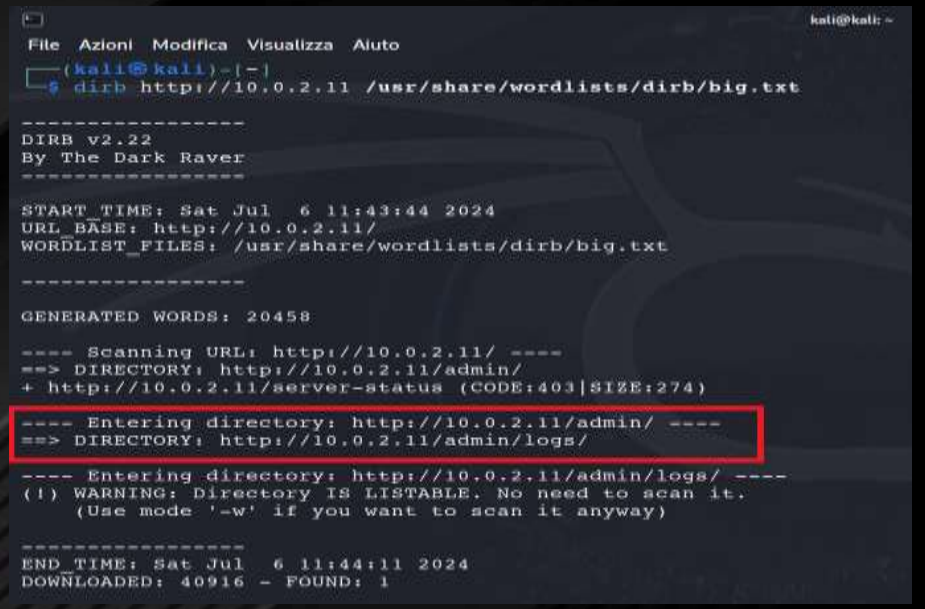



# 5. Target exploitation

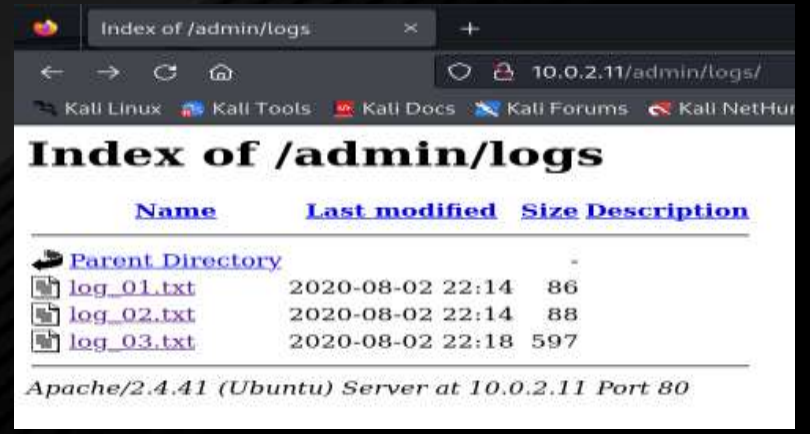
## ESPLORAZIONE WEB APPLICATION

- Tentando qualche tecnica di **Directory Busting**, scopriamo un percorso interessante con i file di log dell'admin e un cambio password.

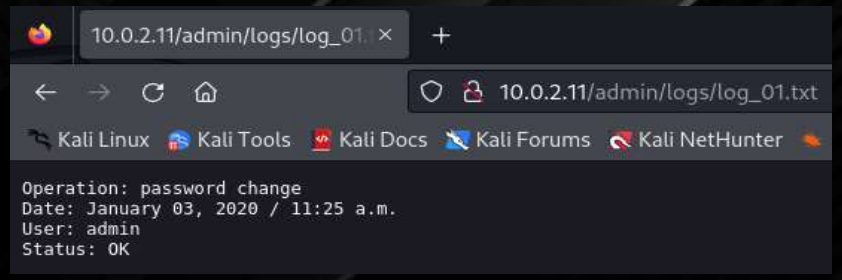
- Possibile vulnerabilità di **Local File Inclusion (LFI)**




DIRB SCAN




ADMIN LOGS

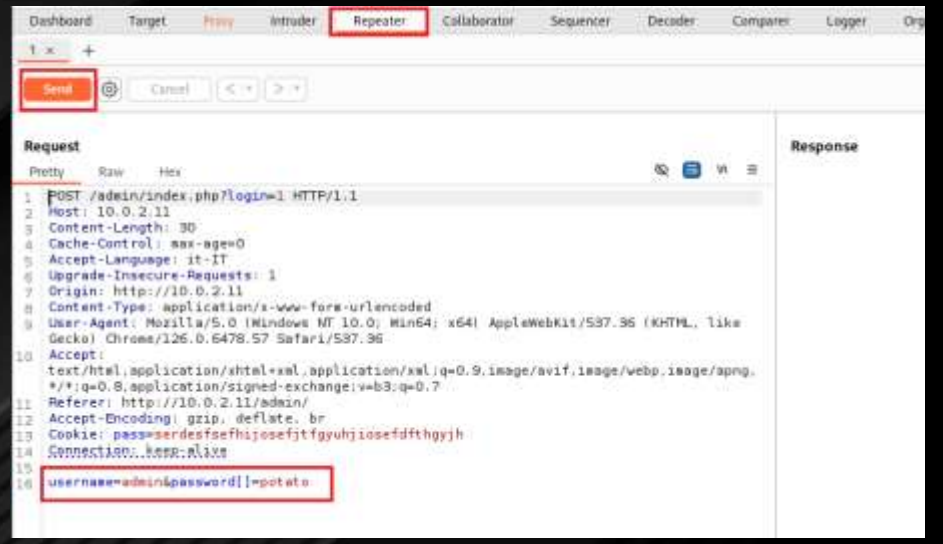



CONTENUTO FILE DI LOG

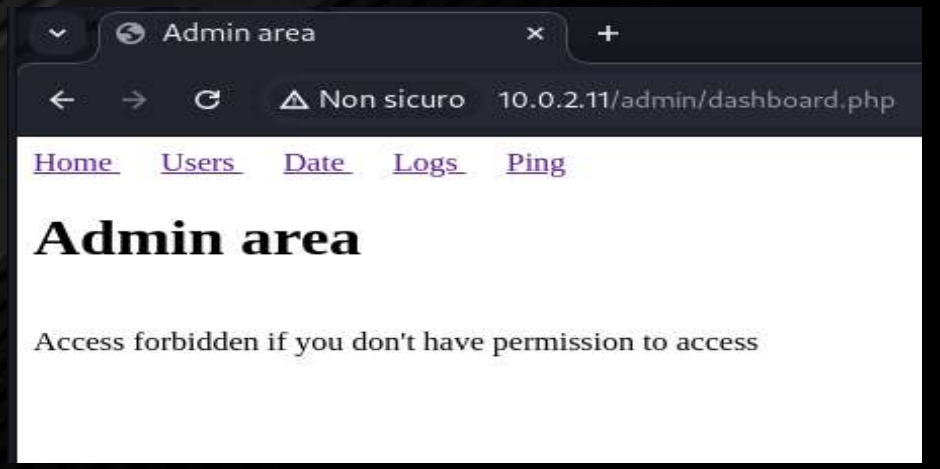
# 5. Target exploitation

## INTERCETTAZIONE RICHIESTE BURP SUITE (LOGIN)

- E' stata sfruttata la vulnerabilità di **type juggling** della **strcmp in PHP**, intercettando la richiesta **POST** e passando come password un **array**. Riusciamo quindi a loggarci come admin ed ad accedere ai **Logs**.




PASSIAMO UN ARRAY COME PASSWORD



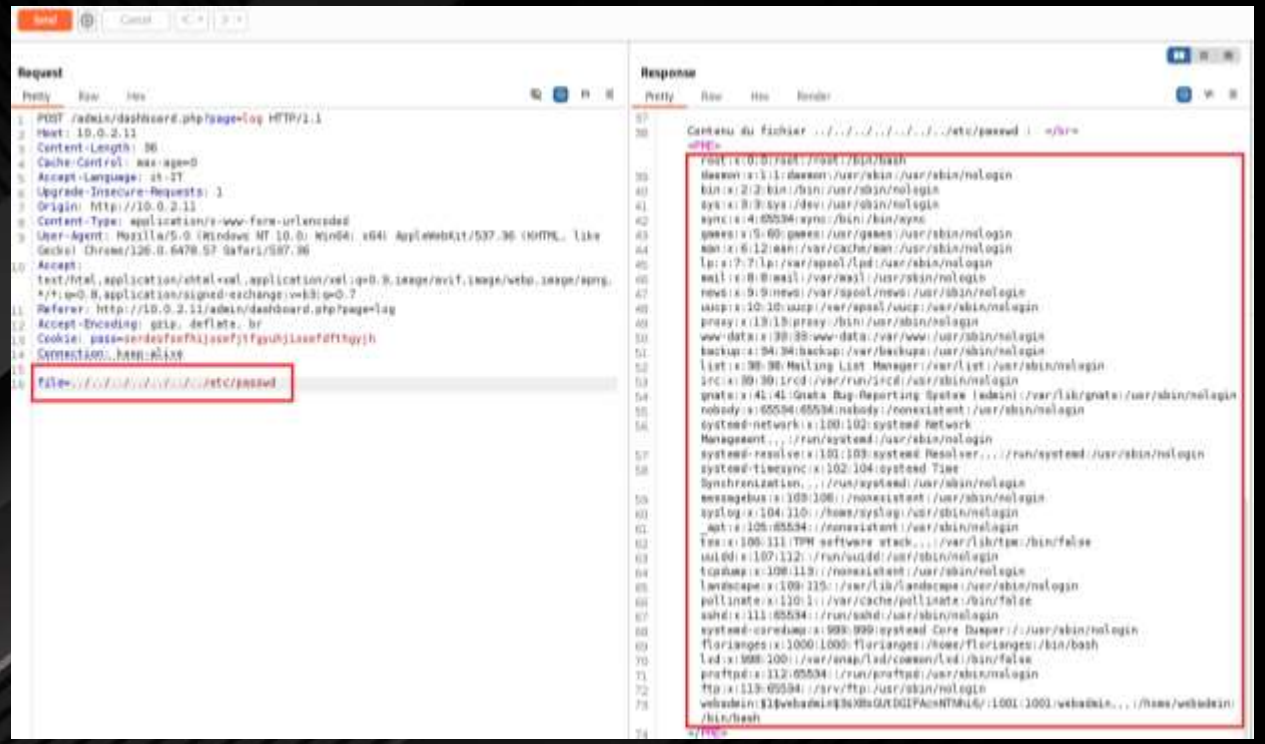

ACCESSO ALLA ADMIN AREA

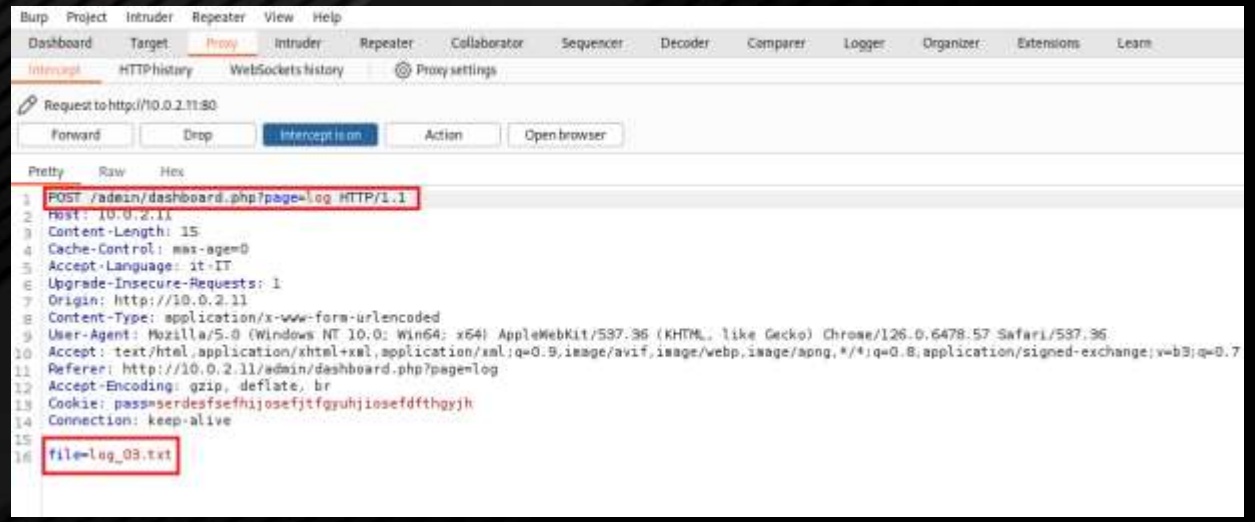

AREA LOGS

# 5. Target exploitation

## INTERCETTAZIONE RICHIESTE BURP SUITE (LOCAL FILE INCLUSION)

Intercettando la richiesta POST sul bottone «**Get the log**», possiamo inserire il percorso **/etc/passwd**, che non dovrebbe essere raggiungibile, ottenendone il contenuto.

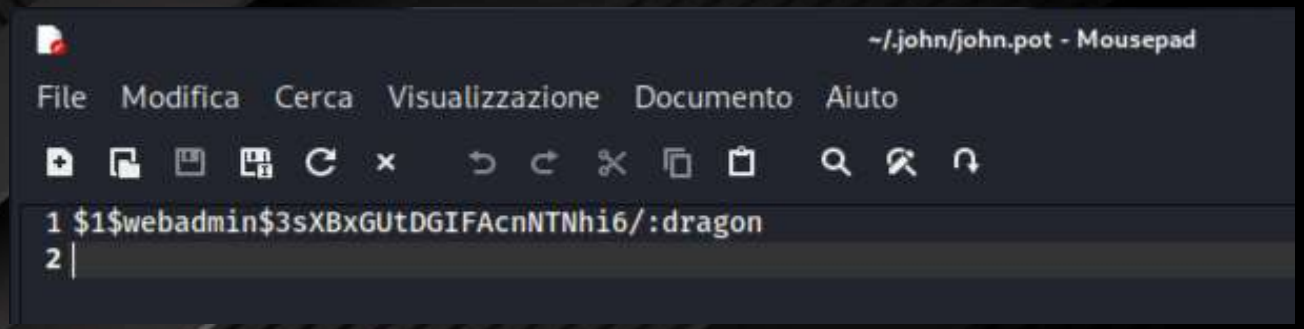


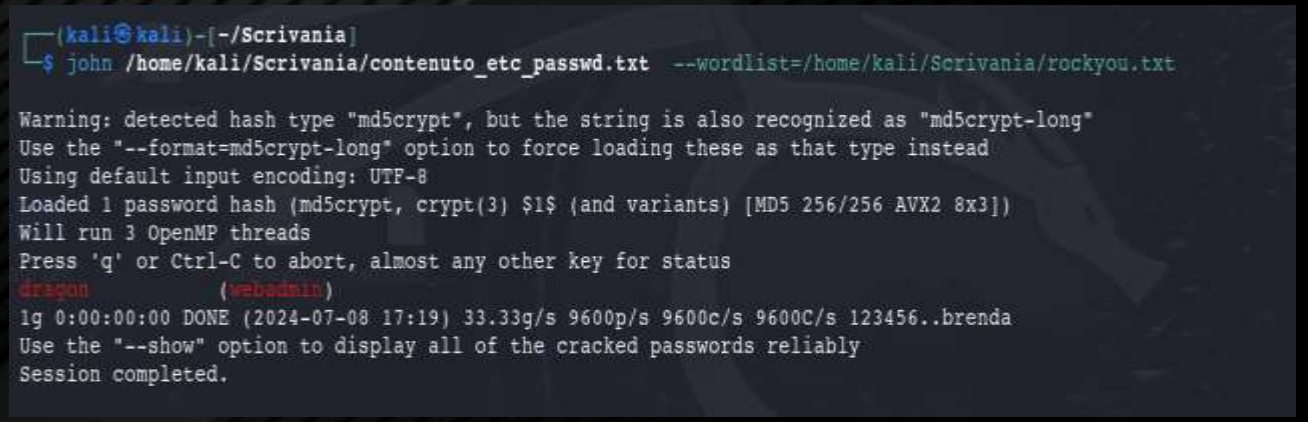



# 5. Target exploitation

## DECIFRATURA DELLA PASSWORD

- Passiamo la lista ottenuta al tool «**John the ripper**» e la wordlist «**rockyou**» per un attacco a dizionario.
- Ogni password verrà hashata in **md5** (message-digest) e confrontata con quelle presenti nella lista.
- Otteniamo la password dell'utente webadmin, ovvero **dragon**

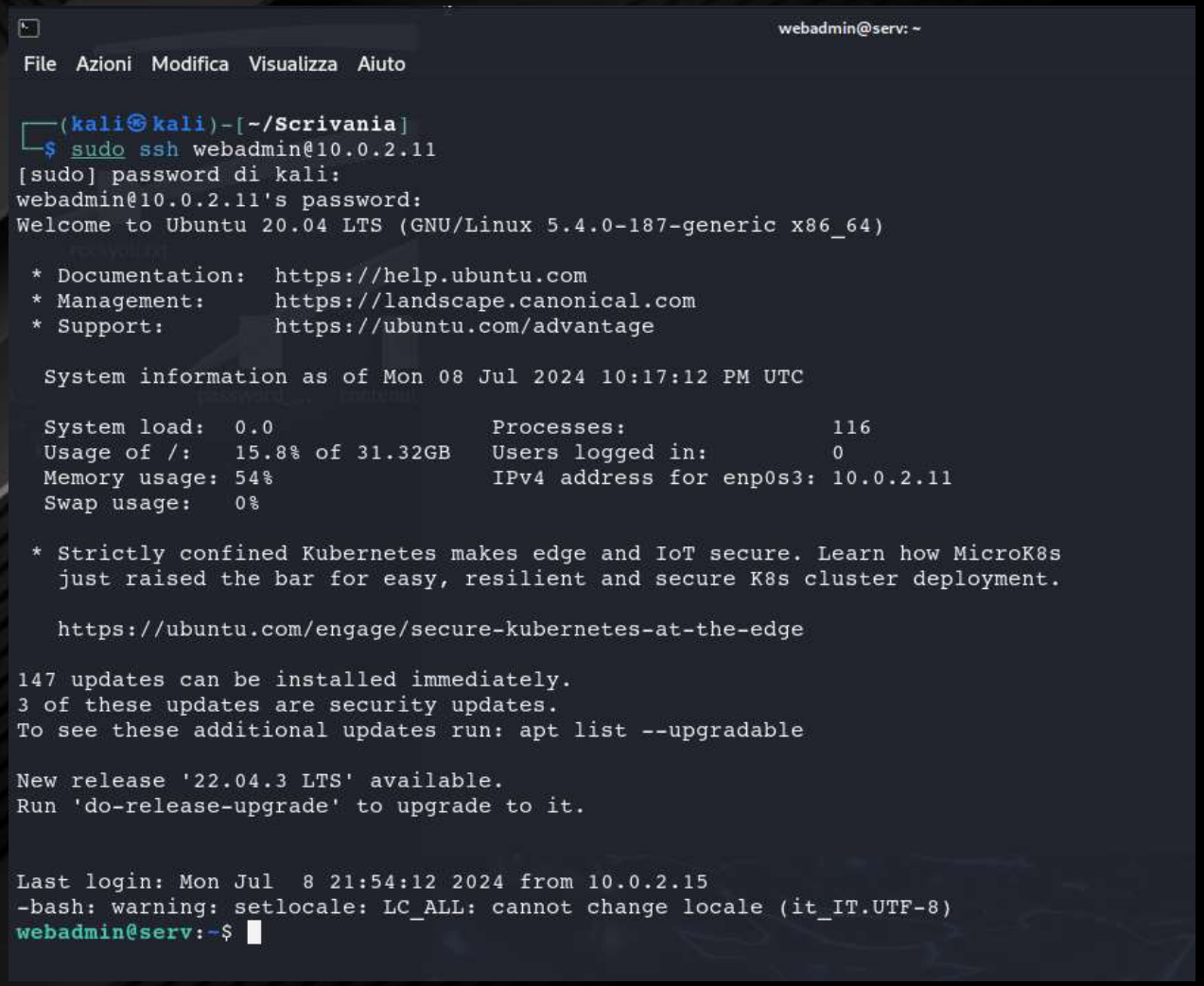


# 5. Target exploitation

## ACCESSO ALLA MACCHINA

- Concludiamo la fase di target exploitation, accedendo alla macchina target tramite servizio SSH come webadmin@10.0.2.11 con password «**dragon**», ottenendo con successo l'obiettivo.




ACCESSO ALLA HOME DI WEBADMIN

# 6. Post-Exploitation

Dopo aver ottenuto l’accesso alla macchina target nella fase di target exploitation, abbiamo 2 obiettivi principali da raggiungere:

1. **Privilege Escalation**: Ottenere i privilegi di root per avere il controllo completo del sistema.
2. **Maintaining Access**: Stabilire una backdoor per mantenere un accesso costante alla macchina.


# 6. Post-Exploitation

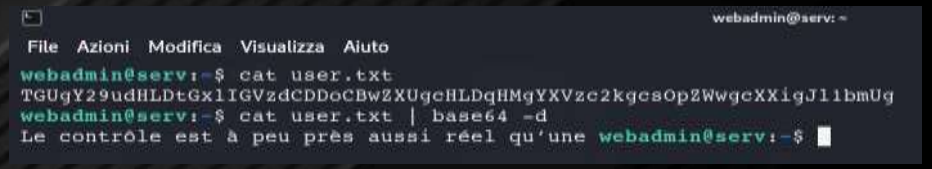
## PRIVILEGE ESCALATION

Tramite l'utente **webadmin**, proveremo a sfruttare possibili file o directory con:

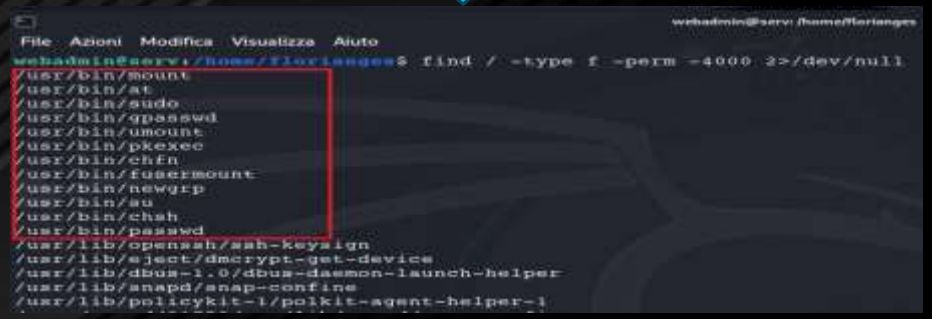
- con bit **SUID** (Set user ID) attivo
- permessi **ingiustamente elevati.**

Per elevare i nostri permessi a quelli di **root**.


HOME WEBADMIN

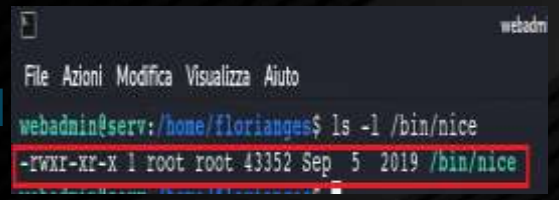



USER.TXT BASE64

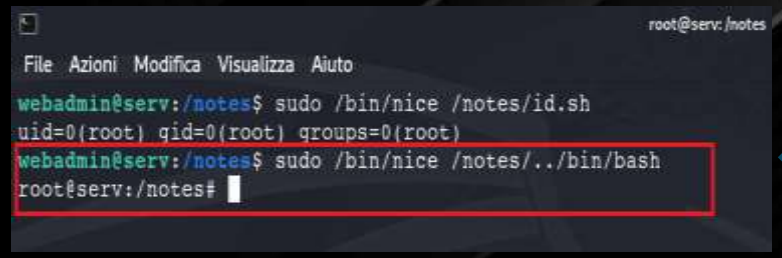



RICERCA FILE SUID


COMANDI SUDO




PERMESSI /BIN/NICE

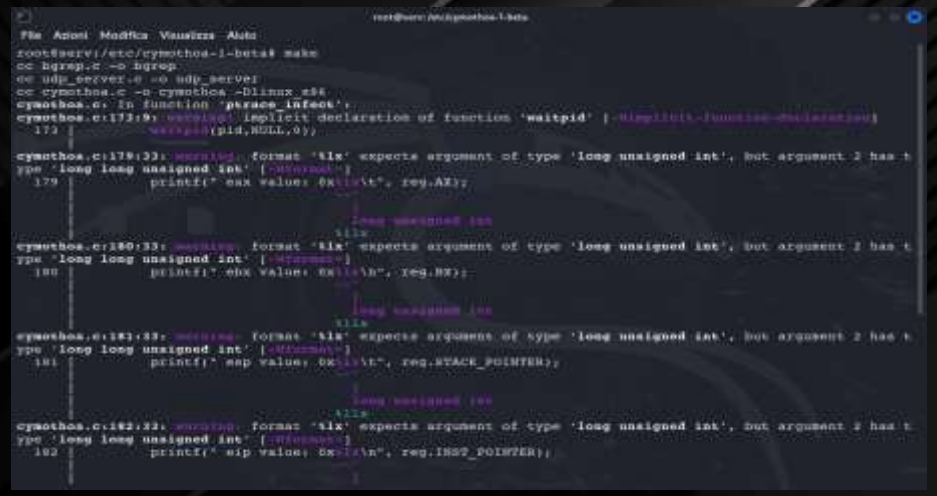



OTTENIMENTO PERMESSI ROOT

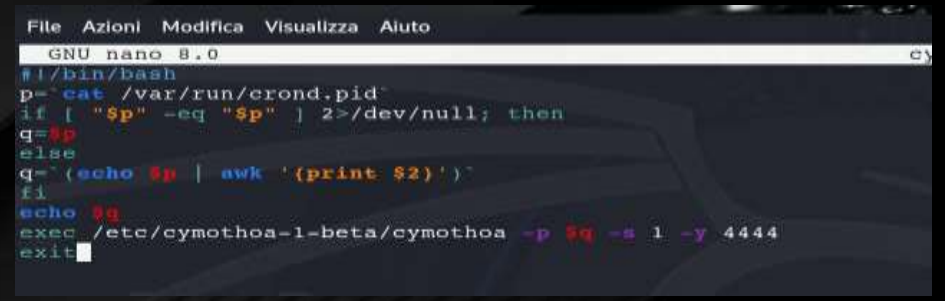
# 6. Post-Exploitation

## MAINTAINING ACCESS (TRASFERIMENTO FILE)

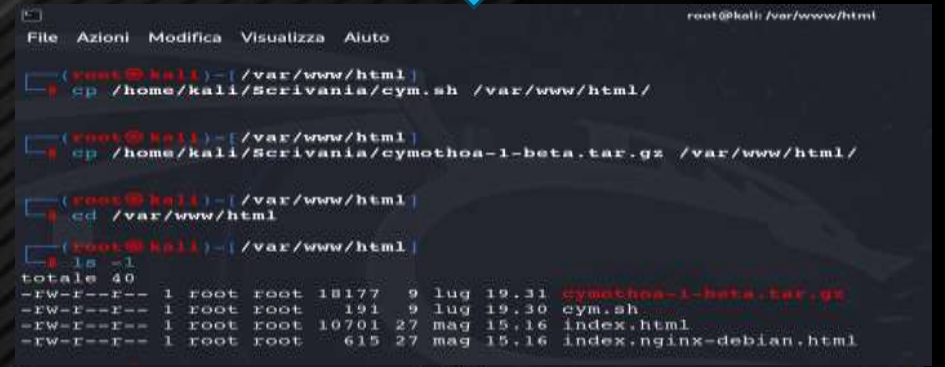
La backdoor viene trasferita alla macchina target tramite **server apache**.




MAKE INSTALL CYMOTHOA




SCRIPT CYMOTHOA





CARICAMENTO SU SERVER APACHE + WGET

# 6. Post-Exploitation

## MAINTAINING ACCESS (BACKDOOR)

In questa ultima fase del penetration test, creiamo gli script di avvio automatico e persistenza della backdoor

root@serv: /etc/systemd/system

File Azioni Modifica Visualizza Aiuto

```
  GNU nano 4.8                                    rc-local.service
[Unit]
Description=Avvio della backdoor all'avvio del sistema
After=network.target

[Service]
Type=forking
ExecStart=/etc/rc.local start
TimeoutSec=0
StandardOutput=tty
RemainAfterExit=yes
SysVStartPriority=99

[Install]
WantedBy=multi-user.target
```

RC-LOCAL.SERVICE

root@serv: /etc

File Azioni Modifica Visualizza Aiuto

```
root@serv:/etc# printf '%s\n' '#!/bin/bash' 'exit 0' | sudo tee -a /etc/rc.local
#!/bin/bash
exit 0
root@serv:/etc# nano rc.local


Use "fg" to return to nano.

[1]+  Stopped                 nano rc.local
root@serv:/etc# nano rc.local
root@serv:/etc# chmod +x /etc/rc.local
root@serv:/etc#
```

CREO RC.LOCAL + CHMOD

root@serv: /etc

File Azioni Modifica Visualizza Aiuto

```
  GNU nano 4.8                                      rc.local
#!/bin/bash
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

echo "Ubuntu 20.04 LTS" >/etc/issue
echo "IP address: $(hostname -I)\n">>/etc/issue

sh /etc/init.d/cym.sh

exit 0
```

MODIFICA A RC.LOCAL

root@serv: /etc

File Azioni Modifica Visualizza Aiuto

```
root@serv:/etc# sed -i '$d' /etc/rc.local
root@serv:/etc# echo "sh /etc/init.d/cym.sh" >> /etc/rc.local
root@serv:/etc# echo "exit 0" >> /etc/rc.local
root@serv:/etc#
```

PERSISTENZA AL RIAVVIO

# GRAZIE PER L’ATTENZIONE