



Instituto Tecnológico de San Juan del Río



Tópicos de Ciberseguridad

Parchar un ejecutable.

P R E S E N T A:

**Montes Barajas Leonardo Daniel
Jorge Yael Ramírez García
Erika Olvera Pérez**

ISIC

**21590293
20590283
21590394**

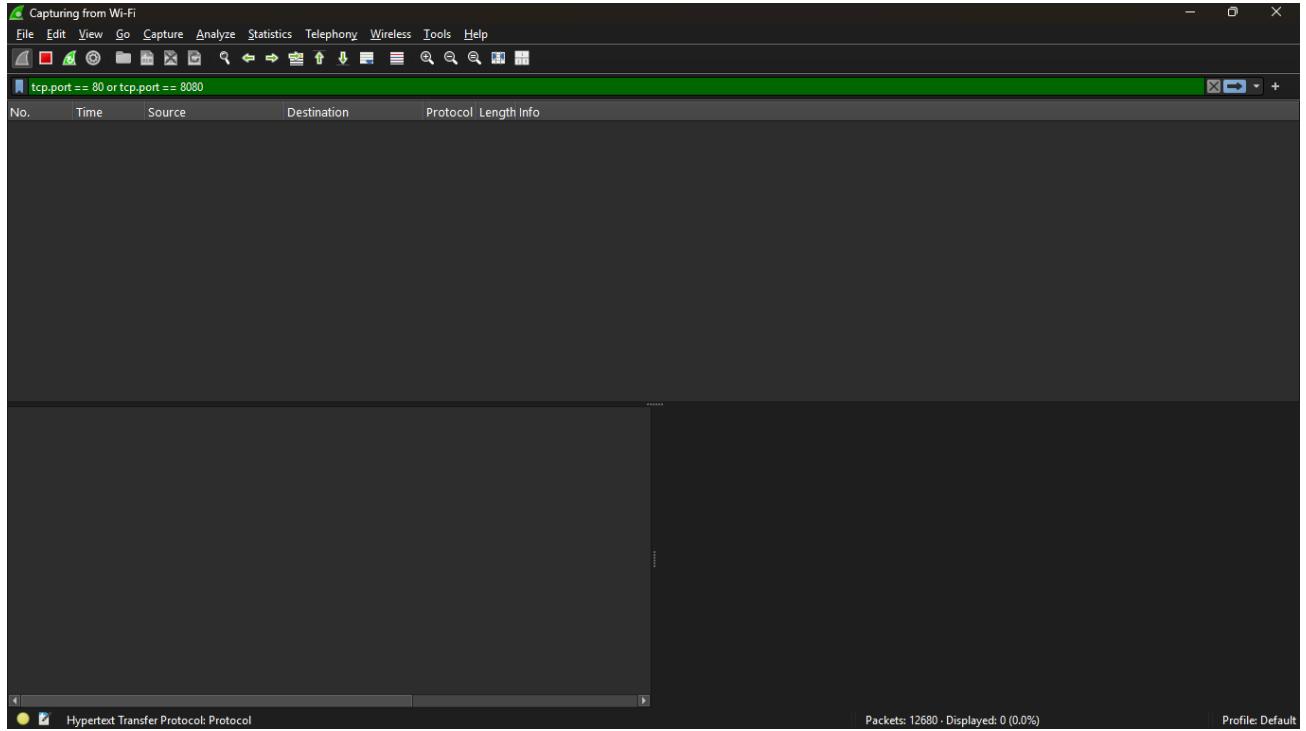
PERIODO [Ene - Jun 26]



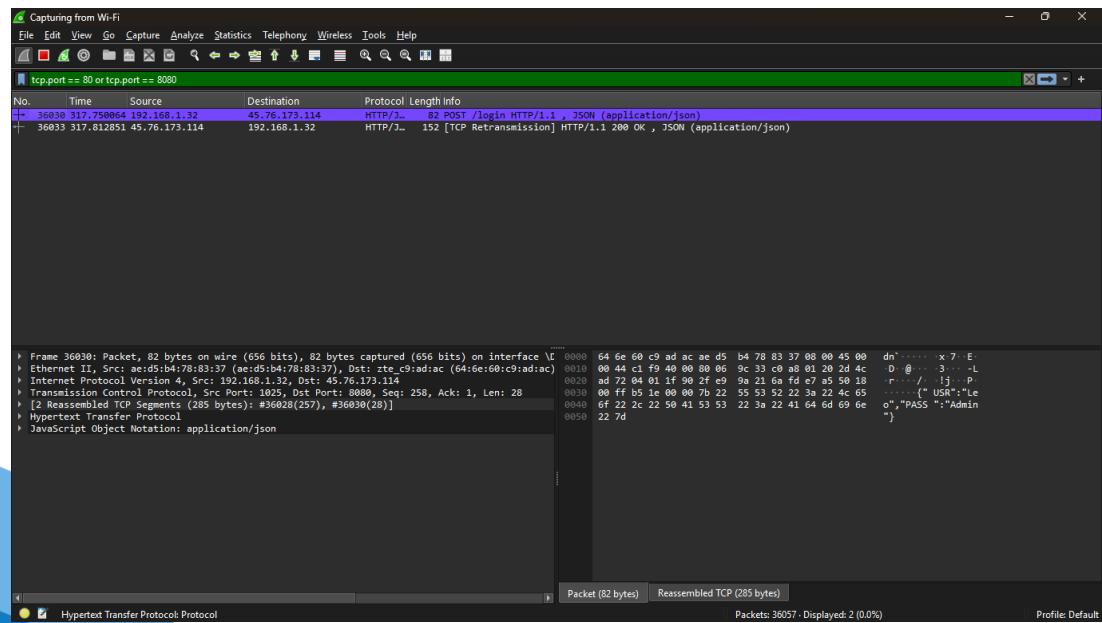
Vamos a empezar con identificar la url y el método HTTP

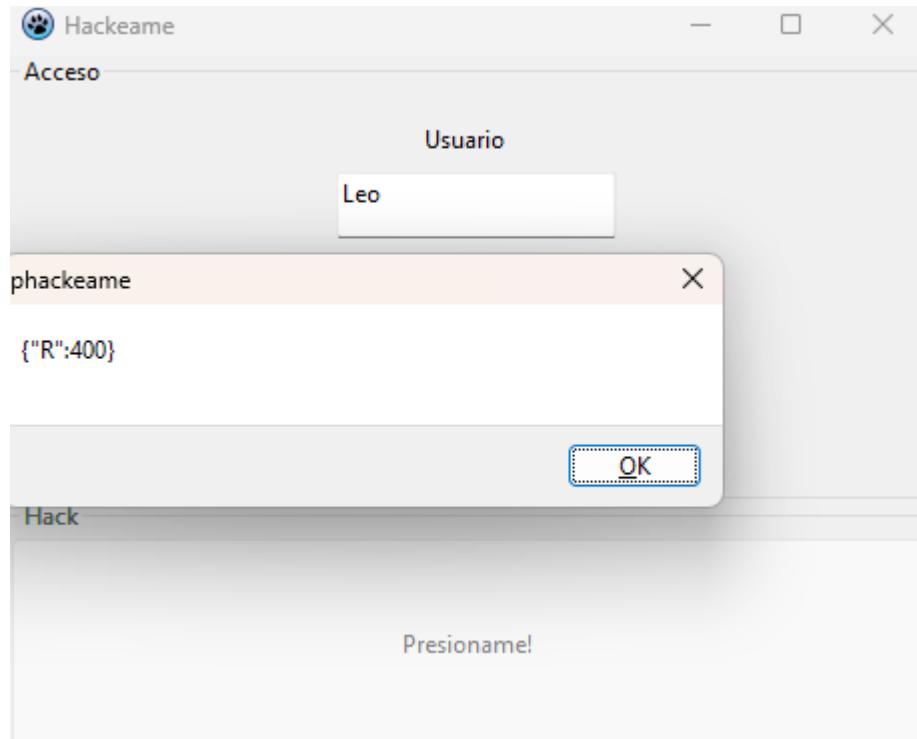
Para ello se usará Wireshark

Se filtra con `tcp.port == 80 or tcp.port == 8080` para ver el método http



Ahora se hará un intento de acceso para que nos proporcione información:





Información relevante extraída:

- ✚ **Método:** POST
- ✚ **Endpoint:** /login
- ✚ **Servidor real:** 45.76.173.114:8080
- ✚ **Formato:** JSON con campos USR y PASS
- ✚ **Respuesta:** HTTP 200 con JSON

Ahora lo que se buscara:

Crear un servidor Python en tu máquina que siempre responda "OK"

Redirigir el exe hacia tu servidor modificando el archivo hosts

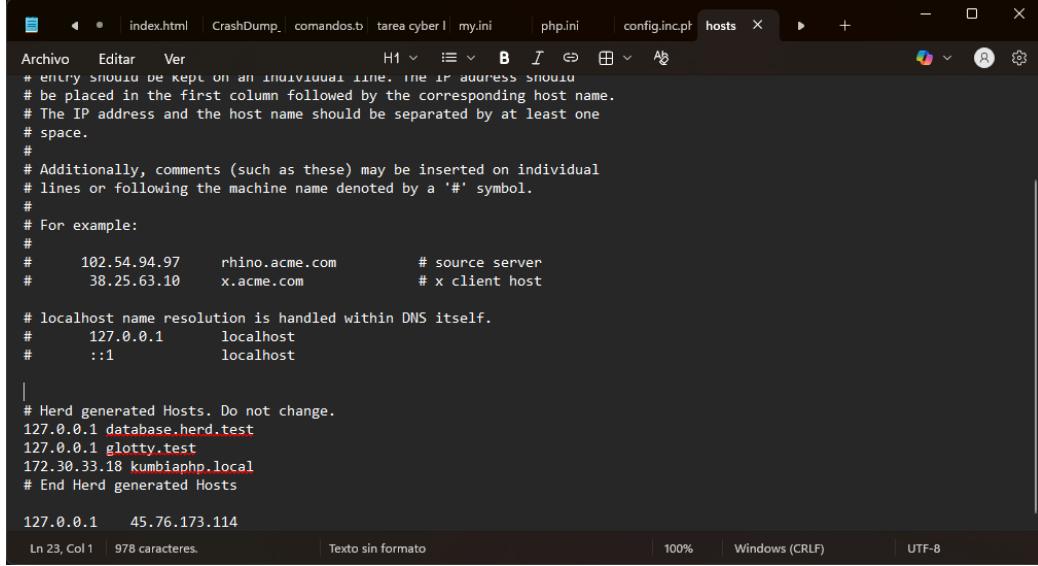
Se crea el servidor en flask:

```

Unidad1 > Practica4 > serverHack.py > ...
  1  from flask import Flask, request, jsonify
  2
  3  app = Flask(__name__)
  4
  5  @app.route('/login', methods=['POST'])
  6  def login():
  7      return jsonify({"status": "ok", "message": "Login correcto"}), 200
  8
  9  if __name__ == '__main__':
10      app.run(host='0.0.0.0', port=8080)

```

Se ejecuta el bloc de notas y se edita la última línea para engañar el exe e interceptarlo:



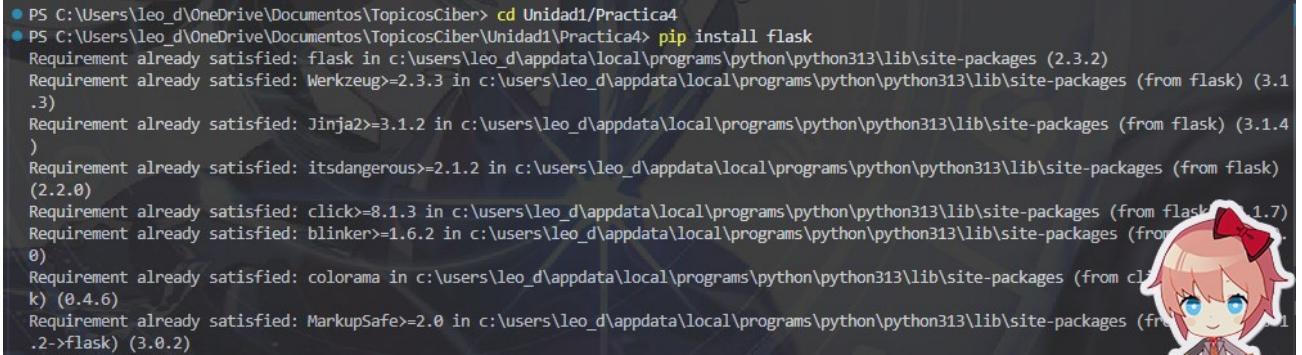
```
# entries should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97    rhino.acme.com      # source server
#      38.25.63.10     x.acme.com          # x client host

# localhost name resolution is handled within DNS itself.
#      127.0.0.1        localhost
#      ::1              localhost

#
# Hand generated Hosts. Do not change.
127.0.0.1 database.herd.test
127.0.0.1 glotty.test
172.30.33.18 kumbiaphp.local
# End Hand generated Hosts

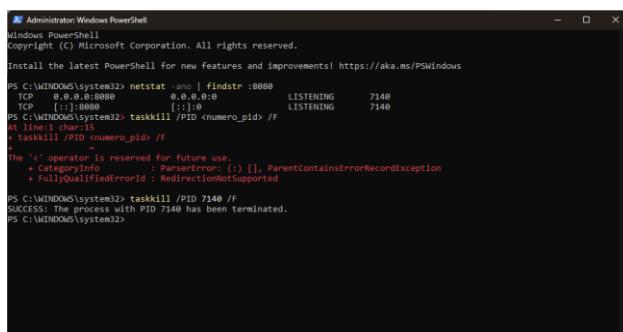
127.0.0.1 45.76.173.114
```

Se instala Flask:



```
PS C:\Users\leo_d\OneDrive\Documentos\TopicsCiber> cd Unidad1\Practica4
PS C:\Users\leo_d\OneDrive\Documentos\TopicsCiber\Unidad1\Practica4> pip install flask
Requirement already satisfied: flask in c:\users\leo_d\appdata\local\programs\python\python313\lib\site-packages (2.3.2)
Requirement already satisfied: Werkzeug>=2.3.3 in c:\users\leo_d\appdata\local\programs\python\python313\lib\site-packages (from flask) (3.1.3)
Requirement already satisfied: Jinja2>=3.1.2 in c:\users\leo_d\appdata\local\programs\python\python313\lib\site-packages (from flask) (3.1.4)
Requirement already satisfied: itsdangerous>=2.1.2 in c:\users\leo_d\appdata\local\programs\python\python313\lib\site-packages (from flask) (2.2.0)
Requirement already satisfied: click>=8.1.3 in c:\users\leo_d\appdata\local\programs\python\python313\lib\site-packages (from flask) (1.7)
Requirement already satisfied: blinker>=1.6.2 in c:\users\leo_d\appdata\local\programs\python\python313\lib\site-packages (from flask) (0)
Requirement already satisfied: colorama in c:\users\leo_d\appdata\local\programs\python\python313\lib\site-packages (from flask) (0.4.6)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\leo_d\appdata\local\programs\python\python313\lib\site-packages (from flask) (2.2->flask) (3.0.2)
```

Mato el proceso que ya tenia en el puerto 8080:



```
Administrator:Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> netstat -ano | findstr :8080
TCP 0.0.0.0:8080               0.0.0.0:0                LISTENING      7140
TCP 0.0.0.0:8080               0.0.0.0:0                LISTENING      7140
PS C:\WINDOWS\system32> taskkill /PID <número_pid> /F
At line:1 char:15
+ taskkill /PID <número_pid> /F
+
The '<' operator is reserved for future use.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : RedirectionNotSupported

PS C:\WINDOWS\system32> taskkill /PID 7140 /F
SUCCESS: The process with PID 7140 has been terminated.
PS C:\WINDOWS\system32>
```

Se ejecuta el servidor:

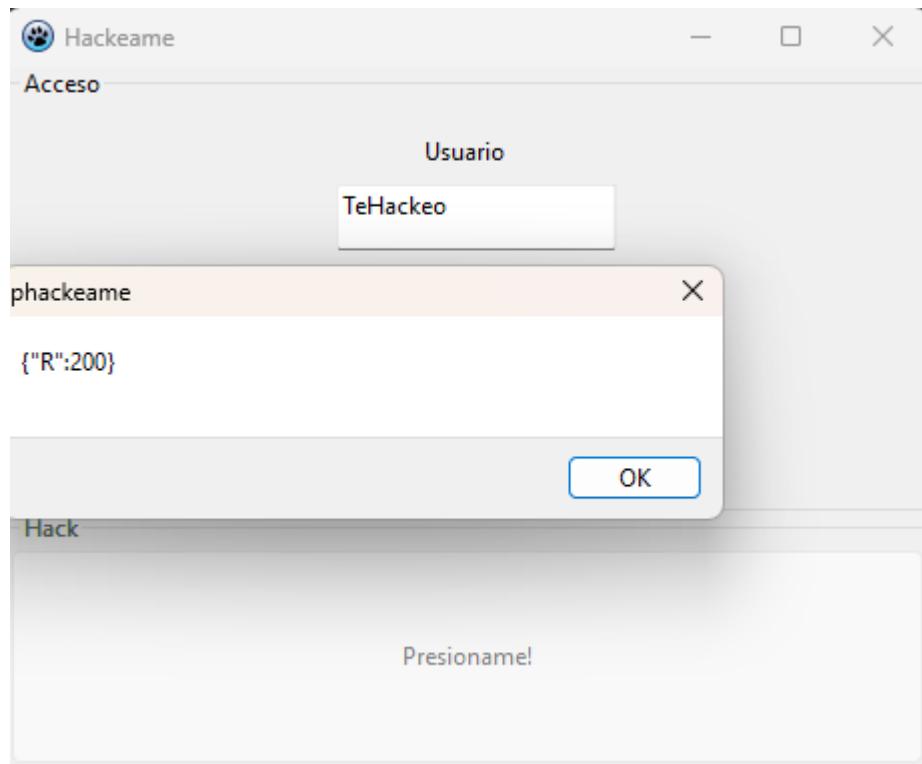
```
PS C:\Users\leo_d\OneDrive\Documentos\TopicsCiber\Unidad1\Practica4> python serverHack.py
 * Serving Flask app 'serverHack'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:8080
 * Running on http://192.168.1.32:8080
Press CTRL+C to quit
```

Se obtuvo un 400 entonces se modificará el json para que de un método html 200. Reinicio de servidor.

Se agrega el alias al adaptador:

```
PS C:\WINDOWS\system32> netsh interface ip add address "Loopback Pseudo-Interface 1" 45.76.173.114 255.255.255.255
```

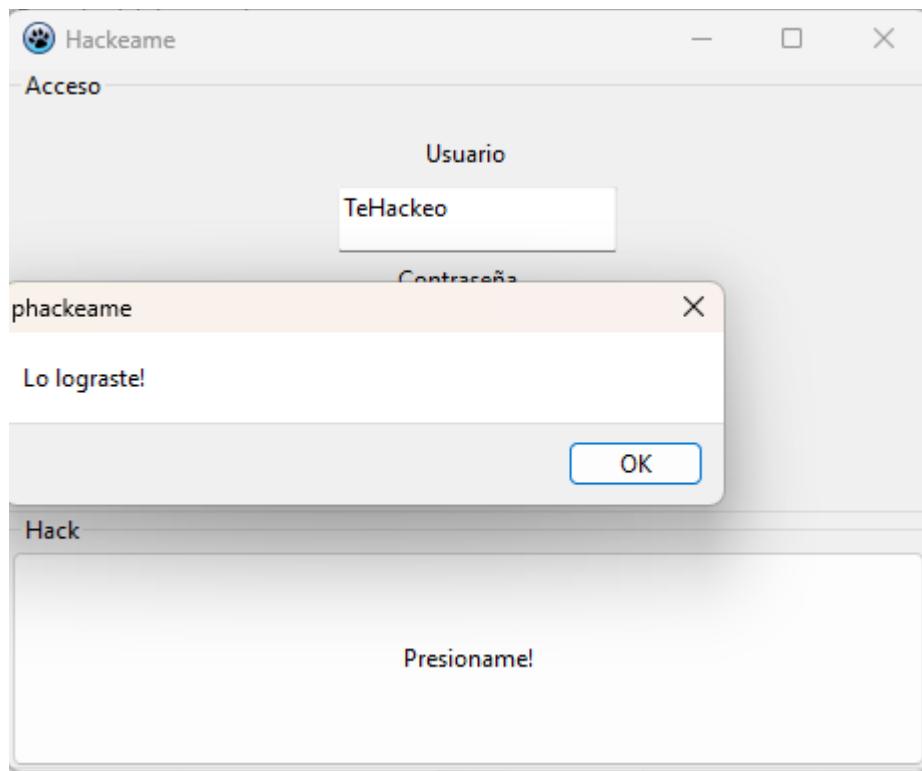
Se vuelve a intentar y se obtiene un 200 es decir success:



Además, en la consola vemos el flujo de actividad:

```
PS C:\Users\leo_d\OneDrive\Documentos\TopicsCiber\Unidad1\Practica4> python serverHack.py
 * Serving Flask app 'serverHack'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://45.76.173.114:8080
Press CTRL+C to quit
45.76.173.114 - - [19/Feb/2026 04:44:48] "POST /login HTTP/1.1" 200 -
```

Y lo logramos se desbloquea el botón:



Metodo:

Man in the Middle (MitM) combinado con API Spoofing o Server Impersonation.

- ⊕ Traffic Interception - Capturaste el tráfico con Wireshark para analizar la comunicación
- ⊕ IP Hijacking - Redirigiste el tráfico destinado al servidor real hacia tu máquina
- ⊕ Fake API / Rogue Server - Creaste un servidor falso que impersona al servidor legítimo y siempre responde con éxito

Código Final Utilizado:

```
Unidad1 > Practica4 > serverHack.py > ...
1   from flask import Flask, request, jsonify
2
3   app = Flask(__name__)
4
5   @app.route('/login', methods=['POST'])
6   def login():
7       return jsonify({"R": 200}), 200
8
9   if __name__ == '__main__':
10      app.run(host='45.76.173.114', port=8080)
```