

# CTF 02 - Reto Python

Autor: Jesús Díaz  
Bootcamp Cyberskills Read Team 2025  
Fecha: 22/05/2025



⚠ Este reto se realizó en un laboratorio virtual controlado con fines educativos. Todos los usuarios y contraseñas son ficticios y no representan sistemas reales.

## Contenido

<b>1. Resumen</b> .....	2
<b>2. Objetivos</b> .....	2
<b>3. Entorno de pruebas</b> .....	2
<b>4. Desarrollo paso a paso</b> .....	2
user1 → user2 .....	3
user 2 → user3 .....	4
user3 → user4 .....	5
user4 → user5 .....	6
user5 → user6 .....	7
user6 → user7 .....	8
user7 → user8 .....	10
user8 → user9 .....	11
user9 → user10 .....	14
user10 → root .....	16
<b>5. Resultados</b> .....	16
<b>6. Aviso legal</b> .....	16

## 1. Resumen

Este reto forma parte del Bootcamp de Ciberseguridad Ofensiva (Red Team) *Cyberskills 2025*. Se trata de un entorno controlado tipo **CTF** (Capture The Flag) en una máquina Ubuntu, diseñado para practicar **análisis de scripts en Python, escalada de privilegios y pensamiento lógico**.

El objetivo es comenzar con un usuario básico (`user1`) e ir progresando hasta obtener acceso como `root`, resolviendo diferentes niveles mediante la interacción con scripts Python y la comprensión de su funcionamiento interno.

## 2. Objetivos

- Analizar el comportamiento de distintos scripts en Python.
- Comprender técnicas básicas de **reversing, decodificación, y automatización de tareas**.
- Practicar la **escalada de privilegios por análisis de código**.
- Desarrollar habilidades de **pensamiento analítico y resolución de retos** en entornos Linux.
- Demostrar la capacidad de seguir una metodología de pentesting controlada y documentada.

## 3. Entorno de pruebas

- **Sistema Operativo:** Ubuntu (entorno virtualizado).
- **IP de la máquina objetivo:** 10.0.3.9
- **Usuario inicial:** user1
- **Contraseña:** St@rt1ng
- **Herramientas utilizadas:**
  - SSH, Python3, cat, yes, head, base64, requests, socket
  - Entorno de ejecución en **laboratorio virtual seguro y aislado**.

## 4. Desarrollo paso a paso

Cada nivel representa un usuario diferente (de `user1` a `user10`). El progreso se logra identificando patrones de vulnerabilidad en los scripts Python.

Resumen de las técnicas aplicadas:

Nivel	Usuario	Técnica utilizada	Contraseña obtenida
1	user1 → user2	Ejecución directa del script	S0Easy&Gu1d3d
2	user2 → user3	Lectura del código y comentario oculto	N0C0mm3nts !
3	user3 → user4	Decodificación Base64	H4rdC0de
4	user4 → user5	Ejecución con parámetros	P@ramet3rs
5	user5 → user6	Petición HTTP externa + clave oculta	Ext3rnalK3Y
6	user6 → user7	Manipulación de archivos y conteo de líneas	=OneM1llion
7	user7 → user8	Comunicación cliente-servidor (sockets)	TCP!S0ck3t
8	user8 → user9	Lógica y tiempo de ejecución	N0tS0Rand0m!

9	user9 → user10	Análisis de privilegios y decodificación hexadecimal	ModifyPrives&Win
10	user10 → root	Escalada mediante grupo sudo	Acceso root

## user1 → user2

- Nos conectamos con el user1 por SSH a la máquina 10.0.3.9 y listamos (ls -la):

```
ssh user1@10.0.3.9

user1@retopython:~$ ls -la
total 72
drwxr-x--- 14 user1 user1 4096 ene 26 2023 .
drwxr-xr-x 13 root  root  4096 may 22 2023 ..
-rw-------  1 user1 user1   329 jun  9 18:10 .bash_history
-rw-r--r--  1 user1 user1  220 ene 26 2023 .bash_logout
-rw-r--r--  1 user1 user1 3771 ene 26 2023 .bashrc
drwx----- 10 user1 user1 4096 jun  9 15:11 .cache
drwx----- 12 user1 user1 4096 ene 26 2023 .config
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 Descargas
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 Documentos
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 Escritorio
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 Imágenes
drwx-----  3 user1 user1 4096 ene 26 2023 .local
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 Música
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 Plantillas
-rw-r--r--  1 user1 user1   807 ene 26 2023 .profile
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 Público
drwx-----  3 user1 user1 4096 ene 26 2023 snap
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 Vídeos
```

- La carpeta Documentos es accesible por owner y grupo (drwxr-xr-x), por tanto user1 y miembros del grupo user1 pueden entrar y otros también tienen lectura/ejecución (listado).
- Cambiamos a Documentos y listamos:

```
user1@retopython:~$ cd Documentos
user1@retopython:~/Documentos$ ll
total 16
drwxr-xr-x  2 user1 user1 4096 ene 26 2023 ../
drwxr-x--- 14 user1 user1 4096 ene 26 2023 ../
-rwxr-x---  1 user1 user1   256 ene 26 2023 nivel1.py*
-rwxr-x---  1 user1 user1   737 ene 26 2023 README.txt*
```

- nivel1.py y README.txt tienen permiso ejecutable para el propietario (-rwxr-x---): propietario tiene rwx, grupo r-x, otros ---. El \* indica ejecutable (según el alias ll).
- Contenido de README.txt:

```
user1@retopython:~/Documentos$ cat README.txt

Reglas:

- Prohibido modificar el contenido de los scripts (el resto de ficheros pueden modificarse).
- Prohibido escribir ficheros nuevos en el directorio.
- La contraseña del siguiente usuario deberá obtenerse EJECUTANDO el script e interactuando con él de alguna forma.
- Se permite leer el código de los scripts para entender su funcionamiento.
- Una vez se supere el nivel, el script mostrará las credenciales del siguiente usuario.
```

**Objetivo:**

- Ejecutar comandos como usuario root

**Ejemplo de uso:**

- En este mismo directorio encontrarás un script que te dará la contraseña para user2

- Ejecútalo:

> python3 nivel1.py

- Accede al siguiente usuario:

> su user2

**¿Serán tan fáciles los siguientes niveles?**

**Suerte**

- El script nivel1.py, al ejecutarlo, muestra la contraseña de user2.

```
user1@retopython:~/Documentos$ python3 nivel1.py
La pass de user2 es: S0Easy&Gu1d3d
```

## user 2 → user3

- Cambiamos a user2 y listamos el directorio Documentos:

```
user1@retopython:~/Documentos$ su user2
Password:

user2@retopython:/home/user1/Documentos$ cd /home/user2/Documentos

user2@retopython:~/Documentos$ ll
total 12
drwxr-xr-x  2 user2 user2 4096 ene 26  2023 ./
drwxr-x--- 14 user2 user2 4096 jun  9 18:10 ../
-rwxr-x---  1 user2 user2  430 ene 26  2023 nivel2.py*
```

- Ejecutamos el script nivel2.py e imprime un mensaje de bienvenida y un mensaje de inicio. Luego simula una "carga" (un spinner/animación) y pregunta si quieres reintentar. Al introducir n el programa termina.

```
user2@retopython:~/Documentos$ python3 nivel2.py
Bienvenido al panel de administración.
Iniciando el motor de búsqueda
Parece que el servicio está detenido. ¿Quieres volver a intentarlo? y/n: n
```

- Leemos el código del script y el comentario **# La pass de user3 es N0C0mm3nts!** contiene explícitamente la contraseña de user3.

```
user2@retopython:~/Documentos$ python3 nivel2.py
Bienvenido al panel de administración.
Iniciando el motor de búsqueda
Parece que el servicio está detenido. ¿Quieres volver a intentarlo? y/n: n

user2@retopython:~/Documentos$ cat nivel2.py

import time
```

```

def loading():
    signs = ["|", "/", "-", "\\"]
    for i in range(100):
        print(signs[i%4].format(i%4), end='\r')
        time.sleep(0.1)

def menu():
    print("Bienvenido al panel de administración.")
    print("Iniciando el motor de búsqueda")
    loading()

opcion = ''
# La pass de user3 es N0C0mm3nts!
while opcion != 'n':
    menu()
    opcion = input("Parece que el servicio está detenido. ¿Quieres volver a intentarlo? y/n: ")

```

### user3 → user4

- Cambiamos a user3 y listamos el directorio Documentos:

```

user2@retopython:~/Documentos$ su user3
Password:

user3@retopython:/home/user2/Documentos$ cd /home/user3/Documentos

user3@retopython:~/Documentos$ ll
total 12
drwxr-xr-x  2 user3 user3 4096 ene 26 2023 ./
drwxr-x--- 14 user3 user3 4096 jun  9 18:10 ../
-rwrxr-x---  1 user3 user3  549 ene 26 2023 nivel3.py*

```

- Ejecutamos el script nivel3.py. El script pide un **nombre de usuario** y una **contraseña**, y muestra "El usuario no existe" cuando los valores introducidos no coinciden con los esperados.

```

user3@retopython:~/Documentos$ python3 nivel3.py
Bienvenido al panel de administración.
Iniciando el motor de búsqueda
Introduce el nombre de usuario: user3
Introduce la pass: 123456
El usuario no existe

```

- Leemos el código fuente del script:

```

user3@retopython:~/Documentos$ cat nivel3.py
import requests
import base64

def decode():
    base64_message = 'SDRyZEMwZGU='
    message_bytes = base64.b64decode(base64_message)
    message = message_bytes.decode('ascii')
    return message

def menu():
    print("Bienvenido al panel de administración.")
    print("Iniciando el motor de búsqueda")

```

```

menu()

user = input("Introduce el nombre de usuario: ")
passw = input("Introduce la pass: ")

if (user == "devadmin") and (passw == "H4rdC0ding&F4il"):
    print("La pass de user4 es: " + decode())
else:
    print("El usuario no existe")

```

- Contiene una cadena codificada en base64. Para decodificarla el usuario debe introducir la pass H4rdC0ding&F4il implicita en la condición if.
- Ejecutamos de nuevo el script nivel3.py con la credencial H4rdC0ding&F4il y muestra la contraseña de user4.

```

user3@retopython:~/Documentos$ python3 nivel3.py
Bienvenido al panel de administración.
Iniciando el motor de búsqueda
Introduce el nombre de usuario: devadmin
Introduce la pass: H4rdC0ding&F4il
La pass de user4 es: H4rdC0de

```

#### user4 → user5

- Cambiamos a user4 y listamos el directorio Documentos:

```

user3@retopython:~/Documentos$ su user4
Password:

user4@retopython:/home/user3/Documentos$ cd /home/user4/Documentos

user4@retopython:~/Documentos$ ll
total 12
drwxr-xr-x  2 user4 user4 4096 ene 26 2023 ./
drwxr-x--- 14 user4 user4 4096 jun  9 18:10 ../
-rwxr-x---  1 user4 user4  655 ene 26 2023 nivel4.py*

```

- Leemos el código fuente del script nivel4.py

```

user4@retopython:~/Documentos$ cat nivel4.py
import sys
import time
import base64

def decode():
    base64_message = 'UEByYW1ldDNycw=='
    message_bytes = base64.b64decode(base64_message)
    message = message_bytes.decode('ascii')
    return message

def loading():
    signs = ["|", "/", "-", "\\"]
    for i in range(40):
        print(signs[i%4].format(i%4), end='\r')
        time.sleep(0.1)

```

```

def menu():
    print("Bienvenido al panel de administración.")
    print("Iniciando el motor de búsqueda")
    loading()

args = sys.argv
menu()

if (len(args)%2 == 0) and (args[4] == 'debug'):
    print("La pass de user5 es: " + decode())

else:
    print("Parece que el servicio está detenido. Vuelve a intentarlo")

```

- La parte de código relevante es el bloque if-else. La condición del if determina que si el número de argumentos es par y el argumento en la posición 4 es 'debug', se muestra la contraseña de user5. En caso contrario muestra un mensaje de error.
- Si ejecutamos directamente sin argumentos muestra error puesto que no se está cumpliendo la condición if:

```

user4@retopython:~/Documentos$ python3 nivel4.py
Bienvenido al panel de administración.
Iniciando el motor de búsqueda
Parece que el servicio está detenido. Vuelve a intentarlo

```

- Por tanto ejecutamos de nuevo cumpliendo la condición (nivel4.py también cuenta como argumento) y obtenemos la contraseña para user5:

```

user4@retopython:~/Documentos$ python3 nivel4.py a b c debug d
Bienvenido al panel de administración.
Iniciando el motor de búsqueda
La pass de user5 es: P@ramet3rs

```

## user5 → user6

- Cambiamos a user5, accedemos a Documentos y listamos su contenido:

```

user4@retopython:~/Documentos$ su user5
Password:
user5@retopython:/home/user4/Documentos$ cd /home/user5/Documentos
user5@retopython:~/Documentos$ ll
total 12
drwxr-xr-x  2 user5 user5 4096 ene 26  2023 ./
drwxr-x--- 14 user5 user5 4096 jun  9 18:10 ../
-rw-r-x---  1 user5 user5   591 ene 26  2023 nivel5.py*

```

- Revisamos el código fuente del script nivel5.py:

```

user5@retopython:~/Documentos$ cat nivel5.py
import requests
import base64

def decode():
    base64_message = 'RXh0M3JuYWxLM1k='
    message_bytes = base64.b64decode(base64_message)
    message = message_bytes.decode('ascii')
    return message

```

```

response = requests.get('https://gist.githubusercontent.com/poundifdef/fd0901799a4aeb3a1f3956cfdb3c7746/raw/36f07f44ba9f80eb93fbb336e2bdf4096adef484/trollface.txt')
key = str(response.text).split('\n')[-2].strip()

user_response = str(input("¿Cómo puedo ayudarte? \n"))

if user_response == key:
    print("La pass de user6 es: " + decode())
else:
    print("No puedo ayudarte con eso")
El código de nivel5.py es:

```

- El código necesita una clave para decodificar la contraseña de user6 cifrada en base64. Para ello remite a un archivo de texto ubicado en una URL (trollface.txt) del cual debemos extraer la clave para decodificar la contraseña de user6. Dicha clave se obtiene cuando divide el texto en líneas separadas con saltos de línea (.split('\n')) y toma la penúltima línea del texto [-2]:

-:o+/:...

- A continuación ejecutamos nivel5.py, introducimos como clave lo anterior y obtenemos la contraseña para user6:

```

user5@retopython:~/Documentos$ python3 nivel5.py
¿Cómo puedo ayudarte?
-:o+/:...
La pass de user6 es: Ext3rnalK3Y

```

## user6 → user7

- Cambiamos a user6, accedemos a Documentos y listamos su contenido:

```

user5@retopython:~/Documentos$ su user6
Password:

user6@retopython:/home/user5/Documentos$ cd /home/user6/Documentos

user6@retopython:~/Documentos$ ll
total 16616
drwxr-xr-x  2 user6 user6   4096 jun  9 15:46 .
drwxr-x--- 14 user6 user6   4096 jun  9 18:10 ../
-rw-r-x---  1 user6 user6  1044 ene 26 2023 nivel6.py*
-rw-rw-r--  1 user6 user6     0 jun  9 15:46 '.registered_users.d'$'\247''b'
-rw-rw-r--  1 user6 user6 16999983 jun 10 19:18 .registered_users.db

```

- El código de nivel6.py es el siguiente:

```

user6@retopython:~/Documentos$ cat nivel6.py
import os
import time
import base64

def decode():
    base64_message = 'PTBuZU0xbGxpB24='
    message_bytes = base64.b64decode(base64_message)
    message = message_bytes.decode('ascii')
    return message

```

```

def loading():
    signs = ["|", "/", "-", "\\"]
    for i in range(100):
        print(signs[i%4].format(i%4), end='\r')
        time.sleep(0.1)

def register_user(mail):
    file = open('.registered_users.db', 'a')
    file.write(mail+"\n")

def verify_win():
    file = open('.registered_users.db', 'r').readlines()
    return (len(file) == 1000000)

path = ".registered_users.db"

print("Bienvenido al concurso 1.000.000")
print("Registrate para ganar un fantástico premio. ¿Serás tu el participante 1.000.000?")

user = input("Introduce tu dirección de mail: ")
register_user(user)

print("Estamos verificando si eres el participante 1.000.000 ...")
loading()

if verify_win():
    print("Enhorabuena, has ganado!")
    print("La pass de user7 es: " + decode())
else:
    print("Lo sentimos, no eres el participante 1.000.000")

```

- El programa decodifica la contraseña de user7 cifrada en base64, solo si al introducir el correo electrónico comprueba que se trata del usuario 1 millón. Esto lo hace comprobando en el archivo .registered\_users.db.
- Si se ejecuta el programa directamente, verifica si somos el participante 1.000.000 y al no serlo muestra el mensaje correspondiente:

```

user6@retopython:~/Documentos$ python3 nivel6.py
Bienvenido al concurso 1.000.000
Registrate para ganar un fantástico premio. ¿Serás tu el participante 1.000.000?
Introduce tu dirección de mail: pepito@perez.com
Estamos verificando si eres el participante 1.000.000 ...
Lo sentimos, no eres el participante 1.000.000

```

- Por tanto, hay que interactuar de algún modo con el archivo .registered\_users.db para que cuando compruebe seamos el participante 1 millón. Para ello usamos la siguiente sentencia que “añade” 999.999 líneas al archivo y cuando introduzcamos nuestro email sera la línea 1 millón. De esta forma conseguimos la contraseña para user7.

```

user6@retopython:~/Documentos$ yes "pepito@perez.com" | head -n 999999 > .registered_users.db

user6@retopython:~/Documentos$ python3 nivel6.py
Bienvenido al concurso 1.000.000
Registrate para ganar un fantástico premio. ¿Serás tu el participante 1.000.000?
Introduce tu dirección de mail: pepito@perez.com
Estamos verificando si eres el participante 1.000.000 ...
Enhorabuena, has ganado!

```

**La pass de user7 es: =0neM1llion**

## user7 → user8

- Cambiamos a user7, accedemos a Documentos y listamos su contenido. Hay dos scripts: uno cliente y otro servidor:

```
user6@retopython:~/Documentos$ su user7
Password:
user7@retopython:/home/user6/Documentos$ cd /home/user7/Documentos
user7@retopython:~/Documentos$ ll
total 16
drwxr-xr-x  2 user7 user7 4096 ene 26  2023 ./
drwxr-x--- 14 user7 user7 4096 jun  9 16:18 ../
-rw xr-x---  1 user7 user7  307 ene 26  2023 nivel7_cli.py*
-rw xr-x---  1 user7 user7  951 ene 26  2023 nivel7_srv.py*
```

- Código fuente del script cliente: implementa un cliente de socket que se conecta a un servidor en la dirección 127.0.0.1 (localhost) y puerto 8050, y permite enviarle comandos para recibir respuestas.

```
user7@retopython:~/Documentos$ cat nivel7_cli.py
import socket

host = '127.0.0.1'
port = 8050

obj = socket.socket()
obj.connect((host, port))

print("Conectado al servidor")

while True:
    mens = input("Comando >> ")

    obj.send(mens.encode())

    respuesta = obj.recv(1024)

    print(respuesta.decode())

obj.close()

print("Conexión cerrada")
```

- Código fuente del script servidor: es un servidor socket escrito que implementa una lógica de autenticación tipo OTP (One-Time Password) para revelar una contraseña oculta en base64.

```
user7@retopython:~/Documentos$ cat nivel7_srv.py
import socket
import random
import base64

def getotp():
    return str(random.randint(10000000, 99999999))

def decode_pass():
    base64_string = "TGEgcGFzcyBkZSB1c2VyOCBlczogVENQIVMwY2szdA=="
    base64_bytes = base64_string.encode("ascii")
```

```

sample_string_bytes = base64.b64decode(base64_bytes)
sample_string = sample_string_bytes.decode("ascii")
return sample_string

conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.bind(("127.0.0.1", 8050))
conn.listen(1)
cli, addr = conn.accept()
otp = ""

while True:

    recibido = cli.recv(1024)

    print("Recibo conexion de la IP: " + str(addr[0]) + " Puerto: " +
str(addr[1]))

    comando = recibido.decode()

    if comando == "user":
        otp=getotp()
        cli.send(otp.encode())
    elif comando == otp:
        cli.send(decode_pass().encode())
    else:
        cli.send("404 Command not found".encode())

cli.close()
ser.close()

print("Conexiones cerradas")

```

- Por tanto, ejecutamos en dos terminales distintas, ambos programas.
- Servidor:

```

user7@retopython:~/Documentos$ python3 nivel7_srv.py
Recibo conexion de la IP: 127.0.0.1 Puerto: 46534
Recibo conexion de la IP: 127.0.0.1 Puerto: 46534

```

- Cliente: al ejecutarlo queda a la espera de que introduzcamos un comando. En este caso introducimos “user” para poder conseguir un OTP, que también introducimos a continuación, y de esta forma nos devuelve la contraseña para user8:

```

user7@retopython:~/Documentos$ python3 nivel7_cli.py
Conectado al servidor
Comando >> user
39233764
Comando >> 39233764
La pass de user8 es: TCP!S0ck3t

```

## user8 → user9

- Cambiamos a user8, accedemos a Documentos y listamos su contenido:

```

user7@retopython:~/Documentos$ su user8
Password:
user8@retopython:/home/user7/Documentos$ cd /home/user8/Documentos
user8@retopython:~/Documentos$ ls -la

```

```
total 12
drwxr-xr-x  2 user8 user8 4096 ene 26  2023 .
drwxr-x--- 14 user8 user8 4096 ene 26  2023 ..
-rw-rxr-x---  1 user8 user8 1722 ene 26  2023 nivel8.py
```

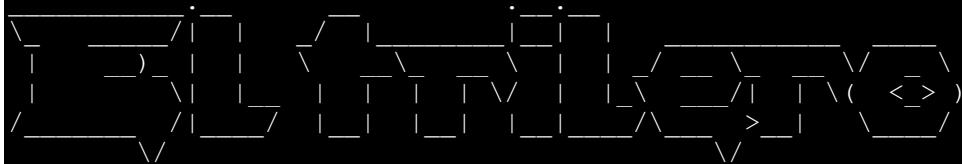
- Código fuente del script nivel8.py: es un minijuego de adivinanza en el que el usuario debe acertar la posición de una "bolita" escondida bajo uno de tres vasos, cinco veces seguidas. Si lo consigue, se le revela una contraseña oculta en Base64.

```
        else:
            win = win + 1
            print(sentence[random.randint(0,len(sentence))-1])

if win == 5:
    print("No se como lo has hecho pero has ganado...")
    print("Aquí tienes la pass de user9: ")
    loading()
    print(decode())
```

- La función `getNewPosition` toma como argumentos `timeexec` (que es la hora a la que se ejecuta el juego) y `round`. De esta forma, divide el string por `:`, obteniendo una lista tipo `["12", "34", "56"]` y selecciona uno de los tres elementos en función de `round % 3`, para que el índice siempre sea 0, 1 o 2. Convierte ese segmento de tiempo a entero. Le suma el número de ronda, hace `% 3` para obtener una posición válida entre 0 y 2 (ideal para ciclos de 3 elementos) y devuelve esa posición `+1` como string.

```
|user8@retopython:~/Documentos$ python3 nivel8.py
```



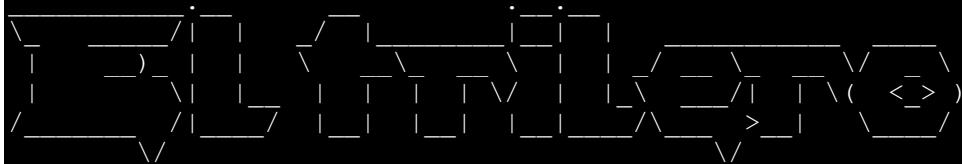
00:02:41  
¿Puedes adivinar dónde está la bolita 5 veces seguidas?  
Estamos reordenando los vasos  
Introduce el número del vaso (1, 2, 3):

- Teniendo en cuenta la hora de ejecución (timeexec) HH = 00, MM = 02 y SS = 41, obtenemos los resultados siguiente para superar el juego:

Ronda i	round % 3	base usada	Cálculo	Resultado
0	0	00	$(0 + 0) \% 3 = 0 \rightarrow 0 + 1 = 1$	1
1	1	02	$(2 + 1) \% 3 = 0 \rightarrow 0 + 1 = 1$	1
2	2	41	$(41 + 2) \% 3 = 1 \rightarrow 1 + 1 = 2$	2
3	0	00	$(0 + 3) \% 3 = 0 \rightarrow 0 + 1 = 1$	1
4	1	02	$(2 + 4) \% 3 = 0 \rightarrow 0 + 1 = 1$	1

- De esta forma, obtenemos la contraseña para user9:

```
|user8@retopython:~/Documentos$ python3 nivel8.py
```



00:02:41 ¿Puedes adivinar dónde está la bolita 5 veces seguidas?  
Estamos reordenando los vasos  
Introduce el número del vaso (1, 2, 3): 1  
Interesante...  
Estamos reordenando los vasos  
Introduce el número del vaso (1, 2, 3): 1  
No está mal para ser novat@  
Estamos reordenando los vasos  
Introduce el número del vaso (1, 2, 3): 2

```
No está mal para ser novat@  
Estamos reordenando los vasos  
Introduce el número del vaso (1, 2, 3): 1  
No está mal para ser novat@  
Estamos reordenando los vasos  
Introduce el número del vaso (1, 2, 3): 1  
Interesante...  
No se como lo has hecho pero has ganado...  
Aquí tienes la pass de user9:  
N0tS0Rand0m!
```

## user9 → user10

- Cambiamos a user9, accedemos a Documentos y listamos su contenido:

```
user8@retopython:~/Documentos$ su user9
Password:

user9@retopython:/home/user8/Documentos$ cd /home/user9/Documentos

user9@retopython:~/Documentos$ ls -la
total 16
drwxr-xr-x  2 user9 user9 4096 ene 26 2023 .
drwxr-x--- 14 user9 user9 4096 ene 26 2023 ..
-rwxrwx---  1 user9 user9  113 ene 26 2023 config.json
-rwxr-x---  1 user9 user9 1702 ene 26 2023 nivel9.py
```

- Código fuente del script nivel9.py:

```
user9@retopython:~/Documentos$ cat nivel9.py
import json
import time
from datetime import date

commands = {
    "low": ["time", "date", "status"],
    "medium": ["active_users", "applications"],
    "high": ["performance", "encoding", "stop service"]
}

def loading():
    signs = "| / - \\"[::]
    for i in range(40):
        print(signs[i%4].format(i%4), end='\r')
        time.sleep(0.1)

def get_privs():
    with open('config.json') as fp:
        config = json.load(fp)
        privs = config["privs"]
    return privs

def menu():
    print("Comandos disponibles:")
    num = 0
    for command in commands[get_privs()]:
        print("> " + str(num) + ". " + command)
        num = num + 1

def execute_low(command):
    if str(command) == "0":
        print(time.strftime("%H:%M:%S", time.localtime()))
    elif str(command) == "1":
```

```

        print(date.today().strftime("%d/%m/%Y"))
    else:
        print("Status: active")

def execute_medium(command):
    if str(command) == "0":
        print("User: user10 - Encoded pass: 6e6957267376697250796669646f4d")
    else:
        print("Web Application Server - Port 80")

def execute_high(command):
    if str(command) == "0":
        print("Disk: 65% RAM: 99.7% Network: 5% ")
    elif str(command) == "1":
        print("Hex(Reverse(password))")
    else:
        print("Service stopped")

print("Bienvenido al panel de administración.")
print("Iniciando el motor de búsqueda")
loading()

while True:
    menu()
    command = input("Elige opción: ")

    try:
        command = int(command)

        if (command < 0) or (command > len(commands[get_privs()])):
            print("Comando no implementado")
            break
        else:
            if get_privs() == "high":
                execute_high(command)
            elif get_privs() == "medium":
                execute_medium(command)
            else:
                execute_low(command)

    except ValueError:
        print("Se espera un número como opción")
        break

```

- Este código implementa un panel de administración simulado por consola, que muestra diferentes comandos según un nivel de privilegios (low, medium, high) obtenido desde un archivo config.json.

```

user9@retopython:~/Documentos$ cat config.json
{"deployment": {"files": ["config.json"]}, "id": "v1", "runtime": "python3", "threadsafe": true, "privs": "high"}

```

- Decodificamos primero la cadena de encoded pass y a continuación invertimos la cadena de texto, consiguiendo así la contraseña de user10:

```

user9@retopython:~/Documentos$ python3
Python 3.10.6 (main, Mar 10 2023, 10:55:28) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> bytes.fromhex("6e6957267376697250796669646f4d").decode()
'niW&svirPyfidoM'
>>>
>>> "niW'svirPyfidoM"[::-1]
"ModifyPrivs&Win"

```

```
>>>
```

## user10 → root

- Cambiamos a user10, accedemos a Documentos y listamos su contenido, pero se encuentra vacío.

```
user9@retopython:~/Documentos$ su user10
Password:

user10@retopython:/home/user9/Documentos$ cd /home/user10/Documentos

user10@retopython:~/Documentos$ ll
total 8
drwxr-xr-x  2 user10 user10 4096 ene 26 2023 ../
drwxr-x--- 14 user10 user10 4096 jun  9 17:06 ./
```

- Comprobamos que user10 pertenece al grupo sudo. Eso indica que tiene permisos para usar sudo. Por tanto, ejecutando sudo su e introduciendo la contraseña de user10 accedemos a la shell de root con privilegios totales y control completo del sistema.

```
user10@retopython:~/Documentos$ groups user10
user10 : user10 sudo

user10@retopython:~/Documentos$ sudo su
[sudo] password for user10:

root@retopython:/home/user10/Documentos#
```

## 5. Resultados

El reto fue completado con éxito, logrando la ejecución de comandos como **root** tras la resolución de 10 niveles. Se aplicaron técnicas de:

- Ingeniería inversa de código Python.
- Análisis lógico de condiciones.
- Decodificación y manipulación de datos.
- Simulación de entornos de red (cliente/servidor).
- Escalada final mediante privilegios de usuario sudo.

## 6. Aviso legal

- Este reto se realizó **únicamente con fines educativos y en un laboratorio virtual controlado**.
- No se ha vulnerado ningún sistema real.
- Los usuarios, contraseñas y direcciones IP empleados son **ficticios** y no representan datos reales.