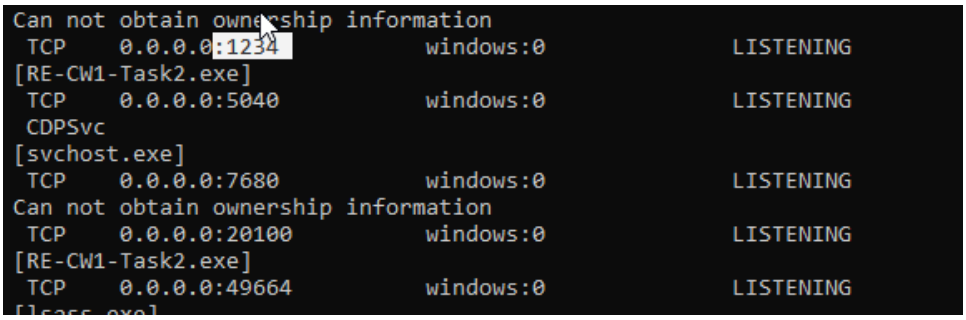
Introduction

The task involved reverse engineering an executable file (RE-CW1-Task2.exe) to understand its functionality and to uncover hidden features or operations. This report details the methodologies and tools used, the step-by-step analysis process, and the key findings.

Tools Used

- **Procmon (Process Monitor)**
- **Ghidra**
- Wireshark
- cmd

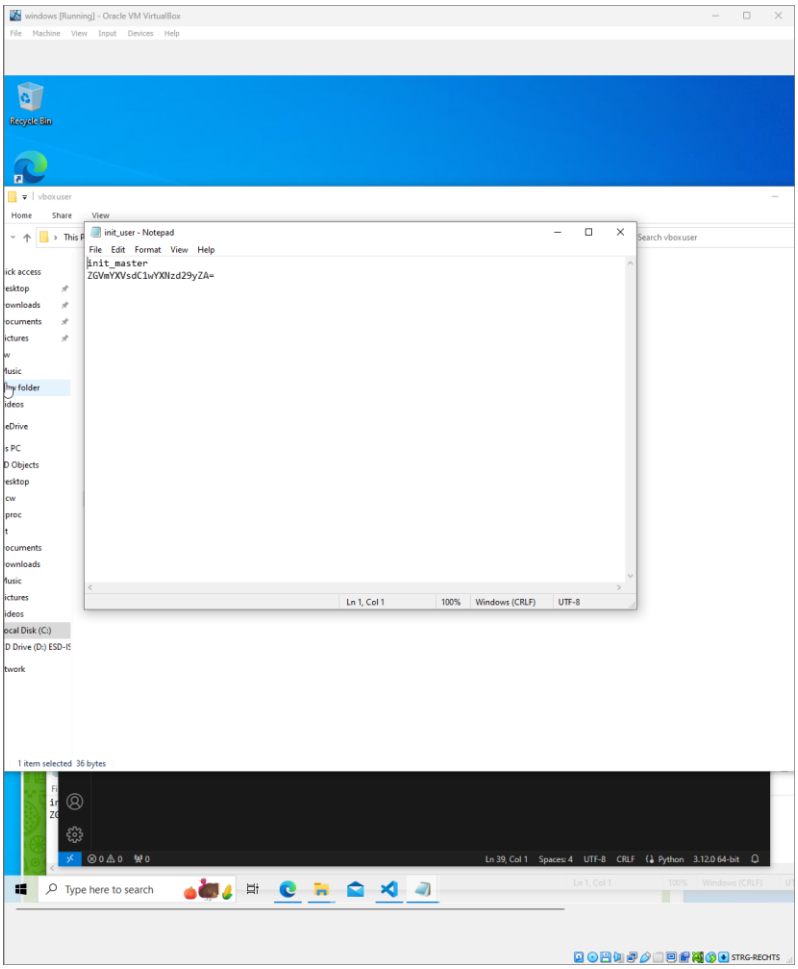
Netstat –ab



Analysis Process and Findings

- **Initial Examination with Procmon:**

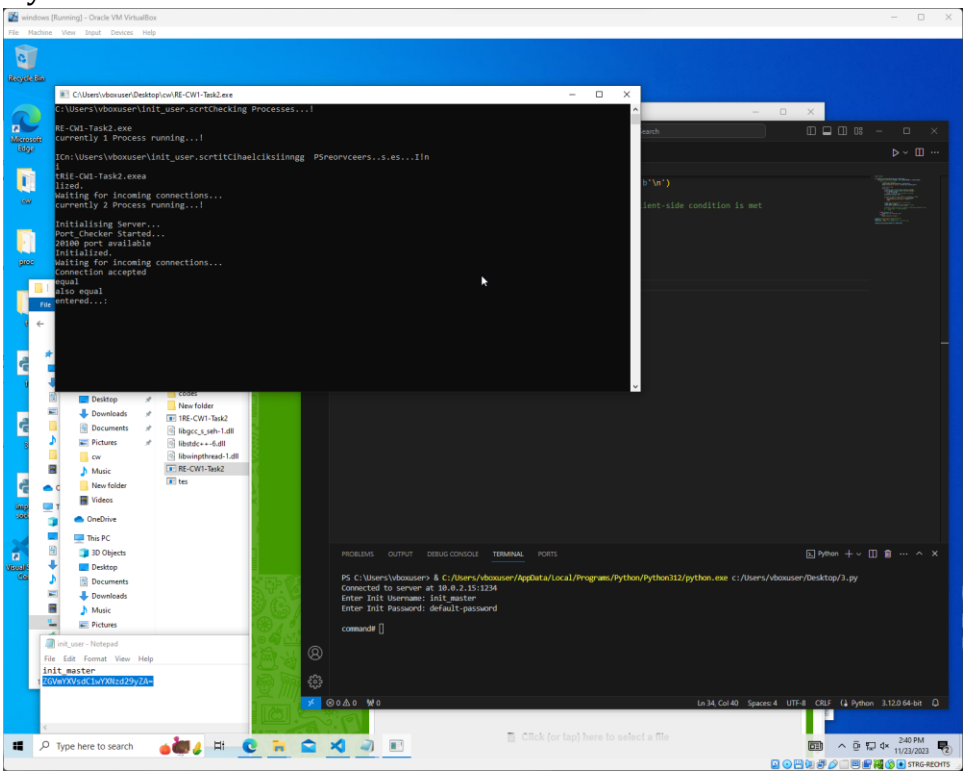
On launching the .exe, Procmon captured the file's interactions with the system. Noticed the creation of 'init_user', a file holding critical user information. The file revealed a username ('init_master') and an encoded password.

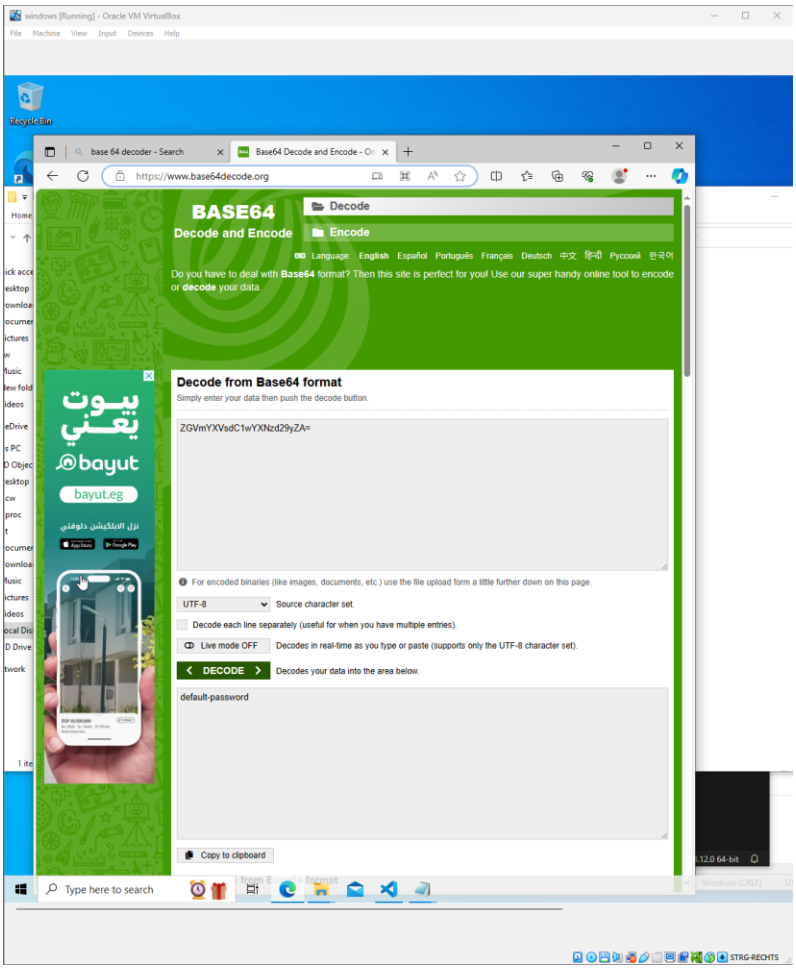


- **Password Decoding and Server Connection:**

Applied base64 decoding to the password, revealing it as 'default-password'. Accessed a console using these credentials, indicating a successful breach into the system's initial

layer.


C:\Users\vboxuser\Desktop\cw\RE-CW1-Task2.exe
C:\Users\vboxuser\init_user.scrtChecking Processes...!
RE-CW1-Task2.exe
currently 1 Process running...!
Waiting for incoming connections...
currently 2 Process running...!
Initialising Server...
Port_Checker Started...
20100 port available
Initialized.
Waiting for incoming connections...
Connection accepted
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\Users\vboxuser> & C:/Users/vboxuser/AppData/Local/Programs/Python/Python312/python.exe c:/Users/vboxuser/Desktop/3.py
Connected to server at 10.0.2.15:1234
Enter Init Username: init_master
Enter Init Password: default-password
command#
init_user - Notepad
init_master
Type here to search



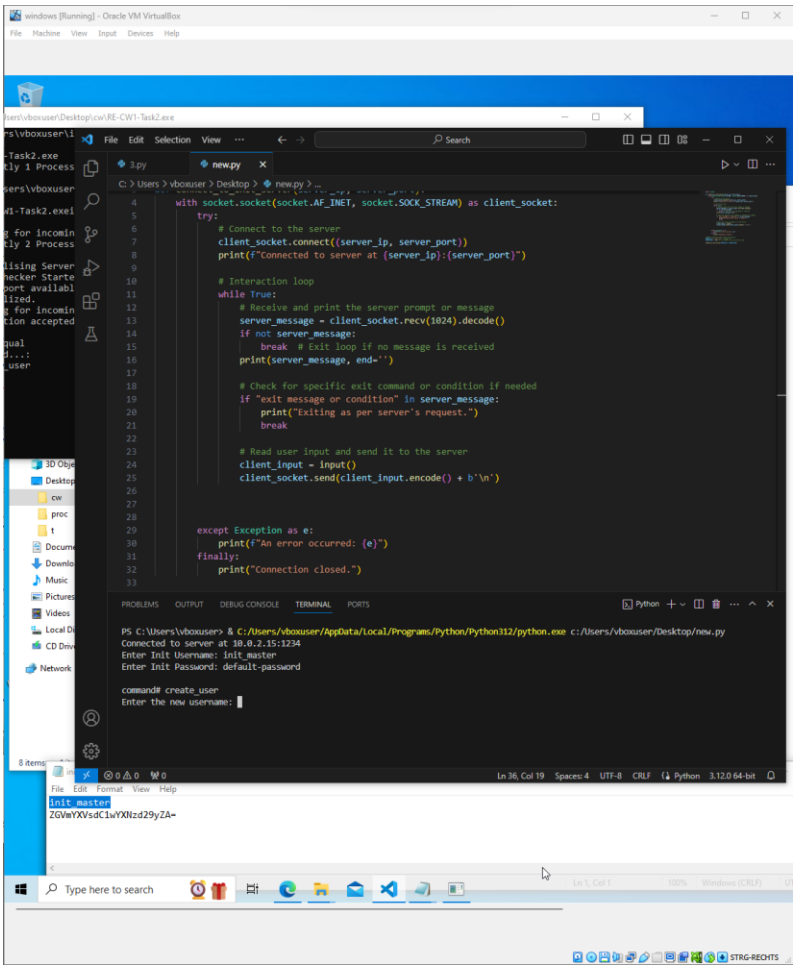
- **Deep Dive into the Source Code with Ghidra:**

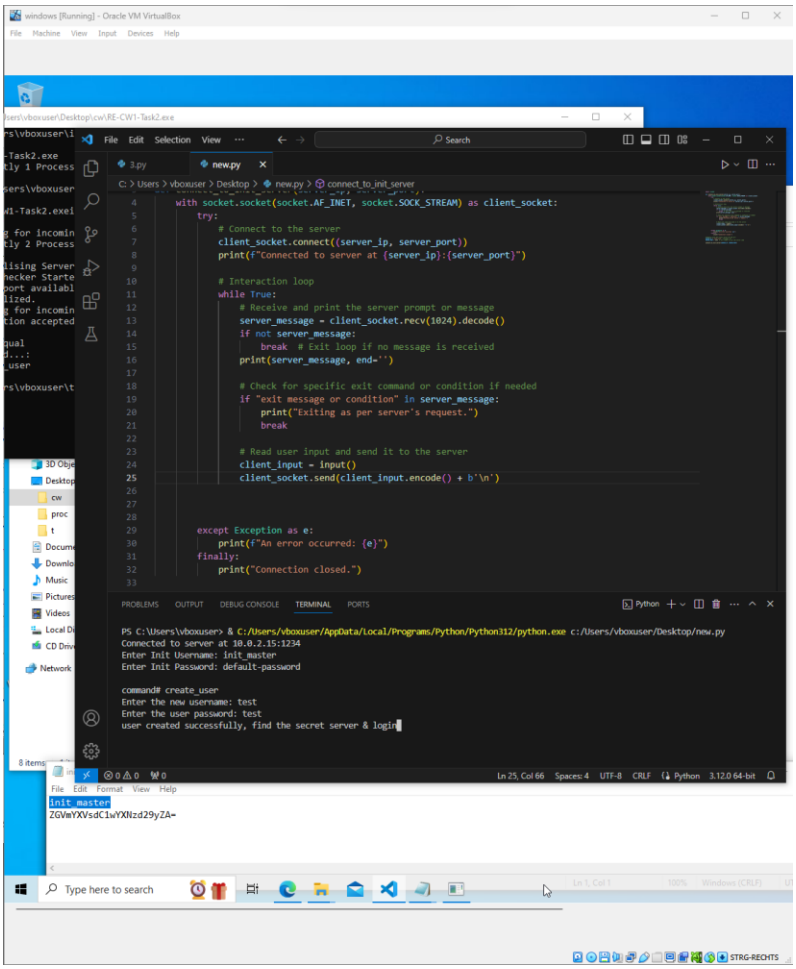
Searched for clues in the .exe's source code using Ghidra.
Identified two significant functions: 'get_hostname' and 'create_user', hinting at the server's expected commands.
This analysis was pivotal in understanding the executable’s internal command structure.

```
if (iVar1 != 0) {
  pbVar3 = std::operator<<((basic_ostream *)&std::cout,"entered...:");
  std::basic_ostream<>::operator<<((basic_ostream<> *)pbVar3,std::endl<>);
  do {
    memset(recvbuf,0,0x200);
    sVar2 = strlen("\ncommand# ");
    send(s_01,"\ncommand# ",(int)sVar2,0);
    iResult = recv(s_01,recvbuf,0x200,0);
    pbVar3 = std::operator<<((basic_ostream *)&std::cout,recvbuf);
    std::basic_ostream<>::operator<<((basic_ostream<> *)pbVar3,std::endl<>);
    iVar1 = strcmp(recvbuf,"get_hostname\n");
    if (iVar1 == 0) {
      get_hostname(in_stack_ffffffffffffac8);
      sVar2 = strlen(host);
      send(s_01,host,(int)sVar2,0);
      memset(recvbuf,0,0x200);
    }
    else {
      iVar1 = strcmp(recvbuf,"create_user\n");
      if (iVar1 == 0) {
        memset(uname,0,0x41);
        memset(pass,0,0x41);
        sVar2 = strlen("Enter the new username: ");
        send(s_01,"Enter the new username: ",(int)sVar2,0);
        recv(s_01,uname,0x41,0);
        sVar2 = strlen("Enter the user password: ");
        send(s_01,"Enter the user password: ",(int)sVar2,0);
        iResult = recv(s_01,pass,0x41,0);
        iVar1 = create_user(in_stack_ffffffffffffac8,in_stack_ffffffffffffad0);
        if (iVar1 != 0) {
          sVar2 = strlen("user created successfully, find the secret server & login");
          send(s_01,"user created successfully, find the secret server & login",(int)sVar2,0);
          closesocket(s_01);
          WSACleanup();
          return 1;
        }
        sVar2 = strlen(
                    "user wasn\'t created, You need to create user then access the secret serv
                    --"
```

- **Interactive Server Commands Execution:**

Executed the 'create_user' function, leading to an interactive prompt for new user credentials. Completion of this step resulted in a message, indicating progression: "User created successfully, find the secret server."


windows [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
File Edit Selection View
Search
3.py
new.py
C: > Users > vboxuser > Desktop > new.py > ...
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
try:
# Connect to the server
client_socket.connect((server_ip, server_port))
print(f"Connected to server at {server_ip}:{server_port}")
# Interaction loop
while True:
# Receive and print the server prompt or message
server_message = client_socket.recv(1024).decode()
if not server_message:
break # Exit loop if no message is received
print(server_message, end='')
# Check for specific exit command or condition if needed
if "exit message or condition" in server_message:
print("Exiting as per server's request.")
break
# Read user input and send it to the server
client_input = input()
client_socket.send(client_input.encode() + b'\n')
except Exception as e:
print(f"An error occurred: {e}")
finally:
print("Connection closed.")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
PS C:\Users\vboxuser> & C:/Users/vboxuser/AppData/Local/Programs/Python/Python312/python.exe c:/Users/vboxuser/Desktop/new.py
Connected to server at 10.0.2.15:1234
Enter Init Username: init_master
Enter Init Password: default-password
command# create_user
Enter the new username:
Ln 36, Col 19 Spaces: 4 UTF-8 CRLF Python 3.12.0 64-bit
File Edit Format View Help
init_master
ZGVmYXVsdC1wYXNzd29yZA==
Type here to search
STRG-RECHTS

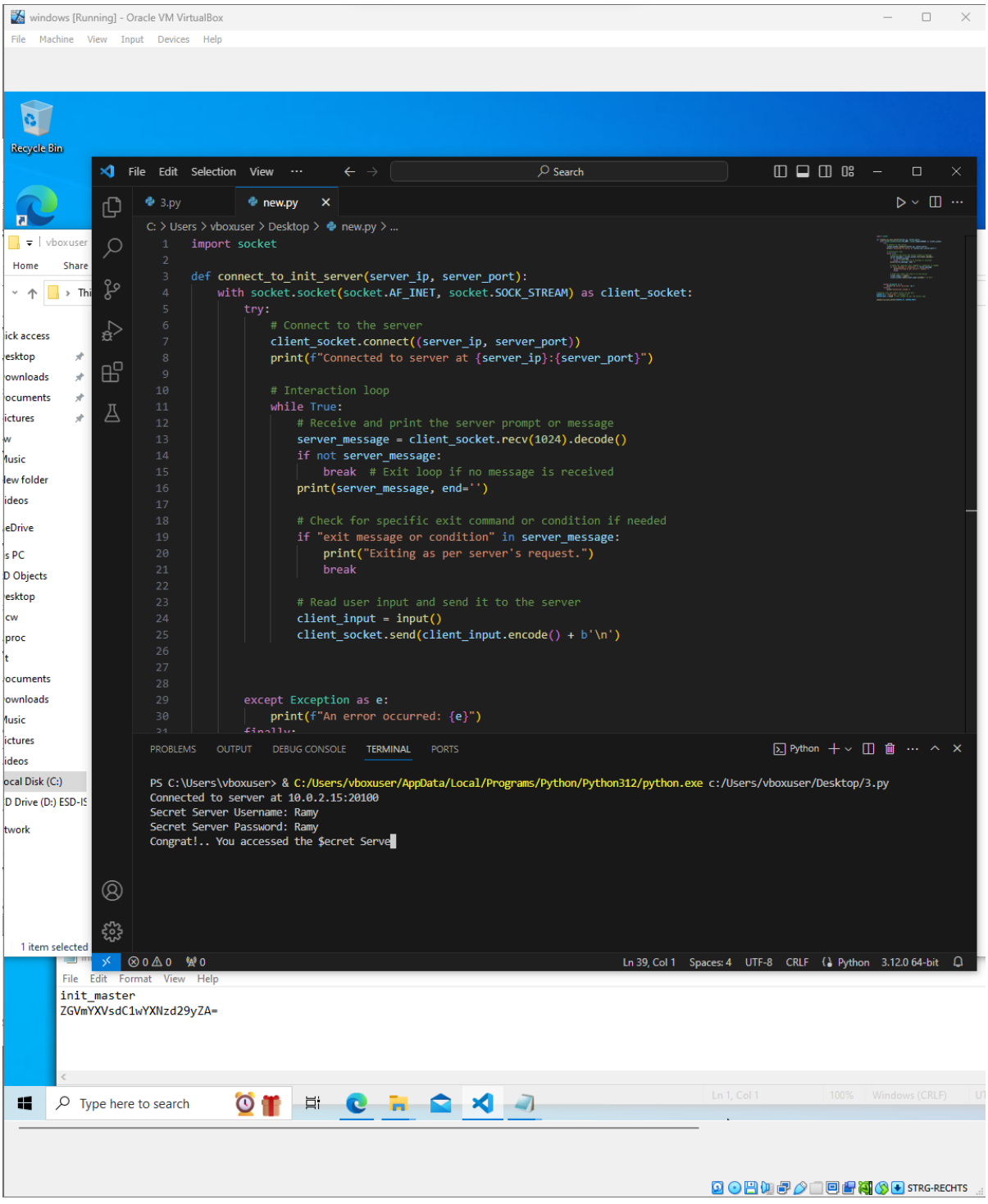


- **Uncovering and Accessing the Secret Server:**

Modified the network connection port to 20100, targeting the secret server's access point.

- **Final Stage and Accessing the Secret Server:**

Entered the newly created username and password. Successfully accessed the secret server, culminating in receiving the message: "Congrats! You accessed the $ecret Server."


windows [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Recycle Bin
File Edit Selection View
Search
3.py
new.py
C: > Users > vboxuser > Desktop > new.py > ...
import socket
def connect_to_init_server(server_ip, server_port):
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
try:
# Connect to the server
client_socket.connect((server_ip, server_port))
print(f"Connected to server at {server_ip}:{server_port}")
# Interaction loop
while True:
# Receive and print the server prompt or message
server_message = client_socket.recv(1024).decode()
if not server_message:
break # Exit loop if no message is received
print(server_message, end='')
# Check for specific exit command or condition if needed
if "exit message or condition" in server_message:
print("Exiting as per server's request.")
break
# Read user input and send it to the server
client_input = input()
client_socket.send(client_input.encode() + b'\n')
except Exception as e:
print(f"An error occurred: {e}")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
PS C:\Users\vboxuser> & C:/Users/vboxuser/AppData/Local/Programs/Python/Python312/python.exe c:/Users/vboxuser/Desktop/3.py
Connected to server at 10.0.2.15:20100
Secret Server Username: Ramy
Secret Server Password: Ramy
Congrat!.. You accessed the $ecret Serve
Ln 39, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.0 64-bit
File Edit Format View Help
init_master
ZGVmYXVsdC1wYXNzd29yZA=
Type here to search
STRG-RECHTS