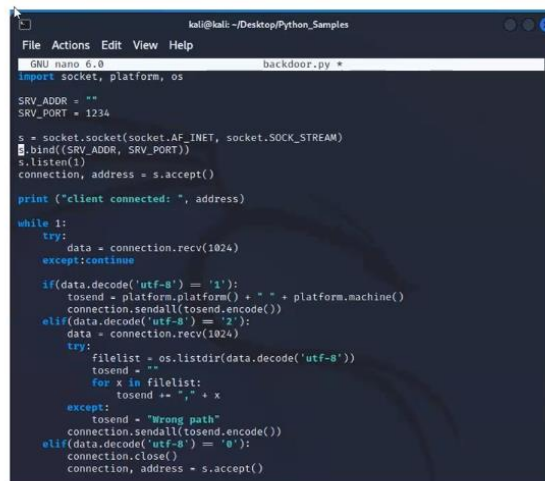


L'esercizio di oggi consiste nel commentare/spiegare questo codice che fa riferimento ad una backdoor.

Inoltre spiegare cos'è una backdoor.



```
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()
```

Una backdoor è letteralmente una “porta sul retro”, essa ci permette di aggirare le procedure di sicurezza attivate in un sistema informatico. Possono essere iniettate dai cracker che desiderano manomettere un sistema, o possono essere installate in modo autonomo da alcuni malware (trojan, worm etc), così da permettere l’accesso ad un utente esterno e prendere il controllo remoto della macchina. In sostanza si tratta di script malevoli in PHP che riescono a bypassare le misure di sicurezza di un server.

Importazioni: il codice importa tre moduli standard di Python: socket per la comunicazione di rete; platform per ottenere informazioni sul sistema operativo, e os per le operazioni di sistema.

Definizione delle costanti: vengono definite due costanti SRV\_ADDR e SRV\_PORT, queste rappresentano l’indirizzo ip del server e la porta su cui il server è in ascolto.

Creazione del socket e avvio dell’ascolto: viene creato l’oggetto socket usando Ipv4 (AF\_INET) e il protocollo TCP (SOCK\_STREAM). Il socket viene associato all’indirizzo e alla porta specificati e l’ascolto avviene con una coda massima di 1 connessione in attesa.

Accettazione della connessione: il server accetta la connessione in entrata. Quando la connessione è stabilita vengono restituiti un oggetto “connection” e l’indirizzo del client. Quindi viene stampato un messaggio che indica che un client è stato connesso.

Gestione delle richieste: Il server entra in un loop per gestire le richieste dal client. Si tenta di ricevere dati dalla connessione, e in caso di errore il loop continua.

Analisi delle richieste e risposte: Il codice esegue diverse azioni in base ai dati ricevuti dal client; se il dato è 1 invia al client una stringa che rappresenta la piattaforma e l’architettura del sistema; se il dato è 2 cerca la lista dei file nella directory specificata dal client e invia l’elenco al client. Se il percorso è errato invia il messaggio “Wrong path”. Se il dato è 0 chiude la connessione e accetta una nuova connessione.