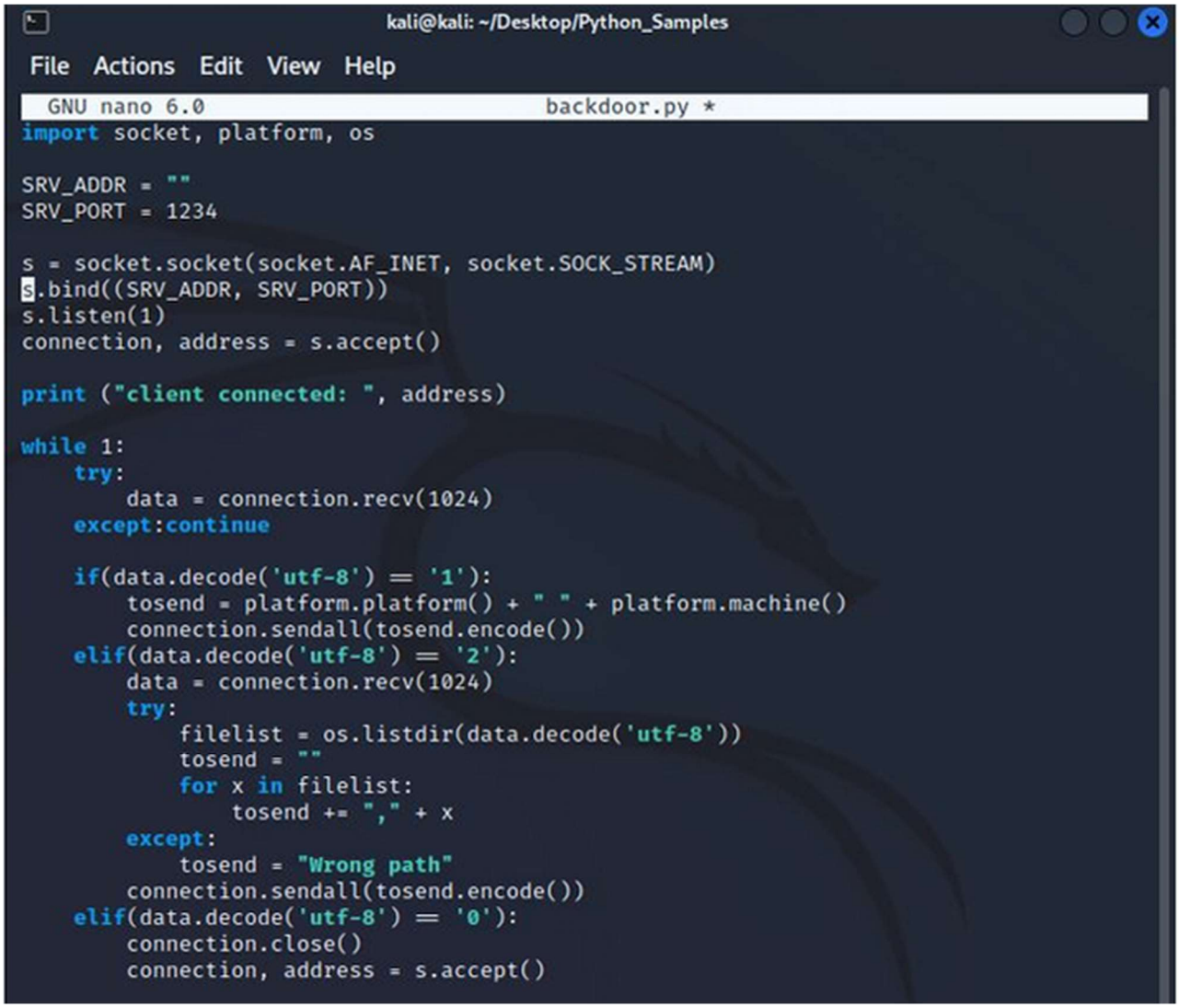

Questo programma esegue una serie di azioni su una ipotetica backdoor che stiamo sfruttando una volta creato un socket.

Per prima cosa vengono importati i moduli **socket**, **platform** e **os** permettendoci di utilizzare tutte le funzioni associate.

Creiamo due variabili: SRV_ADDR con l'indirizzo IP del PC dove vogliamo mettere il servizio in ascolto e SRV_PORT una con una porta che poi useremo per la connessione.

socket.socket crea un nuovo socket (che salviamo nella variabile s); la funzione accetta dei parametri, il primo AF_INET specifica che vogliamo un socket che utilizzi IPv4, il secondo SOCK_STREAM specifica che vogliamo una connessione TCP.

Dopo aver creato un nuovo socket, dobbiamo collegare il socket all'indirizzo e la porta che abbiamo specificato; questa azione prende il nome di binding. Utilizziamo il metodo **bind**, accedendo al nuovo socket s creato; la funzione accetta come parametri l'indirizzo IP e la porta.

listen() configura il socket per ascoltare sulla coppia IP:PORTA che abbiamo indicato; il numero tra parentesi indica il numero massimo di connessioni in coda.

Il metodo **accept** si usa per accettare e stabilire la connessione con il client che tenterà di connettersi al server; restituisce due argomenti: connection, ossia l'identificativo della comunicazione che verrà utilizzato per scambiare i dati e address, l'IP del client che si collegherà.

Una volta che il client si è connesso possiamo far partire lo scambio dati; lo facciamo con un ciclo while sempre vero (1 in Python equivale a vero, quindi il ciclo while in figura è un ciclo infinito).
connection.recv(1024) è utilizzato per ricevere i dati dal client specificando la dimensione del buffer e salvandoli nella variabile data. Se tale operazione (try) non riesce (except) si ritorna all'inizio del while (continue).

A questo punto vi è un **if** con varie condizioni di controllo sulla variabile data decodificata (i dati del client arrivano codificati in formato utf-8 e vanno pertanto decodificati):

Nel caso di valore uguale a 1 nella variabile tosend salviamo informazioni riguardo il sistema operativo e il tipo di macchina; **sendall** viene usato per inviare a tutti i partecipanti alla connessione tali informazioni codificate nel formato utf-8 (default se le parentesi sono vuote).

Nel caso di valore uguale a 2 riceviamo nuove informazioni dal client e le salviamo nuovamente nella variabile data; proviamo (try) a creare una lista di file con **os.listdir** nel path specificato da data, salvando in tosend tale lista di file ognuno separato dalla virgola e inviando a tutti i partecipanti alla connessione tale lista sempre codificata, altrimenti (except) verrà inviato un messaggio 'Wrong path'.

Nel caso di valore uguale a 0 la connessione viene chiusa e si rimane in attesa di una nuova connessione, ricominciando il ciclo while una volta stabilita.

Una **backdoor** (dal termine inglese per porta di servizio o porta sul retro) è un metodo, spesso segreto, per passare oltre (aggirare, bypassare) la normale autenticazione in un sistema informatico
Le backdoor sono spesso scritte in diversi linguaggi di programmazione e hanno la funzione principale di superare le difese imposte da un sistema, come può essere un firewall, al fine di accedere in remoto ad un personal computer, ottenendo per mezzo di un sistema di crittografia un'autenticazione che permetta di prendere il completo o parziale possesso del computer vittima.
Una backdoor può celarsi segretamente all'interno di un ignaro programma di sistema, di un software separato, o può anche essere un componente hardware malevolo come apparati di rete, sistemi di sorveglianza e alcuni dispositivi di infrastruttura di comunicazione che possono avere celate al loro interno backdoor maligne permettendo l'intrusione di un eventuale criminale informatico.