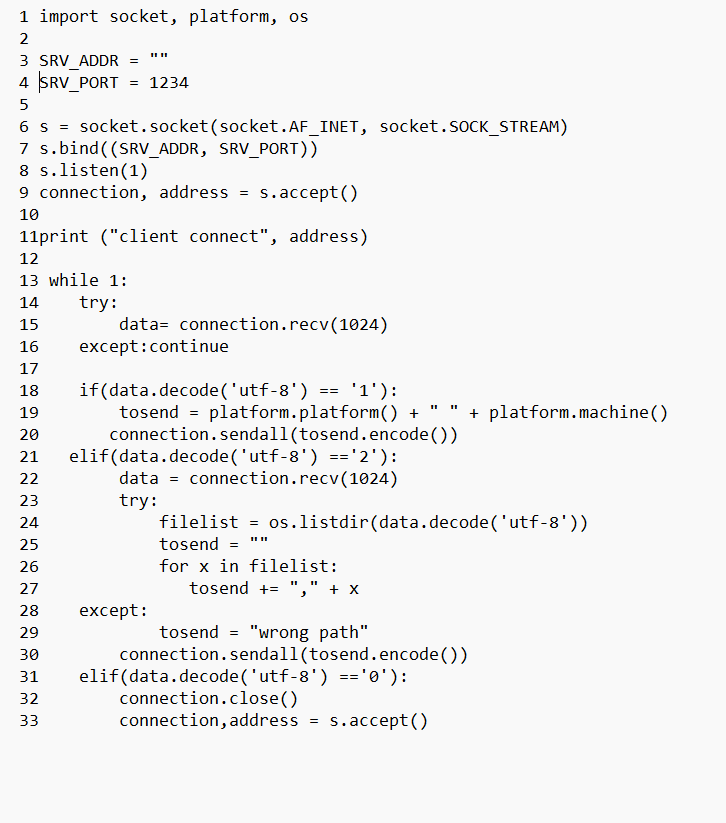
**S3 L4 STEFANO CESARONI**

Una backdoor è una tecnica informatica che consente l'accesso non autorizzato o nascosto a un sistema, applicazione o rete. Solitamente, una baackdoor è un metodo segreto o un'entrata secondaria che bypassa i normali controlli di accesso o autenticazione. Questo può consentire a un attaccante di ottenere un accesso privilegiato al sistema, senza che gli utenti autorizzati o i meccanismi di sicurezza rilevino la presenza dell'attaccante. Le backdoor possono essere create intenzionalmente da sviluppatori di software per scopi di manutenzione o di supporto remoto, ma possono anche essere inserite da malintenzionati come parte di attacchi informatici per sfruttare il sistema in futuro. Possono essere implementate a livello di software, hardware o di rete e sono molto dannose in quanto consentono agli attaccanti di mantenere l'accesso al sistema senza essere scoperti per lunghi periodi di tempo, permettendo loro di rubare informazioni sensibili, danneggiare il sistema o usarlo come punto di partenza per altri attacchi. È importante proteggere i sistemi da potenziali backdoor mediante l'implementazione di misure di sicurezza robuste, come l'aggiornamento regolare del software, l'uso di firewall e antivirus.



**RIGA 1)** Per prima cosa si importano i moduli **socket, platform, os:**

* Il modulo socket consente di creare socket utilizzando la funzione socket.socket().
* Il modulo platform consente di ottenere informazioni sulla piattaforma (sistema operativo e hardware) su cui il programma Python viene eseguito. Questo modulo fornisce una serie di funzioni utili per accedere a queste informazioni per esempio platform.platform() ci restituisce una stringa descrittiva della piattaforma, che può includere il nome del sistema operativo, la versione e altre informazioni pertinenti.

-Il modulo os fornisce una serie di funzioni che consentono di accedere a funzionalità specifiche del sistema operativo. Questo include funzioni per la gestione dei file, la manipolazione del percorso del file system, l'esecuzione di comandi del sistema e la gestione dei processi.

**RIGA 3)** Si crea una variabile chiamata SRV\_ADDR dove andremo ad inserire l’indirizzo IP;

**RIGA 4)** Si crea una variabile chiamata SVR\_PORT dove andremo ad inserire il numero della porta;

**RIGA 6)** Con la funzione socket.socket si va a creare un nuovo sockets. socket.AF\_INET specifica che vogliamo un socket che utilizzi IPV4, socket.SOCK\_STREAM specifica che vogliamo una connesione TCP.

**RIGA 7)** Dopo aver creato il socket è necessario collegarlo alla porta che abbiamo specificato: s.bind indica il metodo “bind” che associa un socket ad un IP:Porta, ((SRV\_ADDR, SRV\_PORT)) indicano rispettivamente l’indirizzo IP e la porta.

**RIGA 8)** Il metodo 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.

**RIGA 9)** Il metodo accept() accetta e stabilisce la connessione con il client che tenterà di connettersi al nostro servizio. Ci restituirà 2 argomenti:

* Connection è il socket che utilizzeremo per scambiare i dati;
* Address è l’IP del client che si connetterà.

**RIGA 11)** Con la funzione print il programma ci restituirà in output il “client connect” e “address”.

**RIGA 13)** Il programma entra in un ciclo while.

**RIGA 15)**  connection.recv(1024) è utilizzato per ricevere i dati dal client mentre il numero 1024 ne indica la grandezza in byte.

**RIGA 17-18)** Una volta ricevuti i dati vengono codificati usando data.decode('utf-8'). In base al contenuto dei dati il programma esegue una determinata azione. Se riceve while '1', otterrò informazioni sul sistema e sulla macchina che verranno contenute nella variabile “tosend” .Platform.platform() restituisce una stringa descrittiva della piattaforma su cui viene eseguito il programma. Questa stringa può includere il nome del sistema operativo, la versione e altre informazioni pertinenti. Platform.machine(): restituisce una stringa che identifica la macchina.

**RIGA 19)** Questa istruzione invia i dati contenuti nella variabile” tosend” al client tramite la connessione.

* **connection.sendall():** Questo metodo invia i dati codificati sulla connessione.
* **tosend.encode():** Prima di inviare i dati attraverso la rete, è necessario codificarli in un formato che possa essere trasferito. In questo caso, la stringa contenuta nella variabile **“tosend”** viene codificata in un array di byte utilizzando il metodo **encode().** Questo metodo converte la stringa in un array di byte utilizzando lo schema di codifica UTF-8.

**RIGA 20-27)** Questa parte del codice gestisce il caso in cui si riceve while 2 . In questa situazione, si aspetta che il client invii un percorso di directory come dati aggiuntivi. Riceviamo i dati aggiuntivi inviati dal client utilizzando **connection.recv(1024)**. Una volta ricevuti i dati, li decodifichiamo utilizzando **data.decode('utf-8')** per ottenere il percorso della directory richiesta dal client. Si utilizza la funzione **os.listdir()** per ottenere l'elenco dei file nella directory specificata dal client. La funzione restituisce una lista di nomi di file e directory presenti nella directory specificata Il risultato di tutto ciò è una stringa che verrà memorizzata nella variabile tosend, che verrà inviata al client come risposta per il comando '2'.

In sintesi, questo processo consente al client di richiedere un elenco dei file presenti in una specifica directory sul server e di ricevere una risposta con l'elenco dei file.

**RIGA 28-30**) Questa parte del codice gestisce l'eccezione che potrebbe verificarsi nel caso in cui il percorso della directory fornito dal client non sia valido o non esista. In caso di eccezione la riga di codice **tosend = "wrong path**", indica che il percorso fornito non è valido.

**RIGA 31-33)** Questa parte del codice gestisce il caso in cui si riceve while'0' . In questo caso, si chiude la connessione corrente con il client e si attende una nuova connessione in ingresso da parte di un altro client.

