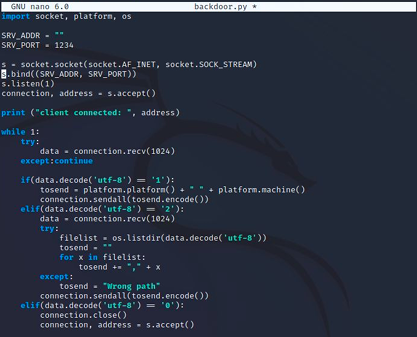
S3/L4

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

Una backdoor letteralmente “*porta sul retro*”, sono righe di codice grazie alle quali una persona può entrare come amministratore all’interno di dispositivi come personal computer, ottenendo per mezzo di un sistema di crittografia un'autenticazione che permetta di prendere il completo o parziale possesso del computer di un’altra persona. Come è facilmente intuibile le backdoor danno la possibilità ad un black hat di accedere da remoto al sistema informatico della vittima su cui vengono montate, che avrà così intero controllo del sistema. Quando le backdoor diventano, come abbiamo detto un qualcosa di illegittimo, prendono più propriamente il nome di “rat” (remote access **trojan)**.

Vediamo ora in cosa consiste il seguente codice:



Alla prima riga troviamo l’importazione di 3 moduli:

Socket 🡪 un programma che funge da server, progettato per la trasmissione e la ricezione di dati attraverso una rete; è il punto in cui il codice di un processo accede al canale di comunicazione per mezzo di una porta ottenendo una comunicazione tra processi che lavorano su due macchine diverse. Sono composti da due parti: indirizzo IP e il numero di porta.

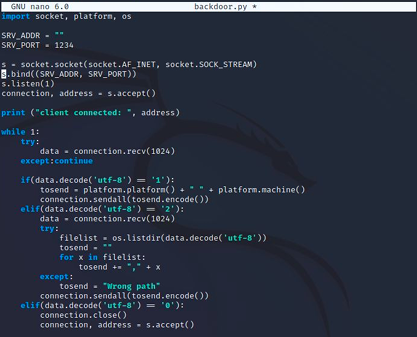
Platform 🡪 è il modulo che mi permette di leggere le caratteristiche hardware e la versione di python da un programma.

Os 🡪 questo modulo ha diverse funzione utili per far interagire il programma con il sistema operativo del computer (per esempio cancellare o rinominare un file, cambiare i permessi in una cartella, trovare l’estensione di un file…)

Dopo di che troviamo altre due righe di codice, rispettivamente:

SRV\_ADDR = è il server address il quale contiene l’’indirizzo IP del server in ascolto

SRV\_PORT = contiene il numero di porta su cui il server ascolta le connessioni in entrata, in questo caso: 1234



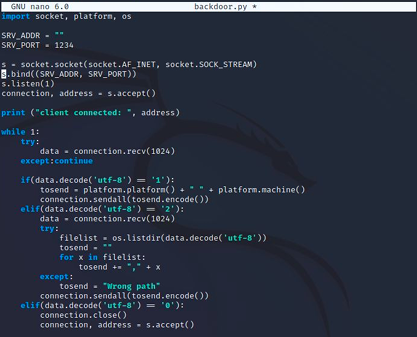
s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) 🡪 crea un socket utilizzando socket.socket(). Il primo parametro socket.AF\_INET specifica che si tratta di un socket di tipo IPv4, mentre il secondo, socket.SOCK\_STREAM specifica che si tratta di un socket TCP (protocollo che si occupa del controllo della trasmissione e della comunicazione dati in rete). Questa riga di codice è una dicitura fissa.

s.bind((SRV\_ADDR, SRV\_PORT)) 🡪 associa l'indirizzo IP alla porta specificati con bind(). In questo modo, il server è in ascolto su quell'indirizzo IP e porta per le connessioni in ingresso.

s.listen(1) 🡪 mette il socket in modalità "ascolto" utilizzando listen(). Il parametro 1 indica che il server può accettare una sola connessione in entrata alla volta. E’ il primo parametro di sicurezza del DOS.

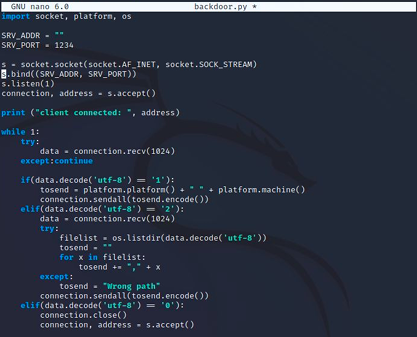
connection, address = s.accept() 🡪 accetta una connessione in entrata. “Connection” è il nuovo socket creato per comunicare con il client, e contiene il suo indirizzo IP e porta.

print("Client connected", address) 🡪 stampa un messaggio per indicare che un client è stato connesso e visualizza l'indirizzo del client.



Questa parte di codice, con il while loop è il cuore del programma:

si entra nel ciclo while per ricevere dati infiniti dal client, con una dimensione di 1024 byte, ovvero quanti byte riesce a recepire il server. Questa si trova all’interno dell’istruzione try except:continue; questa serve ad evitare che il programma vada in errore. Permette dunque di intercettare uno o più errori nell’esecuzione di un blocco di istruzioni tramite la gestione delle eccezioni. Permette al programmatore in caso di imprevisto di correggere automaticamente senza bloccare l’esecuzione del programma.



Troviamo ora la parte di codice di un costrutto if – elif, il quale viene utilizzato per la selezione di un blocco di istruzioni da eseguire in base al verificarsi di una determinata condizione.

If 🡪 Se la condizione data.decode(utf-8’) è uguale a 1 viene inviata una risposta contenente informazioni sulla piattaforma e sulla macchina del sistema attraverso la stessa connessione. Questa sequenza di dati sta venendo decodificata utilizzando ‘utf-8’ in quanto è lo standard ed il modo in cui si ricevono i dati in una connessione di rete.

Elif 🡪 Rappresenta una condizione alternativa rispetto alla precedente. Se i dati ricevuti sono decodificati come ‘2’ si esegue questo blocco di istruzioni. Vengono ricevuti ulteriori dati, tentando di ottenere la lista dei file nella directory specificata e inviando i nomi dei file attraverso la connessione. Con ‘tosend’ viene inizializzata una stringa vuota per preparare il messaggio da inviare, viene creato un messaggio concatenando i nomi dei file separati da virgole. Il blocco ‘except’ viene eseguito se si verifica un’eccezione, questo caso non è possibile ottenere la lista dei file poiché il percorso è errato = wrong path.

Elif 🡪 l’ultima condizione viene verificata quando i dati ricevuti sono decodificati come 0. Se questa condizione è vera, viene eseguito questo blocco di istruzioni; in sintesi questa parte del codice chiude la connessione corrente e accetta una nuova connessione utilizzando ‘s.accept’. In questo caso per esempio il client segnala come 0 il segnale per terminare la connessione corrente e prepararsi per una nuova.