Pratica S3-L4

Una **backdoor** si riferisce a un mezzo nascosto o non documentato per ottenere l'accesso a un sistema o una rete. Si tratta di un modo per aggirare i normali meccanismi di autenticazione o di sicurezza, consentendo a individui non autorizzati di ottenere il controllo su un sistema. Le backdoor possono essere create intenzionalmente dagli sviluppatori di software per scopi di debug o amministrativi, ma possono anche essere inserite in modo dannoso dagli aggressori per sfruttare le vulnerabilità di un sistema.

Il codice fornito sembra essere un script Python che imposta una backdoor di base utilizzando i socket.

I **socket di rete** sono utilizzati nelle reti di computer e permettono di scambiare pacchetti (dati) tra due computer, una sorgente ed un destinatario.

**Importazione dei Moduli:**

****

Questa riga importa i moduli necessari di Python: socket per la gestione delle comunicazioni di rete e **platform** per ottenere informazioni sul sistema.

il modulo **socket** fornisce un'interfaccia di rete, mentre **os** consente l'accesso alle funzioni del sistema operativo.

**Impostazione dell'Indirizzo e della Porta del Server:**

SRV\_ADDR = ""

SRV\_PORT = 1234

Queste righe definiscono l'indirizzo del server e il numero di porta. Possiamo sostituire **""** con l'effettivo indirizzo IP o hostname dove si desidera far funzionare il server.

**Creazione di un Socket:**

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

*This line creates a new socket object using IPv4 (AF\_INET) and TCP (SOCK\_STREAM) for communication.*

**Associazione del Socket:**

s.bind((SRV\_ADDR, SRV\_PORT))

Il socket viene associato all'indirizzo e alla porta del server specificati.

**Ricevere la Connessioni:**

s.listen(1)

Il socket inizia ad ricevere le connessioni in entrata, con un massimo di 1 connessione in attesa.

**Accettazione di una Connessione:**

connection, address = s.accept()

print("Client connected:", address)

Lo script attende e accetta una connessione in entrata. s.accept() si blocca finché non viene stabilita una connessione. Una volta connesso, stampa l'indirizzo del client.

**Gestione dei Dati in un Ciclo:**

while True:

try:

data = connection.recv(1024)

except:

continue

Lo script entra in un ciclo infinito in cui cerca continuamente di ricevere dati dal client. Se si verifica un'eccezione durante la ricezione dei dati, il programma passa alla successiva iterazione.

**Elaborazione dei Dati Ricevuti:**

if data.decode('utf-8') == "1":

tosend = platform.platform()

connection.sendall(tosend.encode())

Se i dati ricevuti sono "1", vengono recuperate le informazioni sulla piattaforma usando platform.platform() e inviate nuovamente all’utente

elif data.decode('utf-8') == "2":

data = connection.recv(1024)

try:

filelist = os.listdir(data.decode("utf-8"))

tosend = "\n".join(filelist)

except:

tosend = "Wrong path"

connection.sendall(tosend.encode())

Se i dati ricevuti sono "2", ci si aspetta di ricevere un percorso di directory dal client. Viene quindi tentato di elencare i file in quella directory e la lista (o un messaggio di errore) viene inviata nuovamente all’utente.

elif data.decode('utf-8') == "0":

connection.close()

connection, address = s.accept()

Se i dati ricevuti sono "0", viene chiusa la connessione corrente e si attende una nuova connessione chiamando s.accept().