

```
Clone di Clone di Kali Linux Clone [Running]
Applications Places Terminal Dec 5 16:57

kali@kali: ~
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)-[~]
$ python backdoor.py
client connected: ('192.168.32.100', 40664)
Traceback (most recent call last):
  File "/home/kali/backdoor.py", line 30, in <module>
    connection.sendall(tosend.encode())
    ^^^^^^^^^^^^^^^^^
AttributeError: 'str' object has no attribute 'encode'. Did you mean: 'encode'?

kali@kali: ~
$ python client_backdoor.py
Type the server IP address: 192.168.32.100
Type the server port: 1234
Connection established

0) Close connection
1) Get system info
2) List directory contents

-Select an option: 1
Linux-6.3.0-kali1-amd64-x86_64-with-glibc2.37 x86_64

-Select an option: 2
Insert the path:1
```

S3/L2

Il compito di oggi verte sulla creazione di una `backdoor.py` e un `client_backdoor.py`. Come si evince dalla figura, ho creato sulla stessa macchina virtuale Kali entrambi i codici dove hanno interagito.

Se avessi voluto provarla su un altro clone sempre Kali o su un'altra piattaforma avrei dovuto cambiare IP dove la backdoor avrebbe mantenuto l'originale andando ad intercettare l'IP del `client_backdoor`.

Di seguito troviamo le funzionalità dei due codici:

backdoor.py (Server Backdoor)

Funzionalità:

Accetta connessioni in entrata da client.

Dopo la connessione, entra in un loop per ricevere comandi dal client.

Supporta i seguenti comandi:

'1°': Invia informazioni sulla piattaforma e sulla macchina al client.

'2°': Riceve un percorso dal client, elenca i file nella directory specificata e invia l'elenco al client.

'0°': Chiude la connessione con il client.

Differenze Chiave:

Utilizza una struttura di controllo **while True** per rimanere in ascolto delle connessioni e gestire comandi continuamente.
Riceve comandi dal client e risponde di conseguenza.
Invia informazioni sulla piattaforma e sulla macchina, elenca i file nella directory specificata o chiude la connessione.

client_backdoor.py (Client Backdoor)

Funzionalità:

Chiede all'utente di inserire l'indirizzo IP del server e la porta per connettersi.

Presenta un menu numerato con le seguenti opzioni:

'0': Chiude la connessione.

'1': Invia il comando al server per ottenere informazioni sulla piattaforma e sulla macchina.

'2': Chiede all'utente di inserire un percorso e invia il comando al server per elencare i file nella directory specificata.

Differenze Chiave:

Chiede all'utente di selezionare un'opzione dal menu numerato.

Invia comandi al server in base alle scelte dell'utente.

Riceve e visualizza le risposte dal server.

Connessione tra Server e Client:

Avvio del Server:

backdoor.py viene avviato su un host che funge da server.

Esecuzione del Client:

client_backdoor.py viene eseguito su un altro host e si connette al server utilizzando l'indirizzo IP e la porta specificati.

Interazione:

L'utente può interagire con il menu del client, selezionare opzioni e inviare comandi al server.

Risposta del Server:

Il server processa i comandi ricevuti e invia le risposte al client.

Cautela nella Creazione di Backdoor:

Utilizzo Etico e Legale:

La creazione di backdoor deve essere guidata da scopi etici e legali.

Rispetto della Privacy:

L'utilizzo di backdoor deve rispettare la privacy e non violare leggi o regolamenti.

Responsabilità:

Gli sviluppatori devono essere responsabili e consapevoli delle implicazioni etiche nell'implementazione di backdoor.