HATHEBOX

EPICODE

S3-L4

TEAM 1 - ETHICAL HACKING

CONTENT

HATHEBOX

GOALS AND OBJECTIVES

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

```python
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 += "," + x
        except:
            tosend = "Wrong path"
        connection.sendall(tosend.encode())
    elif(data.decode('utf-8')=='0'):
        connection.close()
        connection, address = s.accept()
```

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

```python
SRV_ADDR = ""
SRV_PORT = 1234
```

```python
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((SRV_ADDR, SRV_PORT))
```

```python
s.listen(1)
connection, address = s.accept()
```

IMPORT LIBRERIE

- **socket**: Questa libreria fornisce le funzionalità di rete, necessarie per la comunicazione tramite socket.
- **platform**: Fornisce metodi per ottenere informazioni sul sistema operativo e sull'hardware.
- **os**: Fornisce funzionalità per l'interfacciamento con il sistema operativo, come l'accesso al filesystem.

DEFINIZIONI COSTANTI

- **SRV_ADDR**: L'indirizzo del server a cui il server si legherà per ascoltare le connessioni in arrivo. In questo caso, è vuoto, il che significa che il server accetterà connessioni su tutte le interfacce di rete disponibili.
- **SRV_PORT**: La porta su cui il server ascolterà le connessioni in arrivo.

CREAZIONE DEL SOCKET E ASSOCIAZIONE IP:PORT

- Viene creato un oggetto socket usando socket.socket().
- Il metodo bind() associa il socket all'indirizzo e alla porta specificati.

START SCOLTO CONNESSIONI / ACCETTAGIONE CONNESSIONE IN ARRIVO

- **listen()** mette il socket in modalità "passiva", permettendo al server di accettare connessioni in arrivo. L'argomento 1 specifica il numero massimo di connessioni in coda che possono essere accettate prima di rifiutare le nuove connessioni.
- **accept()** blocca il server finché non viene ricevuta una connessione in arrivo. Quando arriva una connessione, restituisce una tupla contenente un nuovo socket (connection) e l'indirizzo del client.

```
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 += "," + x
    except:
        tosend = "Wrong path"
    connection.sendall(tosend.encode())
elif(data.decode('utf-8')=='0'):
    connection.close()
    connection, address = s.accept()
```

```
connection.close()
```

COMUNICAZIONE CLIENT

- Il server entra in un ciclo while infinito per gestire la comunicazione con il client.
- recv() riceve i dati inviati dal client. L'argomento 1024 specifica la dimensione massima dei dati che possono essere ricevuti in byte.

ELABORAZIONE COMANDI RICEVUTI

- Il server decodifica i dati ricevuti dal client in formato UTF-8 e li confronta con i comandi previsti ('1', '2', '0').
- A seconda del comando ricevuto, il server risponde fornendo informazioni di sistema o elencando i file in una directory specificata dal client.

CHIUSURA DELLA CONNESSIONE

- Dopo aver completato la comunicazione con il client, la connessione viene chiusa.

CONCLUSIONS

sembra esserci una potenziale vulnerabilità che potrebbe essere utilizzata come backdoor. Nello specifico, il codice accetta comandi dal client senza autenticazione e esegue operazioni basate su questi comandi. Questo potrebbe consentire a un utente malintenzionato di eseguire comandi dannosi sul server senza autorizzazione.

Ecco alcuni punti che indicano la potenziale vulnerabilità:

1. *Nessuna Autenticazione: Il server non richiede alcuna forma di autenticazione per accettare i comandi dal client. Questo significa che qualsiasi client che conosca l'indirizzo e la porta del server può inviare comandi e interagire con il server.*
2. *Esecuzione di Comandi: Il server esegue comandi basati sui dati ricevuti dal client senza alcuna verifica o sanitizzazione. Ad esempio, il server può eseguire comandi di sistema operativo come elencare i file in una directory specificata dal client. Questo potrebbe essere sfruttato da un utente malintenzionato per eseguire comandi dannosi sul server.*

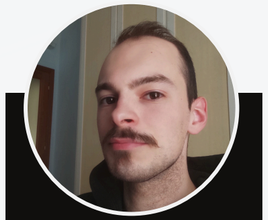
HATHEHACK

Una backdoor è una vulnerabilità o una funzionalità nascosta in un software o in un sistema informatico che consente a un utente non autorizzato di accedere al sistema, spesso bypassando le normali procedure di autenticazione e sicurezza. Essenzialmente, è un punto di accesso segreto che può essere sfruttato da un attaccante per ottenere un accesso non autorizzato al sistema.

OUR TEAM

Iosif
Castrucci
TL 1

Mario
Reitano
TEAM

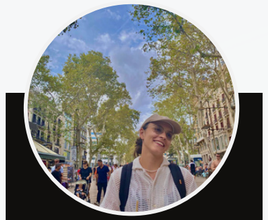


Giovanni
Sannino
TEAM 1

Andrea Di
Benedetto
TEAM 1

Luca Lenzi

TEAM 1



Mara Dello
Russo
TEAM 1

Morgan
Petrelli
TEAM 1