

Fichier `mysite/index.html`

1. Pendant que vous êtes connecté à dvwa, invoquez dans un autre onglet `mysite/index.html`. Déconnectez-vous de dvwa et reconnectez-vous : qu'observez-vous ? Expliquez.

password\_new=forceg&password\_conf=forceg&Change=Change# ». Lorsqu'on ouvre cette page web, le navigateur effectue une requête vers l'URL de l'image ce qui a pour effet de changer la page de DVWA. Le nouveau mdp est « forceg ».

connaître les infos de login de la victime ». Vrai, faux ? Justifiez.

Vrai, le pirate n'a pas besoin des infos de login de la victime, car il fait en sorte que la victime elle-même exécute les actions en se connectant sur la page.

### 3. Quelles sont les méthodes de protection implémentées dans les autres niveaux de sécurité ?

#### Niveau low :


```
Accept-Language: fr,fr-FR;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cache-Control: max-age=0
Connection: keep-alive
Content-Length: 47102
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryW4axCbaBLmAlWScu
Cookie: security=low; PHPSESSID=9474694f50dd7e11ea7cb980095da6f0
DNT: 1
Host: 10.157.14.145
Origin: http://10.157.14.145
Referer: http://10.157.14.145/dvwa/vulnerabilities/upload/
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.64 Safari/537.36 Edg/101.0.1210.53
```

#### Niveau medium :

**▼ Général**

**URL de la demande:** http://10.157.14.145/dvwa/vulnerabilities/csrf/?password\_new=test&password\_conf=test&Change=Change

**Méthode de demande:** GET

**Code d'état:**  200 OK

**Adresse distante:** 10.157.14.145:80

**Stratégie de point d'accès:** strict-origin-when-cross-origin

#### ▼ En-têtes de demande [Afficher la source](#)

**Accept:** text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.9  
**Accept-Encoding:** gzip, deflate

```
Accept-Language: fr,fr-FR;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cache-Control: max-age=0
Connection: keep-alive
Content-Length: 47102
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryAY6YyEb5IPdnrOi0
Cookie: security=medium; PHPSESSID=9474694f50dd7e11ea7cb980095da6f0
DNT: 1
Host: 10.157.14.145
Origin: http://10.157.14.145
Referer: http://10.157.14.145/dvwa/vulnerabilities/upload/
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.64 Safari/537.36 Edg/101.0.1210.53
```

Au niveau medium, il y a un champ HTTP Referer en plus.

Au niveau de sécurité High, il est nécessaire de connaître l'ancien mot de passe pour changer de mot de passe :

**Change your admin password:**

Current password:

New password:

Confirm new password:

Change

**Password Changed**

▼ Général

**URL de la demande:** http://10.157.14.145/dvwa/vulnerabilities/csrf/?password\_current=force&password\_new=test&password\_conf=test&Change=Change

**Méthode de demande:** GET

**Code d'état:** 200 OK

**Adresse distante:** 10.157.14.145:80

**Stratégie de point d'accès:** strict-origin-when-cross-origin

**Accept-Language:** fr,fr-FR;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6

**Cache-Control:** max-age=0

**Connection:** keep-alive

**Content-Length:** 47102

**Content-Type:** multipart/form-data; boundary=----WebKitFormBoundary1pHvld1PHcTzydy1

**Cookie:** security=high; PHPSESSID=9474694f50dd7e11ea7cb980095da6f0

**DNT:** 1

**Host:** 10.157.14.145

**Origin:** http://10.157.14.145

**Referer:** http://10.157.14.145/dvwa/vulnerabilities/upload/

**Upgrade-Insecure-Requests:** 1

**User-Agent:** Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.64 Safari/537.36 Edg/101.0.1210.53

## File Upload

Upload de oyster.jpeg

Choose an image to upload:

Aucun fichier n'a été sélectionné

../../../../hackable/uploads/oyster.jpeg successfully uploaded!

4. Où ce fichier est-il téléchargé dans l'arborescence ?

L'image uploadé se trouve dans le répertoire : `/var/www/dvwa/hackable/uploads`

Upload de simple.php :

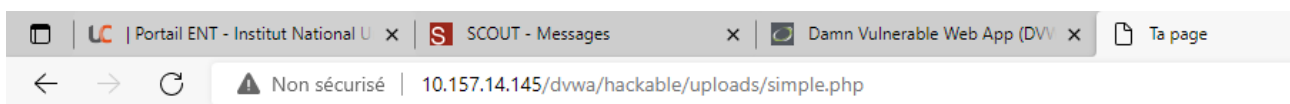
### Vulnerability: File Upload

Choose an image to upload:

Aucun fichier n'a été sélectionné

../../../../hackable/uploads/simple.php successfully uploaded!

On arrive bien à executer le fichier simple.php chargé sur le serveur web :

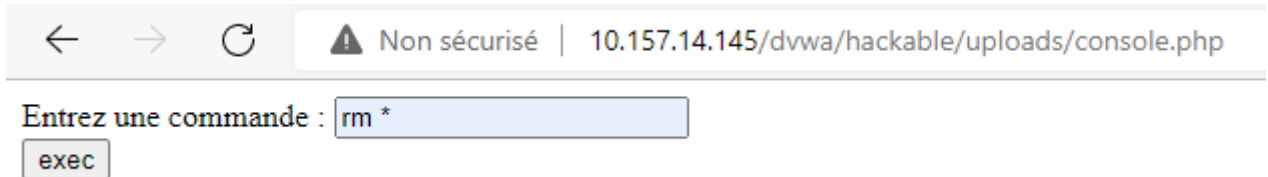


Laissez-moi entrer, laissez-moi !

5. Une fois le script chargé sur le serveur, pouvez-vous l'utiliser pour « nettoyer » le répertoire en cours ?

Si le compte associé au serveur web avait tous les droits, que pourriez-vous faire ?

En entrant la commande `rm *` dans le champ de saisie de la page `console.php`, on supprime le contenu du répertoire :



← → ↻ ⚠ Non sécurisé | 10.157.14.145/dvwa/hackable/uploads/console.php

Entrez une commande :

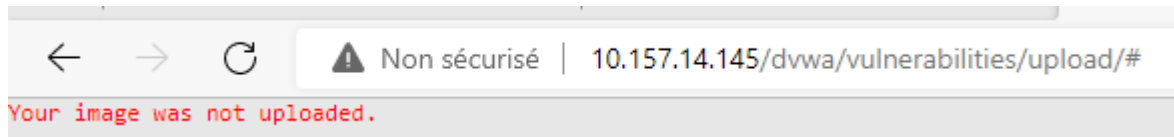
```
msfadmin@metasploitable:/var/www/dvwa/hackable/uploads$ ls
aaaa chestnut.gif chestnut.jpeg console.php dvwa_email.png eL0chestnut.jpeg oyster.jpeg simple.php test.html
msfadmin@metasploitable:/var/www/dvwa/hackable/uploads$ ls
msfadmin@metasploitable:/var/www/dvwa/hackable/uploads$
```

Si le compte associé au serveur web avait tous les droits, on pourrait effectuer des opérations sur tous les fichiers du système, voler des clés privées, certificats, exécuter des applications, installer des services, se configurer une prise en main à distance sur le serveur, etc.

6. Pendant que vous imaginez le pire scénario possible, le manager du site a défini des mesures de sécurité équivalentes au niveau *medium*. Après avoir appliqué cette configuration, pouvez-vous charger sur le serveur des images en .gif ? En .jpeg ? Des fichiers texte ? Du php ?

Avec un niveau de sécurité medium, le serveur :

- refuse le chargement de fichier .php, .gif, .txt
- accepte le chargement de fichier .jpeg



7. Comment ce filtrage est-il implémenté sur le serveur ?

Ce filtrage est implémenté en PHP via un filtre sur l'extension de fichier reconnu par PHP. Le script n'autorise que les images au format jpeg :

```
if(($uploaded_type == "image/jpeg") && ($uploaded_size < 100000)){  
  
    if(!move_uploaded_file($_FILES['uploaded']['tmp_name'], $target_path)) {  
  
        $html .= '<pre>';  
        $html .= 'Your image was not uploaded.';  
        $html .= '</pre>';  
  
    }else {  
  
        $html .= '<pre>';  
        $html .= $target_path . ' successfully uploaded!';  
        $html .= '</pre>';  
  
# systemctl status apache2  
# Apache2 web server  
}nit.d/apache2; generated; vendor preset: disabled)  
else{m/apache2.service.d  
if  
    echo '<pre>Your image was not uploaded.</pre>';  
})  
xsv-generator(8)
```

## Burpsuite

Changez, dans le corps de la requête, les trois premières lettres du nom du fichier. Recevez-vous un message d'erreur local ? Un message d'erreur du serveur distant ?

```
-----5781451958993420831360062238
Content-Disposition: form-data; name="uploaded"; filename="image.jpeg"
Content-Type: image/jpeg
```

8. Changez, dans le corps de la requête, les trois premières lettres du nom du fichier. Recevez-vous un message d'erreur local ? Un message d'erreur du serveur distant ?

J'ai changé les 3 premières lettres du fichier par KKK, le client obtient la réponse suivante :



Choose an image to upload:

No file selected.

../../hackable/uploads/KKKge.jpeg succesfully uploaded!

9. Revenez sur la page du script console.php et entrez la commande ls. Conclusion ?

Entrez une commande :

KKKge.jpeg console.php oyster.jpeg

En modifiant le corps de la requête sur BurpSuite, le nom du fichier enregistré par le serveur web a changé par rapport au nom du fichier envoyé par le client.

10. Comment pouvez-vous utiliser ces résultats pour charger un nouveau script sur le serveur ?

On pourrait uploader un script php sur la page d'upload, valider, intercepter la requête et changer la valeur de la balise Content-Type par « image/jpeg ». Ainsi le serveur pensera recevoir un fichier image alors que ce sera un fichier PHP

11. A partir de cette utilisation de Burp, expliquez pourquoi l'utilisation du champ HTTP\_REFERER n'est que le niveau medium de l'implémentation de CSRF.

HTTP\_REFERER est un champ optionnel du protocole HTTP qui indique l'agresseur d'origine de la page d'où provient la requête. Grâce au proxy on peut modifier la valeur de HTTP\_REFERER pour que le serveur PHP accepte l'image uploadée