Rapport Exercices Sécurité des Applications : Série 6 - suite

Thomas Dagier

December, 04, 2020

1 File upload, double extensions (UPLO1)

Dans cet exercice, nous devons uploader un code en php qui nous permet de récuperer le mot de passe. D'après la description du challenge, ce mot de passe se situe dans un fichier nommé : `.passwd` qui se situe à la racine de l'application.

En naviguant à travers les galleries disponibles, on remarque qu'un bouton nous permet d'uploader des fichiers aux formats .png, .gif et .jpg. Le soucis est que l'on ne peut pas lui demander d'uploader un fichier .php sinon on tombe sur une erreur qui nous dit que le format n'est pas supporté.

Il y a donc plusieurs problèmes que l'on doit pouvoir résoudre afin de trouver le mot de passe : - Le premier est de trouver comment faire passer un fichier php pour une image de sorte que, une fois uploadé, si on se place sur l'url de l'image, le code php soit éxécuté. - Le second problème est de réaliser ce code : comment faire pour afficher le contenu du fichier `.passwd`.

Ayant déjà fait du php pour des serveurs web auparavant, le second point n'est pas un problème mais il ne me sert à rien si je n'arrive pas à uploader le fichier. J'ai donc cherché sur google une manière de faire passe un fichier .php pour une image. Il s'avère que c'est très simple :

> *Does the attacker still possible to launch his php file eventhough I have renamed it to 'helloworld.txt'?*
>
> The first answer were:
>
> *If it's renamed to .txt then it won't run the PHP, so you should be fine, but double check to make sure it's not uploaded as .php.txt*

Figure 1: source : medium.com

Il suffit alors de créer un fichier `first_test.php.png` dans lequel on peut mettre notre code.

J'ai donc testé tout simplement d'envoyer un fichier avec le contenu php le plus simple possible :

```php
<?php
        echo "<pre> coucou les copains ! </pre>";
?>
```

Figure 2: fonction toute simple pour vérifier que le fichier est bien uploadé

On peut donc tester d'uploader le fichier pour voir le résultat :

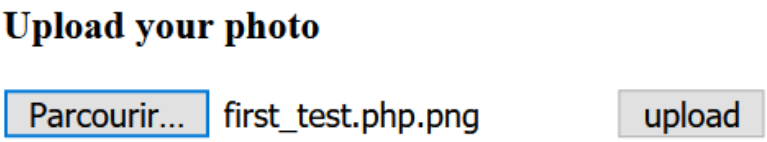

Figure 3: premier upload

Il est donc possible d'uploader le fichier ce qui ne nous met pas de message d'erreur puisque l'extension est effectivement celle d'une image .png :

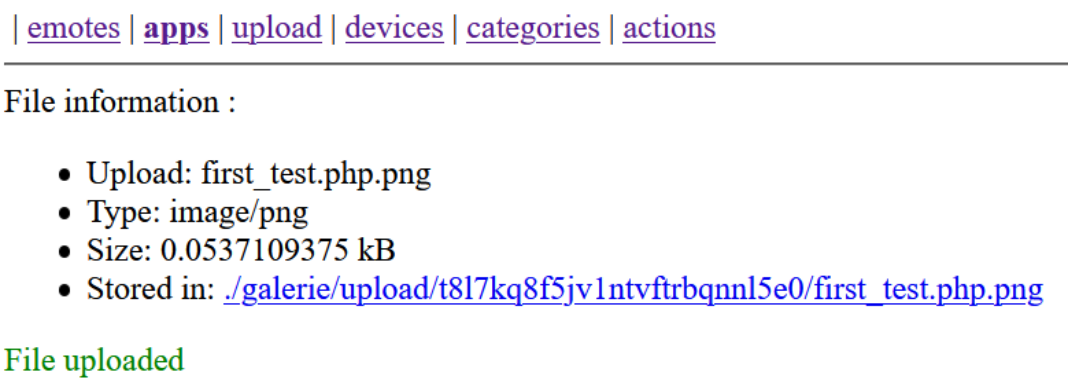

Figure 4: validation de l'upload

Puisque rien ne pose problème, on peut cliquer sur le lien pour aller vers l'image qui vient d'être uploadée :

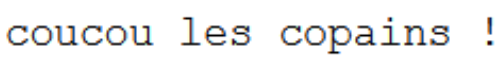

Figure 5: affichage du contenu de la page

On remarque qu'avec ce code php tout simple, on peut afficher du contenu sur ce qui est censé être une image. On peut donc théoriquement faire de même pour afficher le contenu d'un fichier. Cependant, il devient un peu plus compliqué de trouver exactement quoi afficher.
Sachant que j'avais déjà fait du php pour des serveurs web en stage et pour mon site, je connais la fonction shell_exec() accessible très facilement depuis le manuel php, qui permet d'éxécuter des commandes que l'on peut utiliser en shell. Cette fonction me permettais de manager des permissions pour les accès aux serveurs dans l'entreprise où j'étais en stage, il est donc possible de la réutiliser pour afficher le contenu d'un fichier.

Puisque l'on sait que le fichier contenant le mot de passe se trouve à la racine, son pwd est : `/.passwd`. Il nous suffit d'afficher sont contenu avec la commande cat et on se retrouve avec le code php :

```php
<?php
   $mdp = shell_exec('cat /.passwd');
   echo "<pre> $mdp </pre>";
?>
```

Figure 6: code php pour afficher le contenu du fichier .passwd

Il nous suffit de refaire les étapes précédentes puis afficher le contenu de la page qui est :

```
Gg9LRz-hWSxqqUKd77-_q-6G8
```

Figure 7: contenu du fichier

Le mot de passe trouvé, on peut le rentrer sur la page de départ pour obtenir les points du challenge :

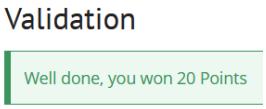

Figure 8: flag de validation

2 File upload, MIME type (UPLO2)

Dans le second exercice, nous devons trouve le mot de passe qui se trouve dans le même fichier. J'ai d'abord essayé de réutiliser la méthode du premier exercice. Au début, tout semblait fonctionner mais il s'avère qu'au moment de lancer le code php qui est censé être une image, plus rien ne marche.

Ceci est dû à la gestion des fichiers qui, cette fois-ci ne permet pas de lancer du code php de la même manière. Sur le forum de cet exercice, il est indiqué qu'il ne faut pas chercher à uploader un fichier png mais vraiment un fichier php. Le probblème qui se pose est que la base qui permet d'upload n'autorise toujours que les images .png, .jpeg et .gif, il faut donc trouver un moyen de contourner la restriction nous obligeant à uploader un des formats autorisés.

Figure 9: mauvaise extension de fichier pour l'upload

Une solution qui permet de bypasser ce système est justement de modifier le MIME Type (ou type de média) qui est accepté.

- `image/png` : Portable Network Graphics ; enregistré[18] (attention, à l'instar du jpeg sur le navigateur Internet Explorer le type MIME peut être « image/x-png »).

Figure 10: source : source : wikipedia.org pour le type de média qui correspond aux png

En admettant que le but est d'uploader un fichier au format .php malgré les restrictions, il faut utiliser un outil qui permet d'intercepter des requêtes, d'en modifier le contenu puis de les forwarder comme si de rien n'était.

Sur le forum, plusieurs personnes parlaient d'un outil qui s'appel Tamper Data. Il se trouve que j'ai pas du tout réussi à l'installer donc j'ai utilisé un autre outil beaucoup plus simple : burp. Il suffit simplement d'ouvrir le logiciel et de regarder l'ardesse ip du proxy et son port :

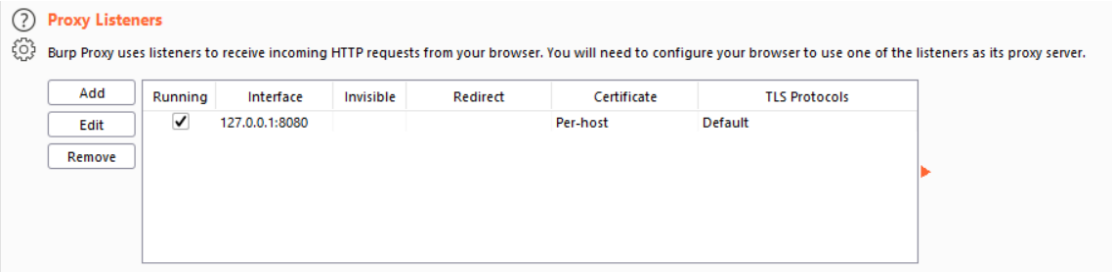

Figure 11: config du proxy burp

Il suffit donc d'aller dans les paramètres du PC, changer le proxy par défaut et le port le temps de l'exercice puis de le remettre une fois terminé.

Dans l'onglet proxy, justement, il nous est possible d'activer ou non l'interception des requêtes. Dans notre cas, la requête à intercepter est uniquement l'upload que l'on va pouvoir forwarder une fois les modifications effectuées.

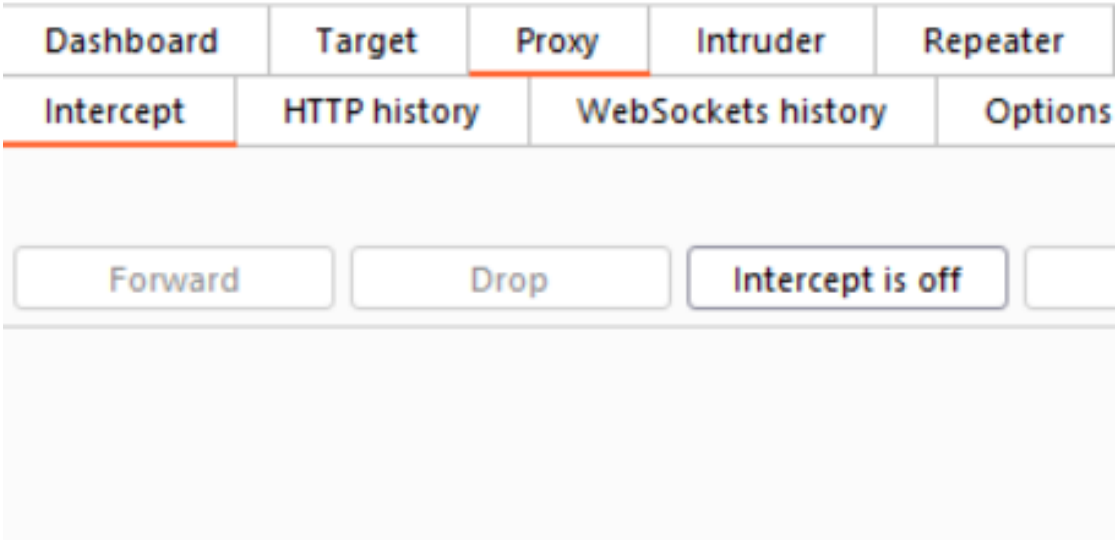

Figure 12: interception des requêtes via burp

Si on retourne donc sur la page d'upload, il nous faut effectivement un fichier, cette fois-ci au format php et on peut normalement garder le même code que dans le premier exercice puisque ca reste un bloc php classique :

```php
<?php
   $mdp = shell_exec('cat /.passwd');
   echo "<pre> $mdp </pre>";
?>
```

Figure 13: contenu du fichier php

Lorsque l'on upload le fichier ce n'est pas censé fonctionner pusique l'extension n'est pas tolérée. Cependant voyons ce qui se passe lorsque l'on intercepte la requête :

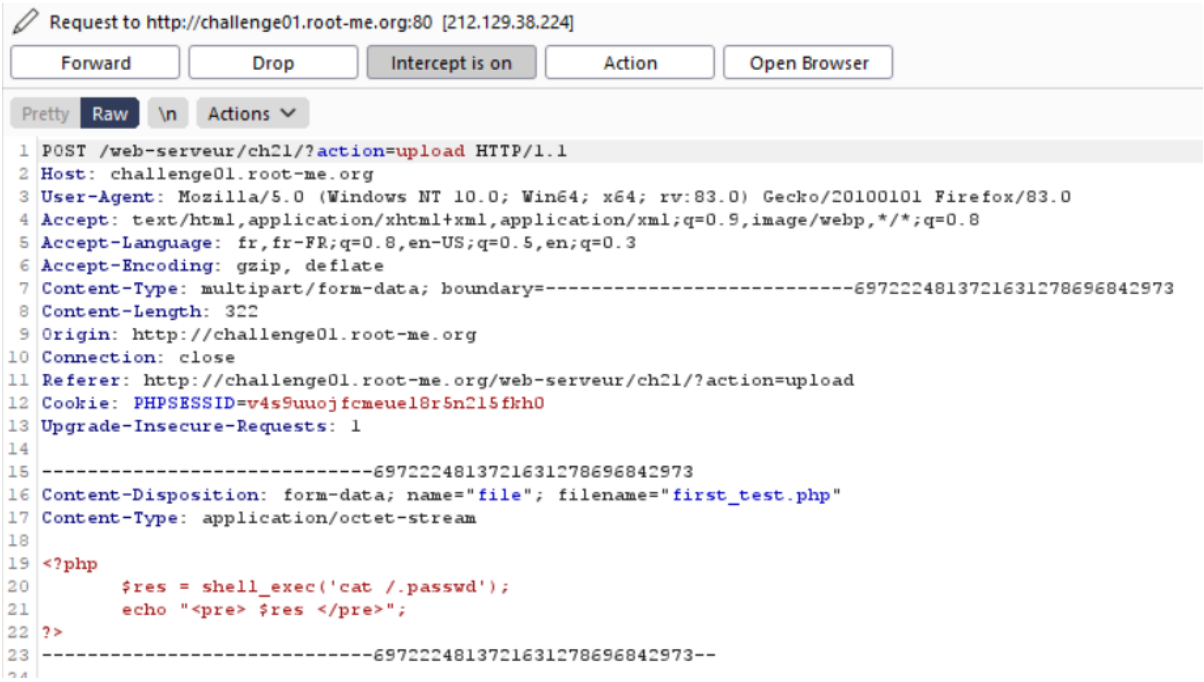

Figure 14: contenu de la requête interceptée

Le bouton en haut à gauche nous permet de forwarder la requête mais dans notre cas, nous voulons la modifier :

Figure 15: MIME Type à modifier pour faire croire que le fichier est bien un .png

On peut alors forwarder la requête et on observe que le fichier est accepté :

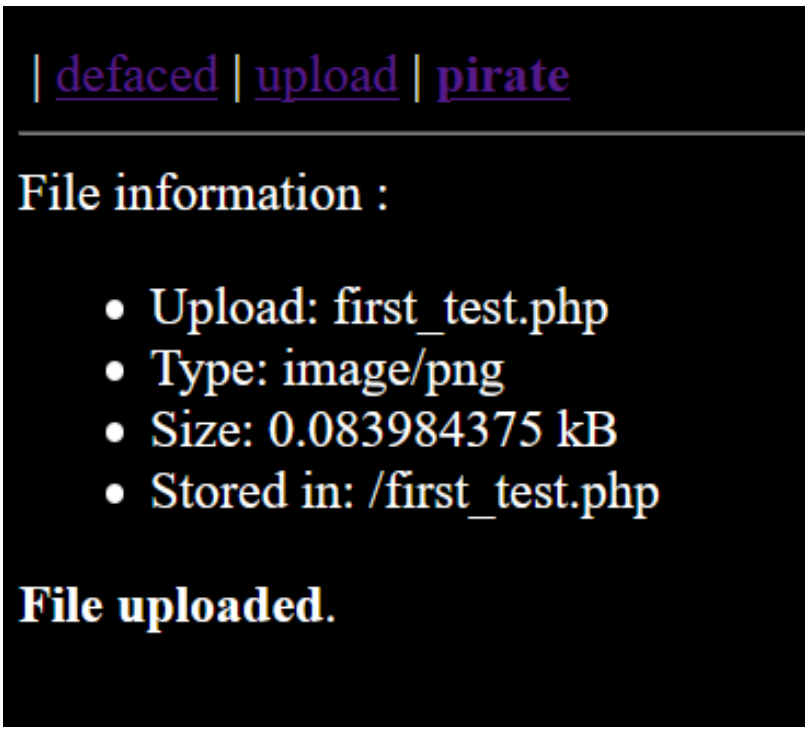

Figure 16: validation de l'upload du fichier malgré son mauvais format

Il nous suffit donc d'ouvrir le fichier pour récupérer le mot de passe :

a7n4nizpgQgnPERy89uanf6T4

Figure 17: affichage du résultat

On peut donc terminer ce challenge en entrant le mot de passe :

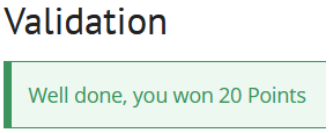

Figure 18: affichage du résultat