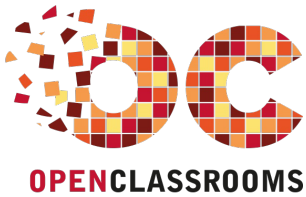
register_globals et écrasement de données

Par luc@s



www.openclassrooms.com

Dernière mise à jour le 7/03/2013

Sommaire

register_globals et écrasement de données

Par luc@s
Mise à jour : 07/03/2013

Certains hébergeurs activent la directive register_globals du PHP.INI.
Activée, cette directive permet d'enregistrer les variables super-globales ($_POST, $_GET, $_COOKIE, $_ENV, $_SERVER) dans des variables normales. Dès lors, si cette directive est à on et la variable $_POST['text'] existe, alors la variable $text de même valeur sera automatiquement créée.
Apparemment, cette directive ne pose aucun problème et au contraire, facilite la programmation. Mais en réalité, celle-ci peut être à l'origine de gros soucis de sécurité.

Si vous ne comprenez pas tout à fait cette introduction, lisez ceci :
PHP.INI

Sommaire du tutoriel :

Les risques

Imaginez maintenant que vous devez gérer une interface administrateur, et que vous avez ce code :

Code : PHP

```php
<?php

$array_admin=array("admin"=>"passe",
        "test"=>"test");

foreach($array_admin as $login=>$pwd) {
 if($login==$_POST['login'] AND $pwd==$_POST['pwd']) {
  $connected=true;
  break;
 }
}

//La variable $connected contient true si l'utilisateur est
administrateur
if($connected==true) {
 echo "Information confidentielle : [...]";
} else {
 ?>
 <form action="index.php" method="post">
     <p><input type="text" name="login" /> Login</p>
        <p><input type="password" name="pwd" /> Password</p>
        <p><input type="submit" value="Connexion !" /></p>
    </form>
 <?php
}
?>
```

La variable $connected, qui vaut true si l'utilisateur a rentré le bon login avec le bon mot de passe, peut être simplement écrasée si register_globals est activé ! Comment ? C'est simple. Il suffit d'appeler la page de connexion ainsi :

Code : PHP

```
index.php?connected=true
```

Dès lors, la variable $_GET['connected'] est copiée dans $connected. Celle-ci aura donc la valeur true avant même que le test de connexion ait été fait.
Le script affiche alors :

Code : PHP

```
Information confidentielle : [...]
```

Désactiver register_globals

Pour désactiver register_globals, c'est simple.
Il suffit, dans un premier temps, de vérifier (avec ini_get()) que register_globals est activé.
Si c'est le cas, alors il faut éditer le PHP.INI et changer la ligne

Code : PHP

```
register_globals On
```

en

Code : PHP

```
register_globals Off
```

Si l'accès au php.ini vous est restreint, alors vous pouvez utiliser un .htaccess que vous placerez à la racine de votre site avec la ligne suivante :

Code : Apache

```
php_flag register_globals off
```

Que faire quand on ne peut pas désactiver register_globals ?

Dès que vous ne pouvez pas désactiver cette directive, 2 possibilités vous sont offertes.

Supprimer les variables à l'aide de PHP

Il suffit alors de parcourir les variables super-globales à l'aide de `foreach`. Puis on utilise `unset` pour supprimer les variables créées à cause de `register_globals`.

Fichier "unregister.php"

Code : PHP

```
<?php
function register_global_off() {
 if(!is_array($_SESSION)) {
  $_SESSION=array();
 }
```

```
 $_TAB=array_merge($_REQUEST,$_SERVER,$_ENV,$_FILES);
 foreach($_TAB as $key=>$value) {
  unset($GLOBALS[$key]);
 }
}
?>
```

Puis on rajoute cette entête à tous les autres fichiers :

Code : PHP

```
<?php
include("unregister.php");
register_global_off();
?>
```

Avantages

Cette méthode permet de travailler dans un environnement de développement plus sûr quand `ini_set` est désactivé.

Inconvénients

Cette méthode est lourde pour deux raisons.
1- Il faut rajouter une entête supplémentaire à chaque fichier : suivant votre application, cela peut ne pas être un gros problème, mais dans le plupart des cas, c'en est un.
2- Cette fonction utilise pas mal de ressources, bien qu'elle en libère un peu en contrepartie (`unset`).

Programmer correctement en initialisant ses variables

Reprenons l'exemple du point n°1 "Les risques". Pour éviter que la variable `$connected` puisse être modifiée par l'utilisateur, il suffit d'initialiser la variable avec la valeur `FALSE`.
Ainsi, même si `$_GET['connected']` existe, la valeur de `$connected` sera `FALSE` avant l'authentification :

Code : PHP

```
<?php
//La variable $connected contient false : on ne sait pas si
l'utilisateur est administrateur !
$connected=false;

$array_admin=array("admin"=>"passe",
        "test"=>"test");

foreach($array_admin as $login=>$pwd) {
 if($login==$_POST['login'] &amp;amp; $pwd==$_POST['pwd']) {
  $connected=true;
  break;
 }
}

//La variable $connected contient true si l'utilisateur est
administrateur
if($connected==true) {
 echo "Information confidentielle : [...]";
} else {
 ?>
 <form action="index.php" method="post">
     <p><input type="text" name="login" /> Login</p>
        <p><input type="password" name="pwd" /> Password</p>
        <p><input type="submit" value="Connexion !" /></p>
    </form>
```

```
 <?php
}
?>
```

Avantages

Cette méthode est très légère, et incite à programmer correctement.

Inconvénients

Cette méthode n'est pas la plus sûre.

Résumons :
register_globals doit être désactivé.
Si ce n'est pas possible, on a deux choix :

- supprimer dynamiquement les variables créées ;
- initialiser ses variables pour éviter un écrasement de données.

J'espère que ce tutoriel vous a plu. Et n'hésitez pas à donner votre avis ! .

Partager

Ce tutoriel a été corrigé par les zCorrecteurs.