Inhaltsverzeichnis

Sicheres Programmieren mit PHP

Top 5

Überprüfung von Client Daten

Daten, die von einem Client gesendet werden, müssen auf dem Server überprüft werden! Hierzu zählen vor allem die Superglobalen **$_GET**, **$_POST** und **$_REQUEST**, sowie **$_COOKIE**.

Hier gilt auch als Regel, wenn ein Formular Daten z.B. nur über die POST Methode verschicken kann, weil das Attribut **method="post"** gesetzt ist, dann werden auf dem Server die Werte dieses Formular auch nur mit **$_POST** abgefragt. Wird das Formular mit **method="get"** abgeschickt, dann wird entsprechend **$_GET** verwendet.

Von der Verwendung der Superglobalen **$_REQUEST** wird in der Regel abgeraten, da hier alle Werte aus **$_GET**, **$_POST** und sogar **$_COOKIE** enthalten sind.

> Die Faustregel lautet: den Daten, die vom Client an den Server geschickt werden, kann grundsätzlich nicht getraut werden. Es muss deshalb immer eine Validierung der Daten auf dem Server stattfinden. Eine Validierung nur im Client genügt nicht, da diese auch manipuliert werden kann.

include($_REQUEST['security'])

In keinem Fall wird ein **include** oder **require** direkt ein Client Wert übergeben! Hierzu muss ggf. mit einem switch gearbeitet werden, auch wenn das viel Code bedeuten mag!

Das PHP muss zudem so eingestellt sein, dass **include**/**require** aus anderen Verzeichnissen als den erlaubten nicht möglich ist (**open_basedir** Restriction).

Keine Cookies verwenden

Auf dem Client kann **localStorage** oder **sessionStorage** verwendet werden. Auf dem Server kann direkt in die Session gespeichert werden.

Es wird nur ein Cookie verwendet, das ist der Session Cookie. Und das geschieht implizit, d.h. im Projekt muss sich kein Entwickler um die Bereitstellung der Session kümmern.

Cross Site Scripting (XSS)

Bei der Ausgabe von vom Client erhaltenen Inhalten, muss ein Filter resp. Modifier angewendet werden, wenn der Wert in HTML wieder eingesetzt wird. Dafür wird der Smarty Modifier **escape** verwendet:

```
<input type="text" name="foobar" value="{$smarty.get.foobar|escape:html}">
```

Wird ein Wert in JavaScript Code eingesetzt, wird ebenfalls der Smarty Modifier **escape** verwendet:

```
let foobar = '{$smarty.get.foobar|escape:javascript}';
```

SQL Injection

Client Werte müssen immer über **Prepared Statements** übergeben werden. Alle EGOCMS Datenbank Methoden unterstützen dafür den **bind** Parameter (z.B. getPages, getChildren, getDescendants, getAncestors, usw.). Beispiel:

```
// Site oder Page Methoden
$pages = $site->getPages([
 'where' => 'name = :name',
 'bind' => [
  'name' => $_GET['foobar']
 ]
]);

// Allgemeine DB Ojekte (z.B. select, insert, update, delete)
$db = new_db_connection();
$db->select([
 'table' => 'foobar',
 'where' => 'field = :value',
 'bind' => [
  'value' => $_GET['foobar']
 ]
]);
```

Coding Guidelines

Eine kurze Zusammenfassung der wichtigsten Coding Guidelines als PDF.

Literatur

- php-einfach.de PHP Sicherheit
- Sicheres Programmieren mit PHP (TU Chemnitz)
- PHP FAQ Sicheres Programmieren in PHP