Douphp 网站后台存储型 XSS 漏洞分析

简单测试

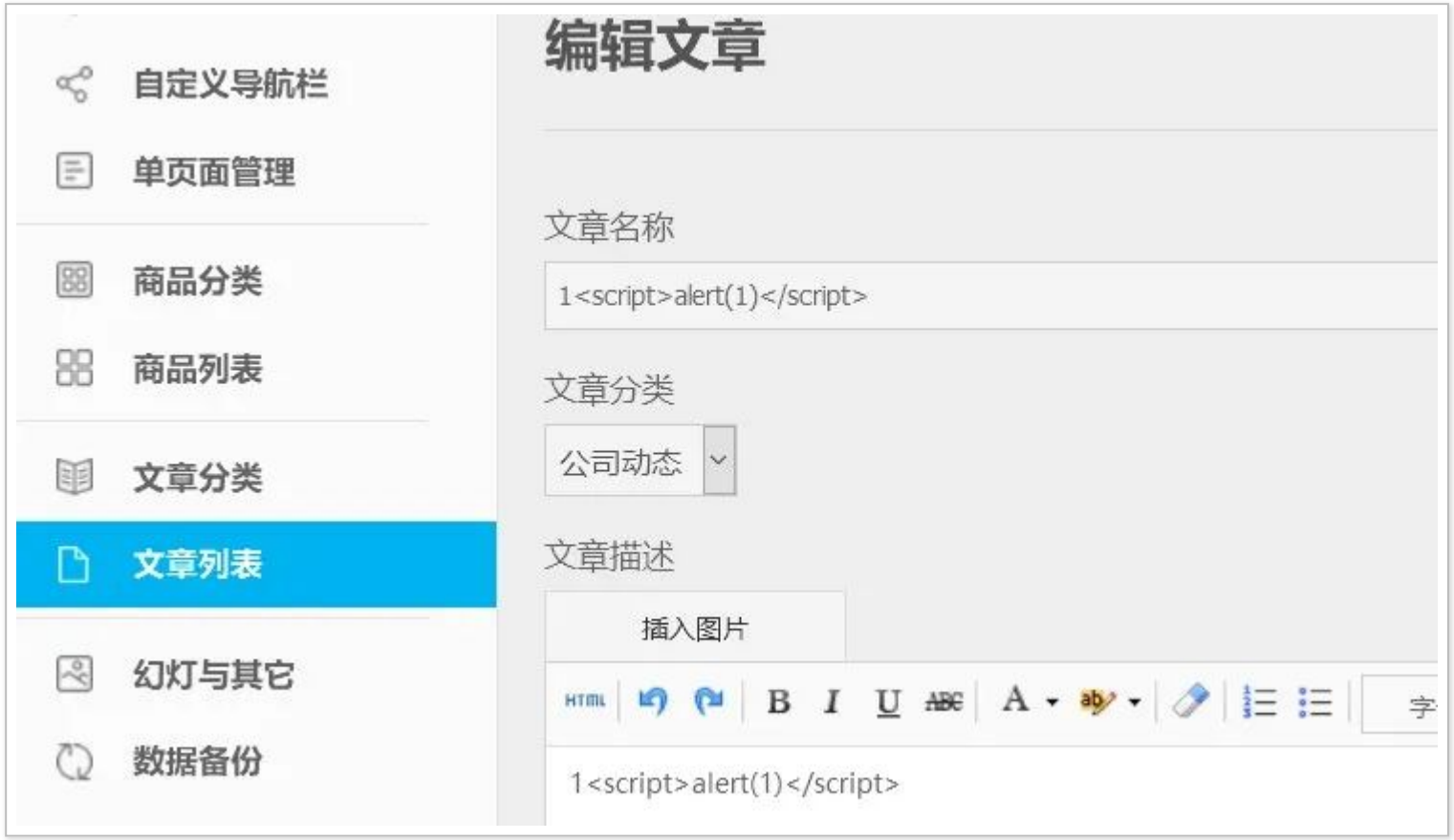
1.1、环境搭建

测试环境：

在官网下载好源码包，解压后访问 upload 目录，根据提示安装好测试环境，即可进入后台页面

1.2、测试 XSS

后台可以新增内容，先试试增加一篇文章，这里框写的是 <script>alert(1)</script>



然后发现页面自动跳转到了文章列表页，并弹出 1

打开前台页面，发现也有弹框 1，是一个存储型的 XSS 漏洞

源码分析

2.1、如何保存到数据库

使用 Seay 代码审计系统对网站源码进行自动审计，通过软件提供的 Mysql 监控功能，发现新建 这篇文章时，执行的 SQL 语句，除了文章描述部分的 <> 被转化成了实体字符，其他如 title 处的 < script>alert(1)</script > 都被完整插入数据库中

```
E = 'dou_article'
_time) VALUES (NULL, '1', '<script>alert(1)</script>', '', '<p>&lt;script&gt;alert...
'添加文章: &lt;script&gt;alert(1)&lt;/script&gt;', '127.0.0.1')
```

查看 admin/article.php 源码进行分析，看起来似乎代码并没有对 $_POST 传递的值进行过滤

```
    $firewall->check_token($_POST['token']);

    $sql = "INSERT INTO " . $dou->table('article') . " (id, cat_id, title,
        defined, content, image, keywords, description, add_time)" . " VALUES (
        NULL, '$_POST[cat_id]', '$_POST[title]', '$_POST[defined]', '$_POST[
        content]', '$image', '$_POST[keywords]', '$_POST[description]', '$
        add_time')";
    $dou->query($sql);

    $dou->create_admin_log($_LANG['article_add'] . ': ' . $_POST['title']);
    $dou->dou_msg($_LANG['article_add_succes'], 'article.php');
}
```

实际上，admin/article.php 在最开始引入了 / inculde/init .php 文件

```
 */
define('IN_DOUCO', true);

require (dirname(__FILE__) . '/include/init.php');

// rec操作项的初始化
$rec = $check->is_rec($_REQUEST['rec']) ? $_REQUEST['rec'] : 'default';
```

而在 admin/inculde/init .php 文件中，实例化了 Check() 与 Firewall 这两个类，调用了 dou_firewall() 方法

```php
// 实例化DouPHP核心类
$dou = new Action($dbhost, $dbuser, $dbpass, $dbname, $prefix, DOU_CHARSET);
$check = new Check();
$firewall = new Firewall();

// 定义系统标识
define('DOU_SHELL', $dou->get_one("SELECT value FROM " . $dou->table('config') . " WHERE
    ));
define('DOU_ID', 'admin_' . substr(md5(DOU_SHELL . 'admin'), 0, 5));

// 豆壳防火墙
$firewall->dou_firewall();
```

这两个类是通过 include/check.class.php 与 include/firewall.class.php 这两个文件来引入，数据的过滤方法就在这里。

```php
require (ROOT_PATH . ADMIN_PATH . '/include/action.class.php');
require (ROOT_PATH . 'include/check.class.php');
require (ROOT_PATH . 'include/firewall.class.php');

// 实例化DouPHP核心类
$dou = new Action($dbhost, $dbuser, $dbpass, $dbname, $prefix, DOU_CHARSET);
$check = new Check();
$firewall = new Firewall();
```

其中，include/check.class.php 文件的 is_number 方法，使用正则对参数进行了过滤，确保传递的 参数是数字

```php
class Check {
    /**
     * +----------------------------------------------------
     * 判断是否为数字
     * +----------------------------------------------------
     */
    function is_number($number) {
        if (preg_match("/^[0-9]+$/", $number)) {
            return true;
        }
    }
```

在 admin/article.php 中调用了该方法，确保 cat _id 参数是数字

```php
        'href' => 'article.php?rec=add'
));

// 获取参数
$cat_id = $check->is_number($_REQUEST['cat_id']) ? $_REQUEST['cat_id'] : ''
    ;
$keyword = isset($_REQUEST['keyword']) ? trim($_REQUEST['keyword']) : '';

// 筛选条件
```

include/firewall.class.php 这个文件中，dou_firewall() 调用了 dou_magic_quot es() 方法

```php
class Firewall {
    /**
     * +----------------------------------------
     * 豆壳防火墙
     * +----------------------------------------
     */
    function dou_firewall() {
        // 交互数据转义操作
        $this->dou_magic_quotes();
    }
```

dou_magic_quotes() 方法的作用，是在 magic_quotes_gpc 没有开启时，调用 addslashes_deep() 方法

```php
    * +----------------------------------------------------------
    * 交互数据转义操作
    * 使用addslashes前必须先判断magic_quotes_gpc是否开启，如果开启的情况下还使用addslashes会导致双层转义，
    即写入的数据会出现/。
    * 服务器默认开启magic_quotes_gpc时会对post、get、cookie过来的数据增加转义字符
    * +----------------------------------------------------------
    */
    function dou_magic_quotes() {
        if (!@ get_magic_quotes_gpc()) {
            $_GET = $_GET ? $this->addslashes_deep($_GET) : '';
            $_POST = $_POST ? $this->addslashes_deep($_POST) : '';
            $_COOKIE = $this->addslashes_deep($_COOKIE);
            $_REQUEST = $this->addslashes_deep($_REQUEST);
        }
    }
```

addslashes_deep()方法的作用，是通过递归的方式，利用 addslashes()函数对' " \ 等特殊字符进行转义，问题应该就是出在这里，函数并未对 <>/() 这些特殊字符进行转义，导致

<script>alert(1)</script > 被完整保存到数据库中

```php
    * +----------------------------------------------------------
    * 递归方式的对变量中的特殊字符进行转义
    * 使用addslashes转义会为引号加上反斜杠，但写入数据库时MYSQL会自动将反斜杠去掉
    * +----------------------------------------------------------
    */
function addslashes_deep($value) {
    if (empty($value)) {
        return $value;
    }

    if (is_array($value)) {
        foreach ((array) $value as $k => $v) {
            unset($value[$k]);
            $k = addslashes($k);
            if (is_array($v)) {
                $value[$k] = $this->addslashes_deep($v);
            } else {
                $value[$k] = addslashes($v);
            }
        }
    } else {
        $value = addslashes($value);
    }

    return $value;
}
```

2.2、如何调用数据

点击文章列表时会弹窗，回头再看 admin/article.php 中的文章列表模块源码，将 $row['title'] 赋值给 "title"，未过滤

```php
        $article_list[] = array (
                "id" => $row['id'],
                "cat_id" => $row['cat_id'],
                "cat_name" => $cat_name,
                "title" => $row['title'],
                "image" => $dou->dou_file($row['image']),
                "add_time" => $add_time
        );
    }

    // 赋值给模板
    $smarty->assign('sort', $dou->get_sort('article', 'title'));
    $smarty->assign('cat_id', $cat_id);
    $smarty->assign('keyword', $keyword);
    $smarty->assign('article_category', $dou->get_category_nolevel('article_category'));
    $smarty->assign('article_list', $article_list);

    $smarty->display('article.htm');
}
```

通过实例化的对象 $smarty 来调用调用 assign 方法，未过滤

```
    * +----------------------------------------------------------
    * 注册变量
    * +----------------------------------------------------------
    */
    function assign($tpl_var, $value = null) {
        if (is_array($tpl_var)){
            foreach ($tpl_var as $key => $val) {
                if ($key != '') $this->_tpl_vars[$key] = $val;
            }
        } else {
            if ($tpl_var != '') $this->_tpl_vars[$tpl_var] = $value;
        }
    }

    /**
```

最后在 admin/templat es/article.htm 中直接取得 $article.title 的值，未过滤

```
<tr>
 <td align="center"><input type="checkbox" name="checkbox[]" value="
 {$article.id}" /></td>
 <td class="m-none" align="center">{$article.id}</td>
 <td><a href="article.php?rec=edit&id={$article.id}">{$article.title}</a>
 <!-- {if $article.image} --> <a href="{$article.image}" target="_blank"><
 img src="images/icon_picture.png" width="16" height="16" align="absMiddle
 "></a><!-- {/if} --></td>
```

最终导致了存储型 XSS 的产生

总结

总结

只是整理了下 XSS 的思路，还请各位师傅指正。