SQL injection in LearnPress <= 3.2.7.2

Security advisory
2020-10-05

Wilfried Bécard
Vulnerability description

Presentation of LearnPress

“LearnPress is a comprehensive WordPress LMS Plugin for WordPress, just like WordPress Moodle or Moodle for WordPress”. LearnPress can be used to create and sell courses on a WordPress instance.

The issue

Synacktiv discovered that LearnPress does not sanitize user input on specific parameters that can be used to alter legitimate SQL queries and inject arbitrary SQL content. An authenticated user on WordPress with at least the contributor privileges is required to exploit this injection.

Affected versions

All versions <= 3.2.7.2 are known to be affected.

Timeline

<table>
<thead>
<tr>
<th>Date</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020-07-01</td>
<td>Advisory sent to LearnPress developers.</td>
</tr>
<tr>
<td>2020-08-23</td>
<td>Version 3.2.7.3 published.</td>
</tr>
<tr>
<td>2020-10-05</td>
<td>Advisory published.</td>
</tr>
</tbody>
</table>

Mitigation

Best practices recommend using parameterized queries and variable binding. These features could be implemented using SQL prepared statements or stored procedures.

For example, in PHP, the PDO API is recommended to implement prepared statements.

1. https://wordpress.org/plugins/learnpress/
Technical description and proof-of-concept

When logged in as a contributor (reminder: administrator > editor > author > contributor > subscriber) on WordPress, the id parameter is not sanitized and can be used to inject arbitrary content into an SQL query.

The code responsible for this vulnerability is located in learnpress/inc/admin/lp-admin-functions.php at line 1690, the $old_post_id parameter is used in the $wpdb→get_results function without sanitization:

```php
function learn_press_duplicate_post_meta( $old_post_id, $new_post_id, $excerpt = array() ) {
    global $wpdb;
    $post_meta_infos = $wpdb->get_results( "SELECT meta_key, meta_value FROM $wpdb->postmeta WHERE post_id=$old_post_id" );
    [...]
}
```

The $old_post_id parameter is received from user input by using the following path:

1. The GET id parameter is received in learnpress/inc/admin/class-lp-admin-ajax.php at line 390:

```php
public static function duplicator() {
    $post_id = LP_Request::get_string( 'id' );
    [...]  
$new_item_id = $curd->duplicate( $post_id, array([...]))
2. Then, learn_press_duplicate_post() is called in learnpress/inc/curds/class-lp-course-curd.php at line 137:

```php
public function duplicate( &$course_id, $args = array() ) {
    if ( ! $course_id ) {
        return new WP_Error( __( '<p>Op! ID not found</p>', 'learnpress' ) );
    }
    if ( learn_press_get_post_type( $course_id ) != LP_COURSE_CPT ) {
        return new WP_Error( __( '<p>Op! The course does not exist</p>', 'learnpress' ) );
    }
    // ensure that user can create course
    if ( ! current_user_can( 'edit_posts' ) ) {
        return new WP_Error( __( '<p>Sorry! You don\'t have permission to duplicate this course</p>', 'learnpress' ) );
    }
    // duplicate course
    $new_course_id = learn_press_duplicate_post( $course_id, $args );
}
```

3. Finally, the vulnerable learn_press_duplicate_post_meta function is called in learnpress/inc/admin/lp-admin-functions.php at line 1671, resulting in an SQL injection.

```php
function learn_press_duplicate_post( $post_id = null, $args = array(), $meta = true ) {
    [...]  
    learn_press_duplicate_post_meta( $post_id, $new_post_id, $exclude_meta );
```
The capability to edit posts is required to exploit this vulnerability, a minimum of contributor privileges are required. The following **curl** request will trigger an SQL injection and sleep for approximately 3 seconds:

```
curl -ski -H 'Cookie: wordpress_92a7607fa4a20e8d9ae07f8580311848=contributor%7C15907[...]'
'http://172.20.0.2/wp-admin/edit.php?lp-ajax=duplicator&id=8+and+%28select+sleep%280.2%29%29'
```

It is important to mention that the id parameter should be a valid post number (additional checks are performed to see if the post exists or not). A user with contributor privileges cannot determine the identifier of a valid post of type course, but this can easily be found by bruteforcing the id parameter.