Vulnerabilities Found in Patreon WordPress plugin

Updated on May 27, 2021 - Marc Montpas

During an internal audit of the Patreon plugin for WordPress, the Jetpack Scan team found several weak points that would allow someone to take over a website.

These vulnerabilities were disclosed to the plugin authors, who promptly released version 1.7.2, which fixes all of these issues. If you're running an older version of the plugin, please update today!

Read on for all of the technical details. If this goes over your head, don't worry. We offer Jetpack Scan to handle malware scanning and automated upgrades or removal for you.

Our team identified various attack vectors, including Local File Disclosure, Cross-Site Request Forgery (CSRF), and Reflected Cross-Site Scripting (XSS) vulnerabilities.

Local File Disclosure vulnerabilities are bugs that bad actors can use to gain access to critical information, such as a website's secret keys and database credentials. Reflected Cross-Site Scripting and Cross-Site Request Forgery vulnerabilities are issues that enable attackers to perform specific actions on behalf of unsuspecting users by tricking them to click carefully crafted malicious links.

If exploited, some of these could enable malicious individuals to take over vulnerable websites.

Local File Disclosure Vulnerability
Affected Versions: < 1.7.0
CVE ID: CVE-2021-24227
CVSSv3: 7.5
CWSS: 83.6

```php
public static function servePatronOnlyImage( $image=false ) {

    if ( ( !isset( $image ) OR !$image ) AND isset( $_REQUEST['patron_only_image'] ) ) {
        $image = $_REQUEST['patron_only_image'];
    }

    if ( !$image OR $image == '') {
        // This is not a rewritten image request. Exit.
        return;
    }

    if ( !( isset( $_REQUEST['patreon_action'] ) AND $_REQUEST['patreon_action'] == 'serve_patron_only_image' ) ) {
        return;
    }

    $upload_locations = wp_upload_dir();

    // We want the base upload location so we can account for any changes to date based subfolders in case there are

    $upload_dir = substr( wp_make_link_relative( $upload_locations['baseurl'] ) , 1 );

    $image = get_site_url() . '/' . $upload_dir . '/' . $image;

    if ( current_user_can( 'manage_options' ) ) {
        Patreon_Protect::readAndServeImage( $image );
    }

    // Below define can be defined in any plugin to bypass core locking function and use a custom one from plugin
    // It is independent of the plugin load order since it checks if it is defined.
    // It can be defined by any plugin until right before the_content filter is run.

    if ( apply_filters( 'ptrn/bypass_image_filtering', defined( 'PATREON_BYPASS_IMAGE_FILTERING' ) ) ) {
        Patreon_Protect::readAndServeImage( $image );
    }

    // Check if the image is protected:

    $attachment_id = attachment_url_to_postid( $image );

    // attachment_url_to_postid returns 0 if it cant find the attachment post id
```

```
    if ( $attachment_id == 0 ) {

        // Couldnt determine attachment post id. Try to get id from thumbnail
        $attachment_id = Patreon_Protect::getAttachmentIDfromThumbnailURL( $image );

        //No go. Have to get out and serve the image normally
        if ( $attachment_id == 0 OR !$attachment_id ) {
            Patreon_Protect::readAndServeImage( $image );
```

Patreon-Connect contained a local file disclosure vulnerability that could be abused by anyone visiting the site. Using this attack vector, an attacker could leak important internal files like wp-config.php, which contains database credentials and cryptographic keys used in the generation of nonces and cookies.

If successfully exploited, this security flaw could lead to a complete site takeover by bad actors.

Reflected XSS on Login Form
Affected Versions: < 1.7.2
CVE ID: CVE-2021-24228
CVSSv3: 8.8
CWSS: 80.6

```
public static function processPatreonMessages() {

    $patreon_error = '';
    if ( isset( $_REQUEST['patreon_error'] ) ) {

        // If any specific error message is sent from Patreon, prepare it
        $patreon_error = ' - Patreon returned: ' . $_REQUEST['patreon_error'];

    }

    if ( isset( $_REQUEST['patreon_message'] ) ) {

        return '<p class="patreon_message">' . apply_filters( 'ptrn/error_message', self::$messages_map[ $_REQUEST['patreon_message'] ] . $patreon_
```

Patreon-Connect hooks on the WordPress login form (wp-login.php) and offers to allow users to authenticate on the site using their Patreon account. Unfortunately, some of the error logging logic behind the scene allowed user-controlled input to be reflected on the login page, unsanitized.

To successfully exploit this vulnerability, an attacker needs to trick his victim into visiting a booby-trapped link containing malicious Javascript code. Since Javascript runs in the victim's browser context, an attacker can adjust the code hidden in that link to do whatever this user's privileges allow him to.

If this attack succeeds against an administrator, the script can completely take over the site.

Reflected XSS on AJAX action 'patreon_save_attachment_patreon_level'
Affected Versions: < 1.7.2
CVE ID: CVE-2021-24229
CVSSv3: 8.8
CWSS: 80.6

```
$args = array (
    'attachment_id' => $attachment_id,
    'patreon_level' => $_REQUEST['patreon_attachment_patreon_level'],
    'message' => $message,
);
echo self::make_image_lock_interface( $args );
```

```
public function make_image_lock_interface( $args = array() ) {

    $interface = '';

    $interface .=  '<div class="patreon_image_lock_modal_content">';
    $interface .=  '<span class="patreon_image_lock_modal_close">&times;</span>';

    $interface .=  ' <form id="patreon_attachment_patreon_level_form" action="/wp-admin/admin-ajax.php" method="post">';
    $interface .=  '<h1 class="patreon_image_locking_interface_heading">Lock Image</h1>';
    $interface .=  '<div class="patreon_image_locking_interface_level">';
    $interface .=  '<span class="patreon_image_locking_interface_input_prefix">$<input id="patreon_attachment_patreon_level" type="text" name="patr
```

The plugin also uses an AJAX hook to update the pledge level required by Patreon subscribers to access a given attachment. This action is accessible for user accounts with the 'manage_options' privilege (i.e., only administrators).

Unfortunately, one of the parameters used in this AJAX endpoint is not sanitized before being printed back to the user, so the risk it represents is the same as the previous XSS vulnerability we described.

CSRF Allowing Attackers To Overwrite/Create User Meta
Affected Versions: < 1.7.0
CVE ID: CVE-2021-24230
CVSSv3: 6.5
CWSS: 42

```
public function toggle_option() {

    if( !( is_admin() && current_user_can( 'manage_options' ) ) ) {
        return;
    }

    $current_user = wp_get_current_user();
```

```
    $option_to_toggle = $_REQUEST['toggle_id'];

    $current_value = get_user_meta( $current_user->ID, $option_to_toggle, true );

    $new_value = 'off';

    if( !$current_value OR $current_value == 'off' ) {
        $new_value = 'on';
    }

    update_user_meta( $current_user->ID, $option_to_toggle, $new_value );

}
```

Some endpoints did not validate the request it received was sent following a legitimate action from a user, which you can do using nonces. One of these unprotected endpoints allowed malevolent individuals to craft a booby-trapped link that would overwrite or create arbitrary user metadata on the victim's account once visited.

If exploited, this bug can be used to overwrite the "wp_capabilities" meta, which contains the affected user account's roles and privileges. Doing this would essentially lock them out of the site, blocking them from accessing paid content.

CSRF Allowing Attackers To Disconnect Sites From Patreon

Affected Versions: < 1.7.0
CVE ID: CVE-2021-24231
CVSSv3: 6.5
CWSS: 26.1

```
if ( isset( $_REQUEST['patreon_wordpress_action'] ) AND $_REQUEST['patreon_wordpress_action'] == 'disconnect_site_from_patreon' AND is_admin() ANI

// Admin side, user is admin level. Perform action:

// To disconnect the site from a particular creator account, we will delete all options related to creator account, but we will leave other plugir

$options_to_delete = array(
    'patreon-custom-page-name',
    'patreon-fetch-creator-id',
    'patreon-creator-tiers',
    'patreon-creator-last-name',
    'patreon-creator-first-name',
    'patreon-creator-full-name',
    'patreon-creator-url',
    'patreon-campaign-id',
    'patreon-creators-refresh-token-expiration',
    'patreon-creator-id',
    'patreon-setup-wizard-last-call-result',
    'patreon-creators-refresh-token',
    'patreon-creators-access-token',
    'patreon-client-secret',
    'patreon-client-id',
    'patreon-setup_is_being_done',
    'patreon-setup-done',
    'patreon-currency-sign',
);

// Ask the API to delete this client:

$creator_access_token = get_option( 'patreon-creators-access-token', false );
$client_id            = get_option( 'patreon-client-id', false );

// Exceptions until v1 v2 transition is complete

$api_version = get_option( 'patreon-installation-api-version' );

if ( $api_version == '1' ) {

    // Delete override - proceed with deleting local options

    foreach ( $options_to_delete as $key => $value ) {
        delete_option( $options_to_delete[$key] );
    }

    update_option( 'patreon-installation-api-version', '2' );
    update_option( 'patreon-can-use-api-v2', true );

    wp_redirect( admin_url( 'admin.php?page=patreon_wordpress_setup_wizard&setup_stage=reconnect_0') );
    exit;
}
```

This one is similar to the last vulnerability in that it's the same kind of attack (CSRF) but is targeting administrators. This particular attack vector works like the previous. An attacker needs to get a logged-in administrator to visit a specially crafted link.

Since this specific endpoint can disconnect a site from Patreon, attackers targeting this attack vector can also do just that, which would prevent new content from being synchronized to the site.

Timeline

- Initial contact attempt (unsuccessful) – Dec 4th
- Second contact attempt – Dec. 11th
- Authors acknowledge the report – Dec. 15th
- Version 1.7.0 is released – Jan 5th
- We report two additional XSS issues – March 9th
- Authors acknowledge the second report – March 9th
- Version 1.7.2 is released – March 11th

Conclusion

We recommend that you check the current version of the Patreon-Connect plugin you are using on your site and, if not 1.7.2, update it as soon as possible!

At Jetpack, we work hard to make sure your websites are protected from these types of vulnerabilities. To stay one step ahead of any new threats, check out Jetpack Scan, which includes security scanning and automated malware removal.

Credits

This security disclosure was made possible thanks to George Stephanis, Fioravante Souza, Miguel Neto, Benedict Singer and Marc Montpas.

*This entry was posted in **Security**, **Vulnerabilities**. Bookmark the **permalink**.*

Marc Montpas

Marc's interests led him to work in the trenches of cybersecurity for the better part of the last decade, notably at companies like Sucuri and GoDaddy. His journey led him to uncover several high-impact security issues while auditing open-source platforms, like WordPress. He's an avid Hacker Capture The Flag player and loves to hypothesize new attack vectors.

Performance (24)
Photos & Videos (9)
Promotions (2)
Releases (166)
Search Engine Optimization (12)
Security (75)
Small Business (16)
Social Media (13)
Support Stories (3)
Tips & Tricks (85)
Uncategorized (5)
Utilities & Maintenance (4)
Vulnerabilities (18)
Website Design (13)
WordAds (1)
WordCamp (3)