Exploit WordPress Plugin vulnerability using static source code analysis techniques

---***---

Submit by: SunCSR (Sun* Cyber Security Research)

I. How WordPress Plugin work?

- **Hooks**
  - events that happen during the execution of the functionality of WordPress, themes or plugins.
  - 1900 hooks : publish_post, save_post, after_signup_user, ...
- **2 types of hook**
  - *Action Hook*: do something at right spot
    - `add_action( 'save_post','wpdemo_my_save_post', 10, 3);`
  - *Filter Hook*: transform data
    - `add_filter( the_title, 'make_title_red', 999);`
- **DB query**
  - `global $wpdb`
- **REST API**
  - `register_rest_route( 'myplugin/v1', '/author/(?P<id>\d+)', array(...) )`

II. Built-in Defense

- **Sanitizers:**
  - **XSS**: addslashes_gpc, esc_attr, esc_attr__, esc_attr_x, esc_html, esc_html__, esc_html_x, esc_js, esc_textarea, tag_escape, wp_htmledit_pre, wp_html_excerpt, _wp_specialchars, wp_specialchars, zeroise,...

- **SQLi**: esc_sql, format_to_post, htmlentities2, sanitize_email, sanitize_file_name, sanitize_html_class, sanitize_key, sanitize_mime_type, sanitize_option, sanitize_sql_orderby, sanitize_text_field, sanitize_title, sanitize_title_for_query, sanitize_trackback_urls, sanitize_user,
- **CSRF**: wp_nonce_field, wp_create_nonce, wp_nonce_url,...

- **$wpdb->prepare**

```
$query = $wpdb->prepare( "UPDATE $wpdb->users SET user_pass = %s, user_activation_key = ''
WHERE ID = %d", $user->user_pass, $user_id );
```

III. Problems

- Open-source
- There is NO official plugin framework
- Incomplete document
  - https://developer.wordpress.org/plugins/
- Sanitizers misuse

	SQLi	XSS
esc_html	✗	✓
esc_attr	✗	✓
sanitize_text_field	✓	✗

IV. From static source code analysis to exploration

1. Using regex to find SQL Injection vulnerability

SQL Injection occurs when user input is not filtered for escape characters and is then passed into an SQL statement. This results in the potential manipulation of the statements performed on the database by the end-user of the application. The SQL statement is constructed by concatenation before it is passed to function to execute, meaning we are vulnerable to maliciously crafted parameters.

Exp 1: (Simple) Regex

```
(?<!prepare)\(('|")SELECT.+FROM.+('|").*\..*
```

Negative lookahead: do not start with prepare

SQL SELECT with string concat

```php
public function get_alb_gals_row( $bwg, $id, $albums_per_page, $sort_by, $order_by, $pagination_type = 0, $from = '' ) {
  $search_value = trim( WDWLibrary::get('bwg_search_' . $bwg) );
  ...
  $search_keys = explode(' ', $search_value);
  ...
  foreach( $search_keys as $search_key) {
    $alt_search .= "`{{table}}`.`name` LIKE '%" . trim($search_key) . "%' AND ";
  ...
  $search_where = ' AND (' . $alt_search . ' OR ' . $description_search . ')';
  ...
  $total = $wpdb->get_var('SELECT count(*) FROM `' . $wpdb->prefix . 'bwg_gallery` WHERE `published`=1' . str_replace('{{table}}', $wpdb->prefix .
  'bwg_gallery', $search_where) );
```

(300,000+ installs) Photo Gallery by 10Web < 1.5.55 - Unauthenticated SQL Injection
https://wpvulndb.com/vulnerabilities/10227

EXP 1: Results

- Blog2Social: Social Media Auto Post & Scheduler < 6.3.1 - Authenticated SQL Injection (60,000+ installs)
  - https://wpvulndb.com/vulnerabilities/10260
- Form Maker by 10Web < 1.13.36 - Authenticated SQL Injection (100,000+ installs)
  - https://wpvulndb.com/vulnerabilities/10237
- AdRotate < 5.8.4 - Authenticated SQL Injection (40,000+ installs)
  - https://wpvulndb.com/vulnerabilities/10249

```php
if (isset($_GET['id'])) $id = esc_attr($_GET['id']);
...
$stats = $wpdb->get_results("SELECT * FROM `{$wpdb->prefix}adrotate_stats` WHERE `ad` = {$id} ORDER BY `id` ASC;");
```

Security flaw

Last week Nguyen Anh Tien from a security firm emailed me that he had discovered a potential flaw in AdRotate.
Turns out that certain URLs passing variables can be exploited in one way or another.

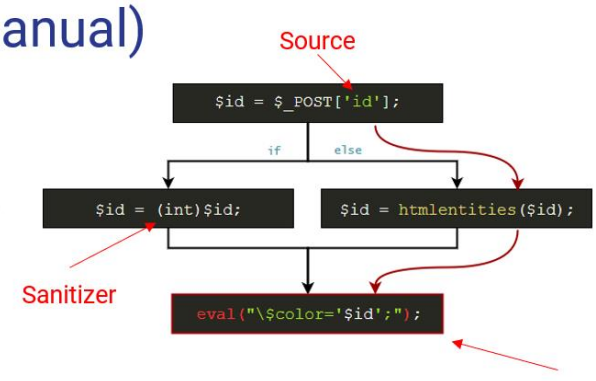
2. Using Taint Analysis

Taint analysis identifies every source of user data — form inputs, headers, you name it — and follows each piece of data through your system to make sure it gets sanitized before you do anything with it.

EXP 2: Taint Analysis (manual)

- https://github.com/viraptor/phply
- https://github.com/ircmaxell/php-cfg
  - PHP code => PHP AST (Abstract Syntax Tree)

1. Find all $wpdb usages
2. SQL not assigned from prepare
3. Using string concat




```
global $wpdb;
foreach($_POST['aDBc_posts_to_clean'] as $post) {
    $post_info = explode("|", $post);
    $wpdb->query("DELETE FROM $wpdb->posts WHERE ID = " . $post_info[1]);
}
```

```
Analyze node -> Foreach(ArrayOffset(Variable('$_POST'), 'aDBc_posts_to_clean'
), None, ForeachVariable(Variable('$post'), False), Block([Assignment(Variabl
e('$post_info'), FunctionCall('explode', [Parameter('|', False), Parameter(Va
riable('$post'), False)]), False), MethodCall(Variable('$wpdb'), 'query', [Pa
rameter(BinaryOp('.', BinaryOp('.', BinaryOp('.', 'DELETE FROM ', ObjectPrope
rty(Variable('$wpdb'), 'posts')), ' WHERE ID = '), ArrayOffset(Variable('$pos
t_info'), 1)), False)])))
  Parent -> 1f28
  Fields ['expr', 'keyvar', 'valvar', 'node']
process_foreach
Analyze node -> ForeachVariable(Variable('$post'), False)
  Parent -> 1d68
  Fields ['name', 'is_ref']
```

EXP 2: Results

```
public function export_full() {
  global $wpdb;
  $slider_ids_string = WDW_S_Library::get('slider_ids', 0);
  $slider_ids_string = rtrim($slider_ids_string, ",");
  ...
  $sliders = $wpdb->get_results('SELECT * FROM ' . $wpdb->prefix . 'wdsslider WHERE id IN (' .
  $slider_ids_string . ')');
```

(50,000+ installs) Slider by 10Web < 1.2.36 - Multiple Authenticated SQL Injection
https://wpvulndb.com/vulnerabilities/10416

```
} else if ( ! empty( $_GET['orderby'] ) and ! empty( $_GET['order'] ) ) {
  $orderby = ( ! empty( $_GET['orderby'] ) ) ? wp_unslash( $_GET['orderby'] ) : $this->primary_col;
  $order   = ( ! empty( $_GET['order'] ) ) ? wp_unslash( $_GET['order'] ) : 'asc';
  $query_to_run = $query;
  $query_to_run .= " order by {$orderby} {$order}";
} else {
  ...
$this->data = $wpdb->get_results( $query_to_run );
```

(100,000+ installs) WP Google Map Plugin <= 4.1.3 - Authenticated SQL Injection
(escalated to wordpress)

3. Using Taint Analysis Tools

EXP 3: Taint Analysis (tool)

- CodeQL: not support PHP ❌
- RIPSTech: paid only ❌
  - Rips-scanner open source: outdated 2016
- Open source
  - PHPWander ❌
    - Need config for each plugin 😕
  - Exakat ❌
  - Psalm (not try yet)
- => Progpilot
  - Wordpress
  - Need custom sources/sinks
- Results
  - 100 hits
  - Exploitable:
  - Still lot of false positive (due to incomplete config, list of sources/sinks) 😅

```php
<?php
require_once './vendor/autoload.php';

$context = new \progpilot\Context;
$analyzer = new \progpilot\Analyzer;

$context->inputs->setFolder("/Users/nguyen.anh.tien/lab/cve/XXX");
$context->inputs->setFrameworks(["wordpress"]);

$context->inputs->setSources([
  ...
]);
$context->inputs->setSinks([
  ...
]);

$context->inputs->setSanitizers([
  ...
]);

$context->inputs->setValidators([
  ...
]);

$analyzer->run($context);
$results = $context->outputs->getResults();
```

EXP 3: Results

```php
function bulk_actions_handler() {
    if( empty($_POST['spamids']) || empty($_POST['_wpnonce']) ) return;
    ...
    if( 'delete' == $action ) {
        $this->removeSpam( $_POST['spamids'] );
    }

}

function removeSpam( $ids ) {
    $ids_string = implode( ', ', $ids );
    ...
    $wpdb->query("DELETE FROM {$wpdb->comments} WHERE
            comment_ID IN ($ids_string)");
}
```

Authenticated SQLi - Spam protection, AntiSpam, FireWall by CleanTalk
(100,000+ installs)

Authenticated SQLi - Contact Form Submissions
(50,000+ installs)

```php
if (isset($_GET['wpcf7_contact_form']) && !empty($_GET['wpcf7_contact_form']))
{
  $form_id = esc_attr($_GET['wpcf7_contact_form']);
  $wpcf7s_columns = $this->get_available_columns($form_id);
...
public function get_available_columns($form_id = 0)
{
  global $wpdb;
  $post_id = $wpdb->get_var("SELECT post_id FROM $wpdb->postmeta WHERE
  meta_key = 'form_id' AND meta_value = $form_id LIMIT 1;");
```

V. Conclusions

- **Site Owners**
    - Always update your plugins (Wordpress 5.5 auto-update)
    - Install WordPress Security Plugins: Sucuri, Wordfence, All in One WP Security & Firewall, ...
- **Hacker/Pentester**
    - Watch closely for new vulns
    - Taint analysis is powerful