Chloe Chamberland

April 20, 2021

Severe Vulnerabilities Patched in Redirection for Contact Form 7 Plugin

On February 11, 2021, our Threat Intelligence team responsibly disclosed several vulnerabilities in Redirection for Contact Form 7, a WordPress plugin used by over 200,000 sites. One of these flaws made it possible for unauthenticated attackers to generate arbitrary nonces for any function. The second flaw made it possible for authenticated attackers to install arbitrary plugins and inject PHP Objects. The third flaw made it possible for authenticated attackers to delete arbitrary posts on a site running the plugin causing a loss of availability.

We initially reached out to the plugin's developer on February 11, 2021. After establishing an appropriate communication channel, we provided the full disclosure the next day on February 12, 2021. The developers of the plugin quickly released a patch the next day on February 13, 2021.

These are considered severe vulnerabilities. Therefore, we highly recommend updating to the latest patched version available, 2.3.5, immediately.

Wordfence Premium users received a firewall rule to protect against any exploits targeting these vulnerabilities on February 11, 2021. Sites still using the free version of Wordfence received the same protection on March 13, 2021.

Description: Unauthenticated Arbitrary Nonce Generation
Affected Plugin: Redirection for Contact Form 7
Plugin Slug: wpcf7-redirect
Affected Versions: <= 2.3.3
CVE ID: CVE-2021-24278
CVSS Score: 5.3 (Medium)
CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
Fully Patched Version: 2.3.4

Redirection for Contact Form 7 is a plugin designed to add redirects to forms created with the popular Contact Form 7 plugin so that users can be redirected immediately after submitting a form. The plugin offers several features like the ability to customize redirects, import settings, and more.

While reviewing the plugin's code, we discovered a nopriv AJAX action that was not actually used anywhere in the plugin, `wp_ajax_nopriv_wpcf7r_get_nonce`, which was hooked to the `wpcf7r_get_nonce` function and made it possible to generate nonces.

```
248 | add_action( 'wp_ajax_nopriv_wpcf7r_get_nonce', array( $this, 'wpcf7r_get_nonce' ) );
249 | add_action( 'wp_ajax_wpcf7r_get_nonce', array( $this, 'wpcf7r_get_nonce' ) );
```

```
260 | public function wpcf7r_get_nonce() {
261 |         $nonce_action = isset( $_POST['param'] ) && wp_unslash( sanitize_text_field( $_POST['param'] ) ) ? wp_unslash
262 |
263 |         $nonce = wp_create_nonce( $nonce_action );
264 |
265 |         wp_send_json_success( array( 'nonce' => $nonce ) );
266 |     }
```

More specifically, a user could set the `param` value to any value while triggering the AJAX action. The `param` value would then be supplied to the `wp_create_nonce()` function as the action value and create a valid nonce for any action, whether the user was logged in or not due to the use of a `nopriv` AJAX action. The function would then return the newly generated nonce back to the user.

At first glance, this doesn't appear to be a significant vulnerability. However, there are a number of plugins in the WordPress repository that insecurely use nonce checks as a substitute for capability checks when enforcing access control. Unfortunately, this meant that this arbitrary nonce generation vulnerability could be used by any unauthenticated user to create a valid nonce. That valid nonce could then grant them access to sensitive functions in any plugin that used nonce checks as the sole means of access control.

Attackers could use this nonce generation vulnerability to perform a plethora of malicious actions, including complete site takeover depending on which other plugins were installed on the site.

Description: Authenticated Arbitrary Plugin Installation
Affected Plugin: Redirection for Contact Form 7
Plugin Slug: wpcf7-redirect
Affected Versions: <= 2.3.3
CVE ID: CVE-2021-24279
CVSS Score: 5.4 (Medium)
CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:N
Fully Patched Version: 2.3.4

One feature that Redirection for Contact Form 7 offers is the ability to import Contact Form 7 forms with their redirection settings along with plugins through their debug importing function. Unfortunately, this functionality was insecurely implemented.

The plugin registered the AJAX action `wp_ajax_import_from_debug`, which was tied to the `import_from_debug` function. This function had no capability checks, nor did it have any nonce protection, making it possible for an authenticated user on a site, including those with just subscriber-level privileges, to trigger the AJAX action and use the functionality of the feature.

```
251 | add_action( 'wp_ajax_import_from_debug', array( $this->wpcf7_utils, 'import_from_debug' ) );
```

```
public function import_from_debug() {
        $data = isset( $_POST['data'] ) && $_POST['data'] ? $_POST['data'] : '';
```

```
            $this->install_plugins( json_decode( $formdata['plugins'] ) );

            $form_id = $this->import_form( $formdata );

            $this->import_actions( $form_id, $formdata['actions'] );

        }
    }
```

All of the data needed to perform the import could be supplied by a user through the 'data' parameter as base64 encoded serialized data. If the 'data' parameter was supplied, then the function would base64 decode and unserialize the data and store the data as `$formdata`. If 'plugins' was contained in the `$formdata`, then the function would trigger the `install_plugins()` function which would check to see if any of the requested plugins were already installed and activated. If any of the requested plugins weren't already installed, then the `install_plugins()` function would trigger the `install_plugin()` function to install and activate them.

```
/**
     * Install and activate a plugin
     *
     * @return void
     */
    public function install_plugin( $plugin_slug ) {

        $api = plugins_api(
            'plugin_information',
            array(
                'slug'   => basename( $plugin_slug, '.php' ),
                'fields' => array(
                    'short_description' => false,
                    'sections'          => false,
                    'requires'          => false,
                    'rating'            => false,
                    'ratings'           => false,
                    'downloaded'        => false,
                    'last_updated'      => false,
                    'added'             => false,
                    'tags'              => false,
                    'compatibility'     => false,
                    'homepage'          => false,
                    'donate_link'       => false,
                ),
            )
        );

        if ( ! is_wp_error( $api ) ) {
            $upgrader = new Plugin_Upgrader( new Plugin_Installer_Skin( compact( 'title', 'url', 'nonce', 'plugin', '

            $upgrader->install( $api->download_link );

            if ( ! is_wp_error( $upgrader->skin->api ) ) {
                return activate_plugin( $plugin_slug );
            } else {
                return $upgrader->skin->api;
            }
        } else {
            return $api;
        }
```

This meant that authenticated users with very minimal permissions, like subscribers, could use this vulnerability to install any plugin from the WordPress repository. An attacker could use this to install plugins containing undisclosed vulnerabilities or other weaknesses and use them to take over a site. For instance, they could install a plugin that insecurely uses nonce protection for access control and combine it with the previous nonce generation vulnerability to further exploit a site.

Description: Authenticated PHP Object Injection
Affected Plugin: Redirection for Contact Form 7
Plugin Slug: wpcf7-redirect
Affected Versions: <= 2.3.3
CVE ID: CVE-2021-24280
CVSS Score: 7.5 (High)
CVSS Vector: CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H
Fully Patched Version: 2.3.4

In addition to the arbitrary plugin installation vulnerability, the `import_from_debug` function was also vulnerable to PHP Object Injection. As previously mentioned, the `import_from_debug` function takes the user supplied 'data' parameter as serialized and base64 encoded data. When supplied, the function base64 decoded and unserialized this data.

```
public function import_from_debug() {
        $data = isset( $_POST['data'] ) && $_POST['data'] ? $_POST['data'] : '';

        if ( $data ) {
            $formdata = unserialize( base64_decode( $data['debug_info'] ) );

            $this->install_plugins( json_decode( $formdata['plugins'] ) );

            $form_id = $this->import_form( $formdata );

            $this->import_actions( $form_id, $formdata['actions'] );

        }
    }
```

Deserializing user-supplied data makes it possible for users to supply PHP Objects in a request to trigger the execution of other PHP classes. Unfortunately, this means that if there is an appropriate magic method, then attackers can do things like remote code execution and arbitrary file creation to completely take over a vulnerable WordPress site.

We did not find any usable magic methods within this plugin which would assist in further exploitation of this vulnerability, however, the arbitrary plugin installation vulnerability could be used to install a plugin that did contain a usable magic method creating a complete POP chain for attackers to exploit.

A magic method is a special PHP function that is added to a class to perform an action when certain conditions are met, like the deserialization of data. Magic methods are typically used to do things like delete files, create files, execute commands, and more which is why they are used with PHP Object injection vulnerabilities to perform harmful actions on a target site's server. For more information on how PHP Object Injection and Magic Methods work, please see this post.

Description: Authenticated Arbitrary Post Deletion
Affected Plugin: Redirection for Contact Form 7
Plugin Slug: wpcf7-redirect
Affected Versions: <= 2.3.3
CVE ID: CVE-2021-24281
CVSS Score: 4.2 (Medium)
CVSS Vector: CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:N/I:L/A:L
Fully Patched Version: 2.3.4

Redirection for Contact Form 7 registers redirects and other features as "actions." You can easily create, duplicate, and remove these actions from any contact form at any time. In order to provide the delete functionality, the plugin registered the `wp_ajax_wpcf7r_delete_action` AJAX action hooked to the `delete_action_post` function.

```
add_action( 'wp_ajax_wpcf7r_delete_action', array( $this->wpcf7_utils, 'delete_action_post' ) );
```

```
/**
     * Delete an action
```

```
        $response['status'] = 'failed';

        if ( $data ) {
            foreach ( $data as $post_to_delete ) {
                if ( $post_to_delete ) {
                    wp_trash_post( $post_to_delete['post_id'] );
                    $response['status'] = 'deleted';
                }
            }
        }

        wp_send_json( $response );
    }
```

The `delete_action_post` function checks for the `data` POST parameter, and if the data contains a 'post_id' it uses the `wp_trash_post()` function to delete the corresponding post id.

Unfortunately, there was no capability check on this action, nor was there any nonce protection, making it possible for any authenticated user, including those with minimal permissions like a subscriber, to delete any post on a vulnerable WordPress site. This could cause a loss of availability.

Description: Unprotected AJAX Actions
Affected Plugin: Redirection for Contact Form 7
Plugin Slug: wpcf7-redirect
Affected Versions: <= 2.3.3
CVE ID: CVE-2021-24282
CVSS Score: 5.0 (Medium)
CVSS Vector: CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:L/I:L/A:L
Fully Patched Version: 2.3.4

In addition to the four vulnerabilities detailed above, there were a few remaining AJAX actions that were unprotected by capability checks and nonces, making it possible for lower level users to execute them. The remaining AJAX actions were significantly less severe, however, attackers could do things like reset the plugin's settings, add actions, delete actions, and more.

Disclosure Timeline

February 11, 2021 – Conclusion of the plugin analysis that led to the discovery of several vulnerabilities in the Redirection for Contact Form 7 plugin. We develop firewall rules to protect Wordfence customers and release them to Wordfence Premium users. We initiate contact with the plugin's developer.
February 12, 2021 – The plugin's developer confirms the inbox for handling discussion the next day. We send over full disclosure.
February 13, 2021 – A newly updated version of Redirection for Contact Form 7 is released containing patches for all of the vulnerabilities.
February 15, 2021 – We confirm the vulnerabilities have been patched.
March 13, 2021 – Free Wordfence users receive firewall rules.

Conclusion

In today's post, we detailed several flaws in Redirection for Contact Form 7 that granted attackers the ability to generate nonces, install arbitrary plugins, delete arbitrary posts, and inject PHP Objects to execute code. These flaws have been fully patched in version 2.3.4. We recommend that users immediately update to the latest version available, which is version 2.3.5 at the time of this publication.

Wordfence Premium users received firewall rules protecting against these vulnerabilities on February 11, 2021, while those still using the free version of Wordfence received the same protection on March 13, 2021.

If you know a friend or colleague who is using this plugin on their site, we highly recommend forwarding this advisory to them to help keep their sites protected as these are severe vulnerabilities that can lead to full site takeover.

Special thank you to Lior Regev at Redirection for Contact Form 7 for an exceptionally fast response in patching the disclosed vulnerabilities.
Did you enjoy this post? Share it!

Comments

6 Comments

Richard *
April 20, 2021
11:37 am

My Contact Form 7 is version 5.4 - what's up?

Chloe Chamberland *
April 21, 2021
6:36 am

Hi Richard,

This affects the "Redirection for Contact Form 7" plugin and not the "Contact Form 7" plugin.

Kadigan *
April 21, 2021
5:22 am

I would like to know just one thing: why was a plugin *of a plugin* exposing plugin installation functionality in the first place?! Shouldn't this be essentially left in the tender care of WP core, as far as separation of concerns goes?

Chloe Chamberland *
April 21, 2021
6:44 am

Hi Kadigan,

The plugin had a debugging feature that would generate debugging information for a form containing the form's data and a list of plugins installed on the site. This debug information could then be used to re-import the contact form and actions along with the plugins installed on the site. This was likely to fix any problems for a form that may have been caused by a plugin being deleted. It was an odd functionality to add, however, it has been removed completed in the updated version of the plugin which is good.

Kadigan *
April 23, 2021
4:54 am

While I can certainly appreciate trying to fix mistakes and such, I feel this shouldn't have been included in the first place -- this is the sort of unexpected, opportune "functionality" that attackers are positively delighted to see. What's worse, this is the sort of functionality you would not expect to see in a plugin like that - so you may not look for it when evaluating the plugin's security profile.

I suppose this is an abject lesson on why you really shouldn't add functionality that's outside of your scope (even if, technically, you're perfectly capable of doing exactly that). Though I suppose I generally have a very similar issue with plugins that expose their own "plugin markets" (like Woo). "Just because you can, doesn't mean you should."

Thanks for a wonderful write-up. Let's just hope someone takes lessons from this.

bacciotti *
April 21, 2021

Now I have to update Cf7 over my websites!

Thanks