WordPress Elementor plugin fixed Safe Mode privilege escalation vulnerability.

BY JEROME BRUANDET MARCH 31, 2020 - 10:51AM [+0700]

Elementor, a popular WordPress plugin installed on 4+ million websites, fixed a high severity vulnerability affecting version 2.9.5 and below. Any authenticated user could enable its Safe Mode feature allowing everyone, logged-in or not, to interact with it and to disable security plugins installed on the website such as firewall, antispam, two-factor authentication or captcha plugins for instance.

Elementor has a Safe Mode feature that is described as “a safe environment that isolates Elementor and WordPress from the themes and plugins that might be causing the error”. It can be enabled by the administrator from the plugin “Tools” page:

The fact that the editor can be loaded without loading any other plugin and the theme is certainly interesting from a developer’s point of view but, because Elementor is accessible to low-privileged users too, it sounds even more interesting from a hacker’s point of view. Let’s see how it works.

When enabling it, it calls the `enable_safe_mode` function from the “elementor/modules/safe-mode/module.php” script:

```
public function enable_safe_mode() {
   WP_Filesystem();

   $this->update_allowed_plugins();

   if ( ! is_dir( WPMU_PLUGIN_DIR ) ) {
      wp_mkdir_p( WPMU_PLUGIN_DIR );
      add_option( 'elementor_safe_mode_created_mu_dir', true );
   }

   if ( ! is_dir( WPMU_PLUGIN_DIR ) ) {
      wp_die( __( 'Cannot enable Safe Mode', 'elementor' ) );
   }

   $results = copy_dir( __DIR__ . '/mu-plugin/', WPMU_PLUGIN_DIR );

   if ( is_wp_error( $results ) ) {
      return false;
   }
}
```

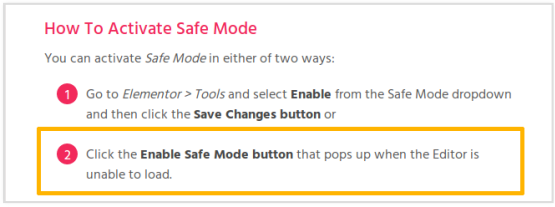
It drops the “elementor/modules/safe-mode/mu-plugin/elementor-safe-mode.php” script into the WordPress “/mu-plugins/” folder:

```
class Safe_Mode {

   const OPTION_ENABLED = 'elementor_safe_mode';

   public function is_enabled() {
      return get_option( self::OPTION_ENABLED );
   }

   public function is_requested() {
      return ! empty( $_REQUEST['elementor-mode'] ) && 'safe' === $
   }

   public function is_editor() {
      return is_admin() && isset( $_GET['action'] ) && 'elementor'
   }

   public function is_editor_preview() {
      return isset( $_GET['elementor-preview'] );
```

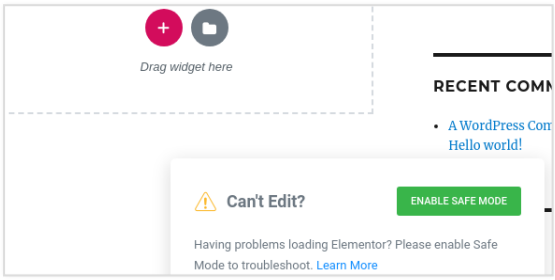
It uses the `pre_option_` hook so that when WordPress queries the database to check which plugins, theme and style sheet must be loaded

(`active_plugins`, `template` and `stylesheet` respectively), it returns that only Elementor should be activated. But this code is accessible to all users, logged-in or not, and can be triggered just by adding a specific query string to an HTTP request that will disable all other plugins while the request is being processed by WordPress.

In Elementor's documentation, we can find some additional info about the Safe Mode activation:



In fact, there isn't one but two ways to enable the Safe Mode: either from the settings page by an admin as described above or from the editor. The latter will only occur if there is a problem while it loads.
With a contributor account, and after messing a bit with some of the HTML code, it wasn't too difficult to force it to appear inside the editor screen:



When clicking the "Enable Safe Mode" button, it will send an AJAX request along with a security nonce to the `handle_ajax_request` function, which will forward it to `ajax_enable_safe_mode` and then will reach our `enable_safe_mode` function that will drop the MU plugin. None of these functions check user capabilities and they only rely on a nonce that is populated in the "elementor/core/common/app.php" script using the `admin_enqueue_scripts` hook.

```
add_action( 'admin_enqueue_scripts', [ $this, 'register_scripts' ] );
```

Because the purpose of this hook is to enqueue scripts for all admin pages, the nonce is accessible to all authenticated users:

```
<script src='http://example.com/wp-content/plugins/elementor/assets/lib/dialog/dialog.min.js?ver=4.7.6'></script>
<script>
var elementorCommonConfig =
{"version":"2.9.4","isRTL":false,"isDebug":false,"activeModules":
["ajax","connect"],"urls":{"assets":"http:\/\/example.com\/wp-
content\/plugins\/elementor\/assets\/"},"ajax":
{"url":"http:\/\/example.com\/wp-admin\/admin-
ajax.php","nonce":"4104ff4191"},"connect":[]};
</script>
<script src='http://example.com/wp-content/plugins/elementor/assets/js/common.min.js?ver=2.9.4'></script>
<script>
```

Therefore, any logged-in user, including a subscriber, can activate the Safe Mode and it can be triggered by anyone, authenticated or not, in the backend or frontend of WordPress. Attackers could use it for several purposes, for instance:

- To send spam: they could disable a captcha and an antispam such as Akismet used to protect a form.
- To bypass a firewall: if there were a vunerability in Elementor or WordPress, attackers could disable the firewall plugin and exploit it.
- To attack the login page: they could disable any plugin used to protect it (two-factor authentication, brute-force protection, activity log, login notification etc) or to hide it.

Note that Elementor's Role Manager won't prevent the exploitation of this vulnerability.
Additionally, if the Safe Mode can be enabled by any user in the backend, it can also be disabled because the `disable_safe_mode` action relies on the same AJAX function and security nonce.

Demonstration

The following video has been edited to obscure some parts of the payload used to perform the attack.

This video demonstrates the exploitation of the Elementor Safe Mode privilege escalation vulnerability.
1. Attackers will use a subscriber account to enable the Safe Mode.
2. They will disable the login page captcha so that they can run a brute-force attack until they find the admin password.
3. They will disable the two-factor authentication security plugin in order to gain access to the administrator account.

Recommendation

Update ASAP if you have version 2.9.5 or below installed.
If you are using our web application firewall for WordPress, NinjaFirewall WP Edition (free) and NinjaFirewall WP+ Edition (premium), you are protected against this vulnerability since March 10, 2020.

Timeline

The vulnerability was reported on March 10, 2020 and a new version 2.9.6 was released on March 12, 2020.

Stay informed about the latest vulnerabilities

- Running WordPress? You can get email notifications about vulnerabilities in the plugins or themes installed on your blog.
- On Twitter: @nintechnet

TAGGED: NINJAFIREWALL, SECURITY, VULNERABILITY, WORDPRESS

PREVIOUS
WordPress Plugins and Themes Vulnerabilities Roundup.

NEXT
Unauthenticated stored XSS vulnerability in WordPress OneTone theme (unpatched).

NinjaFirewall WP+

Web Application Firewall for WordPress. It will give your blog the highest level of protection it deserves.

FREE DOWNLOAD

NinjaFirewall Pro+

Web Application Firewall for PHP applications. It will protect your PHP site, from custom scripts to popular shopping cart and CMS applications.

FREE DOWNLOAD

NinjaScanner

A lightweight, fast and powerful Antimalware scanner for WordPress which includes many features to help you scan your blog for malware and virus.

FREE DOWNLOAD

Code Profiler

Speed up your WordPress website by locating bottlenecks and performance issues in your plugins and themes.

FREE DOWNLOAD

CATEGORIES

Select Category

SEARCH

Search …

RECENT POSTS

1. WordPress FlyingPress plugin fixed broken access control vulnerability.
   November 28, 2022 - 12:13pm [+0700]
2. 8 WordPress plugins fixed high severity vulnerability.

April 12, 2022 - 11:48am [+0700]

3. Unauthenticated function injection vulnerability in WordPress Sparkling theme.
   February 10, 2022 - 5:41pm [+0700]
4. Critical vulnerability in WordPress AdSanity plugin.
   January 25, 2022 - 12:17pm [+0700]
5. Code Profiler: WordPress Website Performance Profiling Made Easy.
   December 19, 2021 - 1:48am [+0700]