

High Severity Vulnerability Patched in TC Custom JavaScript

On June 12, 2020, Wordfence Threat Intelligence discovered an unauthenticated stored Cross-Site Scripting(XSS) vulnerability in TC Custom JavaScript, a WordPress plugin with over 10,000 installations.

Wordfence Premium customers received a new firewall rule to provide protection against attacks targeting this vulnerability the same day. Wordfence users still using the free version received this rule after 30 days, on July 12, 2020.

We attempted to contact the plugin's developer the same day, on June 12, 2020, but we did not receive a response. After 10 days without an initial response, we contacted the WordPress Plugins team on June 22, 2020. An initial patch was released the next day, on June 23, 2020, and a full patch was released on June 29, 2020.

Description: Unauthenticated Stored Cross-Site Scripting(XSS)

Affected Plugin: TC Custom JavaScript Plugin Slug: tc-custom-javascript Affected Versions: < 1.2.2 CVE ID: CVE-2020-14063 CVSS Score: 8.3(high)

CVSS Vector: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:L

Fully Patched Version: 1.2.2

TC Custom JavaScript is a WordPress plugin that allows site owners to add custom JavaScript to every page on a site and emphasizes a minimalist approach. While this kind of functionality can be incredibly useful, it can also be incredibly dangerous without proper access controls.

This plugin ran the ${\tt rccj_core_content::update}$ function immediately upon startup. This was unusual, as most WordPress plugins register functions to be run at predictable points in the WordPress load process.

As a result, the update function ran before WordPress access control functions, such as capability checks and nonce verification, could be used. This, combined with the lack of capability checks and nonce verification on the function, made it possible for unauthorized visitors to use the update function.

The update function used a companion function, has_update_request, to check if the tccj-update POST parameter was supplied and set to update. If this check passed, then the update function would accept the contents of the tccj-update parameter and add it to the database.

While the update function did run htmlentities on the content provided by tccj-content before storing it, this didn't provide any additional safety, as the TCCJ_Core_Frontend::print_script_in_footer function used to display the added script not only used html_entity_decode on the stored content but also added <script> tags around it.

```
class TCC] Core_Frontend {
    polify statistic formula = sanitize_text_filld(_est_option(_tcc]_content', '' ) );
    statistic formula = sanitize_text_filld(_est_option(_tcc]_content', '' ) );
    statistic formula = sanitize_text_filld(_est_option(_tcc]_content', '' );
    statistic formula = sanitize_text_filld(_est_option(_tcc]_content);
    statistic formula = sanitize_text_filld(_est_option(_tcc]_content);
    statistic formula = sanitize_text_filld(_est_option(_tcc]_content);
    if ( statistic filld(_est_option(_tcc]_content);
    if ( statistic filld(_est_option(_tcc]_content);
    }
}

statistic filld(_est_option(_tcc]_content);
    if ( statistic filld(_est_option(_tcc]_content);
    }
}

statistic filld(_est_option(_tcc]_content);

statistic filld(_est_option
```

As such, an attacker could send a Post request to any location on a vulnerable site with the tccj-update parameter set to update and the tccj-content parameter set to malicious JavaScript, and this JavaScript would display in the footer of every page on the site

Malicious JavaScript of this type can be used to redirect visitors to malvertising sites or steal payment information. Even worse, it can detect when an administrator visits the site and send a request on their behalf to infect files with a backdoor or possibly create a new, malicious administrator user account leading to takeover of the entire site.

Timeline

rule available to Wordfence Premium users and attempt to make contact with the plugin's developer.

June 22, 2020 – After failing to make contact with the plugin's developer, we contact the WordPress Plugins team to

June 23, 2020 – The plugin developer releases an initial patch which is still vulnerable to CSRF attacks.

June 29, 2020 – A fully patched version of the plugin is released.

July 12, 2020 – Firewall rule becomes available to free Wordfence users.

Conclusion

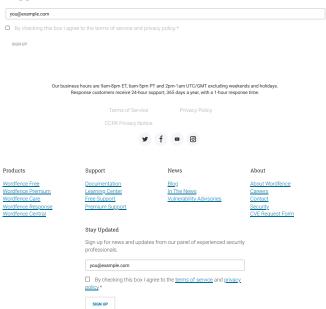
In today's article, we reviewed an unauthenticated stored Cross-Site Scripting(XSS) vulnerability in the TC Custom JavaScript plugin. This flaw has been fully patched in version 1.2.2 and we strongly recommend updating to this version as soon as possible. Sites running Wordfence Premium have been protected against these vulnerabilities since June 12, 2020, while sites still using the free version of Wordfence received the firewall rule on July 12, 2020.

Special thanks to Wordfence QA Lead Matt Rusnak, who initially discovered the vulnerability. Did you enjoy this post? Share it!

Comments

No Comments

Breaking WordPress Security Research in your inbox as it happens.



© 2012-2022 Defiant Inc. All Rights Reserved