Server-Side Template Injection (SSTI) via double template resolution in option rendering

High **bastianallgeier** published **GHSA-jcjw-58rv-c452** 17 hours ago

Package

php **getkirby/cms** (Composer)

Affected versions

<=4.8.0, 5.0.0-5.3.3

Patched versions

4.9.0, 5.4.0

Description

TL;DR

This vulnerability affects all Kirby sites that use option fields (`checkboxes`, `color`, `multiselect`, `select`, `radio`, `tags` or `toggles`) with options from a query or API whose values may not be fully trusted. It also affects direct uses of the `OptionsApi` or `OptionsQuery` classes of Kirby's `Options` package from plugin or site code. The attack requires either an attacker in the group of authenticated Panel users or user interaction of another authenticated user.

This vulnerability is of high severity for affected sites.

Your Kirby sites are *not* affected if you are not using any of the mentioned fields or the `Options` package, if all options are defined statically in the blueprints or if all dynamically gathered options are to be trusted.

Introduction

Server-Side Template Injection vulnerabilities (SSTI) occur when user input is embedded in a template in an unsafe manner and results in remote code execution on the server.

Injected user input is wrongly treated as a template command instead of as a literal string of text. This allows attackers to query arbitrary information from the affected system or call arbitrary methods to perform actions.

In a Kirby site this can be used to access protected site information, alter site content or break site behavior.

Impact

Kirby provides field types (`checkboxes`, `color`, `multiselect`, `select`, `radio`, `tags` and `toggles`) that offer a fixed set of options from a configured list. This configured list can be statically defined in the blueprint or it can come from a Kirby query or (external) API source. Options coming from a query or API are treated as dynamic.

Static options can contain queries in the form `{{ query }}` or `{< query >}` that are then evaluated to a static value. Because the queries are defined in the blueprint, they can be trusted and cannot be controlled by attackers.

However, dynamic options can often not be trusted. This is why the "options from query" and "options from API" modes are intended to resolve the option values and text strings based on queries not defined within the data source but within the blueprint.

Unfortunately, the results of these trusted queries on untrusted source data are run through the query parser a second time in affected Kirby releases.

Because of the double-resolution of dynamic option values and text strings, attackers could place malicious query templates such as `{{ users.first.password }}` or `{{ page.delete }}` in the option sources such as page titles or external API data controlled by the attacker. These queries would then be executed when the field is loaded in the Panel. When the attacker directly accesses the respective Panel view, they could get access to information normally hidden from them. As the malicious query templates are loaded for all users, it could also lead to malicious write access when another user with a higher permission level accesses the manipulated Panel view.

Patches

The problem has been patched in Kirby 4.9.0 and Kirby 5.4.0. Please update to one of these or a later version to fix the vulnerability.

In all of the mentioned releases, we have updated the `Options` logic to no longer double-resolve queries in option values coming from `OptionsQuery` or `OptionsApi` sources. Kirby now only resolves queries that are directly configured in the blueprints.

Credits

Thanks to **@offset** for responsibly reporting the identified issue.

Severity

High 7.6 / 10

CVSS v4 base metrics

Exploitability Metrics	
Attack Vector	Network
Attack Complexity	Low
Attack Requirements	Present
Privileges Required	Low
User interaction	None
Vulnerable System Impact Metrics	
Confidentiality	High
Integrity	High
Availability	None
Subsequent System Impact Metrics	
Confidentiality	None
Integrity	None
Availability	None

Learn more about base metrics

CVSS:4.0/AV:N/AC:L/AT:P/PR:L/UI:N/VC:H/VI:H/VA:N/SC:N/SI:N/SA:N

CVE ID

CVE-2026-34587

Weaknesses

▶ CWE-1336

Credits

offset Reporter