ACF Security Principles

- Introduction
- Security Considerations
- Installation
- Outputting ACF Data Securely

Introduction

Advanced Custom Fields is a plugin for WordPress that allows you to easily store custom data attached to WordPress data types, such as posts, pages, users, taxonomies, etc.

This additional data is stored in the database in either associated meta tables, or the WordPress options table, depending on the location where an ACF field group is displayed.

ACF uses core WordPress functionality for reading and writing from the database tables, so it directly inherits a lot of security and compatibility from WordPress core.

Security Considerations

By default, the data stored inside ACF is not publicly visible. The exception to this is when a field group or individual fields are made available via the REST API.

The ACF Shortcode has historically allowed access to any ACF field value in any location. This meant any user with access to publish posts on your site could access any field they knew the name of. The ACF Shortcode was disabled by default in ACF 6.3.0 for new installations of ACF. ACF 6.3.4 also restricted access to prevent users accessing ACF field values in non-public posts.

Version 6.3.6 of ACF introduced the new security model for accessing fields, with access to fields by editors using the ACF Shortcode or Block Bindings controlled via a new toggle in the “Presentation” tab of each field. For any fields created before 6.3.6, this value is set to true to ensure the previous behavior is inherited, but it will be set to false for any new fields created after 6.3.6.

If a user tries to access a field which is not set to “Allow Access to Value in Editor UI”, the output will be blank, but an error message will be displayed when previewing the post to explain why the value will not be shown.

We recommend users disable editor UI access to all fields unless it’s intended for content editors to be able to access the value of the field via the ACF shortcode or block bindings, especially if that field contains information that is intended to be secure, such as access keys or other secret information.

Installation

ACF is a standard WordPress plugin and can be installed as either a normal plugin or a must-use plugin.

There are no specific security considerations regarding installation, and normal WordPress plugin rules and permissions should be applied. Users installing ACF for the first time should follow the steps outlined here.

Ensuring access to security updates is critical. ACF has instituted an alternative update mechanism, as documented here. This process should be completed for every WordPress site where the free version of ACF is installed. Once completed, plugin updates should be available in the WordPress admin as normal.

Outputting ACF Data Securely

ACF will attempt to use the `unfiltered_html` capability to ensure only admins can store potentially unsafe HTML, but it’s possible for users of contributor level or higher to bypass this protection using other WordPress provided methods of updating post meta. For this reason it’s very important that you escape your field values on output too.

Many users will use ACF to store potentially unsafe HTML, such as iframes or script tags, so by default, ACF does not perform any escaping when using `get_field()`. Therefore, users are required to escape fields on output in their template or themes when using `get_field` or `get_sub_field`.

We provide `acf_esc_html` as a function which can help you do this, which requires you to pass a string which will be run through the WordPress kses system. Alternatively, you could use any other escaping function, such as `wp_kses_post` or `wp_kses` directly, or other PHP escaping options.

Whenever possible, we recommend formatting your fields and theme so that limited HTML is stored in ACF field values and rendered by your theme. For example, instead of storing an entire HTML link tag in your field values and rendering that in your theme, you could store the link URL in your

field value, and in your theme use the more specific `esc_url()` function to escape it:

```
$link = ‘<a href=”’ . esc_url( get_field( ‘your_link_url_field’ ) ); . ‘“>link text</a>`
```

Prior to ACF 6.2.7, `the_field` and `the_sub_field` were also unescaped, but users had no options to escape values using these functions. Since ACF 6.2.7, these functions have handled escaping automatically with `acf_esc_html`.

Since ACF 6.2.5, the output of the ACF Shortcode has also been escaped by default.

□□□ #1
□ □□□ □□□ 2026-08-29 15:44:03 UTC
□ □□□ □□□ 2026-08-29 15:44:03 UTC