ALGORITHME SOLUTIONS INC.

Sanitization Wordpress functions

by:
Donavan Martin
Candidate to the engineering profession (CEP at oiq.qc.ca)

Version non-finale (public)
Translated with DeepL.com
Sherbrooke – June 1, 2021

Contents

1 Introduction **2**

1.1 Data lifecycle for Wordpress renderings 2

2 Sanitize functions (sanitize_*) **2**

3 Escaping functions (esc_*) **3**

4 Propagation of infected data **4**

5 Additive effect of adding extensions and themes **5**

6 Conclusion **7**

References **7**

1 Introduction

The use of Wordpress is very widespread. Today, anyone can create a personalized siteweb in a few clicks for a small fee, but at what price? To answer this question, it is important to understand the life cycle of data in Wordpress and thus avoid certain security holes by using sanitize functions of Wordpress. This publication is more for developers, but can be understood by most of Wordpress users.

1.1 Data lifecycle for Wordpress renderings

At a high level, the system proposed by Wordpress is simple. Indeed, this content manager offers efficient ways to publish information according to several levels of user permissions. The data cycle in wordpress can be summarized in 3 actions:

- Adding (content creation)
- Backup (database / files)
- Rendering (data display)

The high-level model of the data lifecycle for renderings in Wordpress is discussed in Figure 1-1.

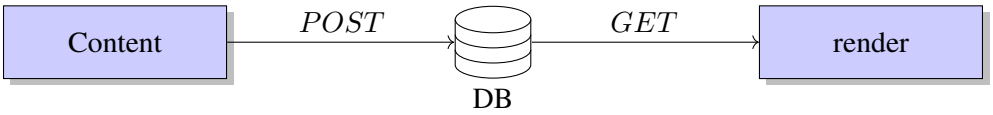

Figure 1-1: High-level data cycle model for rendering in Wordpress.

2 Sanitize functions (sanitize_*)

The sanitizing functions are used to eliminate invalid characters. They allow to secure the data entry in the database.[1]

Here is the current list:

- sanitize_email()
- sanitize_file_name()
- sanitize_html_class()
- sanitize_key()

- sanitize_meta()
- sanitize_mime_type()
- sanitize_option()
- sanitize_sql_orderby()
- sanitize_text_field()
- sanitize_title()
- sanitize_title_for_query()
- sanitize_title_with_dashes()
- sanitize_user()
- esc_url_raw()
- wp_filter_post_kses()
- wp_filter_nohtml_kses()

However, there is nothing to stop you from modifying the data in a database by using SQL queries directly, for example. So although sanitizing functions are useful for inserting data into the database, sanitizing functions are not enough.

3 Escaping functions (esc_*)

Escape functions ensure that data is rendered on the user's web page. Similar to sanitizing functions, escape functions validate the data on the server before sending it to the user. [1]

Voici la liste actuels:

- esc_html()
- esc_url()
- esc_js()
- esc_attr()
- esc_textarea()

4 Propagation of infected data

A common mistake made by Wordpress users is to consider the system's features secure by default. However, this is not the case. For example, it is enough to modify the title of a Wordpress "Post" with an HTML <script> tag to inject code into the client browser in order to validate the absence of appropriate escape functions. Here is a simple example showing that the "title" field of a Wordpress "Post" does not use a sanitizing function. The figure 4-1 shows the title of a post that has been infected in the database. During the tests we used the latest version available (Wordpress 5.7.2).

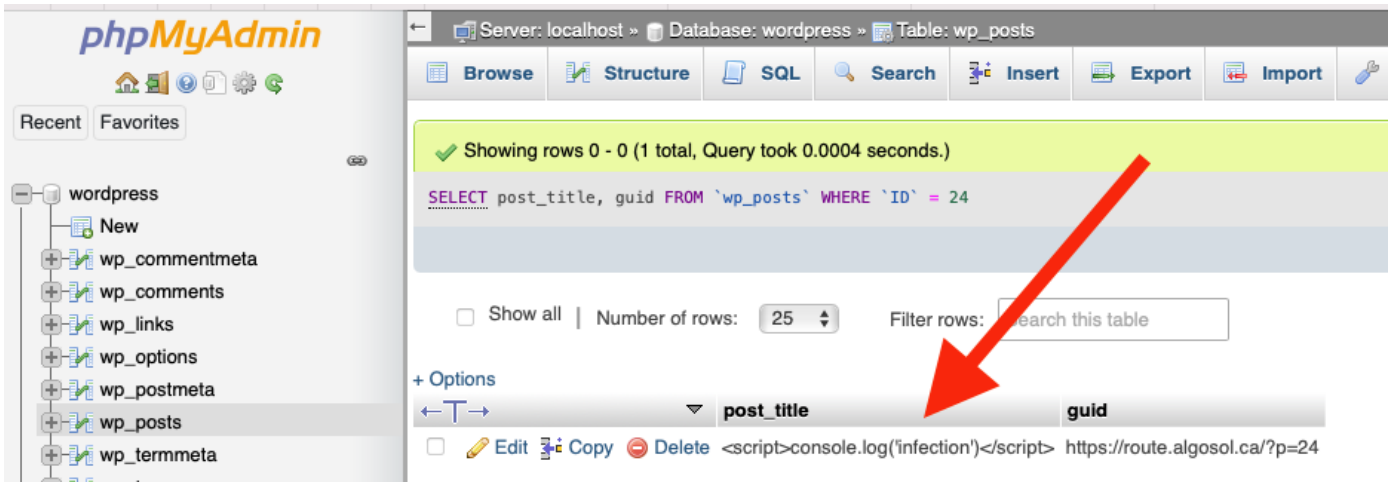

Figure 4-1: Title of a post that has been infected in the database

Afterwards, you just have to open the web page to check the code execution. The figure 4-2 shows that Wordpress does not use an escape function to render the page title.

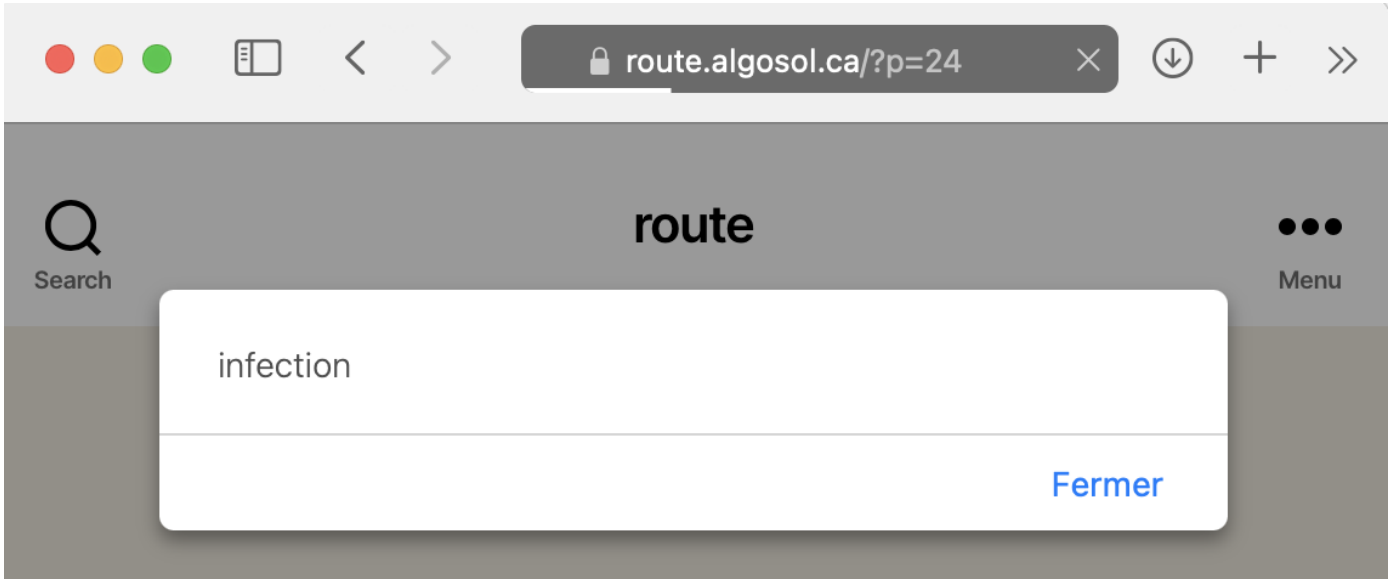

Figure 4-2: Execution of the script injected in the title

5 Additive effect of adding extensions and themes

By principle of efficiency, the addition of Wordpress extensions has now become a must. Indeed, extensions allow to quickly add features and themes allow to display a professional looking visual rendering. However, very few users validate the use of sanitizing functions and escape functions, because the majority of users consider extensions and themes to be safe.

In this regard, the figure 5-1 shows the additive effect of flaws due to the lack of default validation in the data cycle for renderings in Wordpress. This diagram adds the location of sanitizing functions and escape functions to the high-level diagram. The extensions and themes area is framed by a dotted line. In this area we notice that the probability of the number of data validation flaws is a function of the number of extensions used and the theme.

It is therefore important to limit the number of active extensions on the Wordpress platform to reduce the probability of potential vulnerabilities.

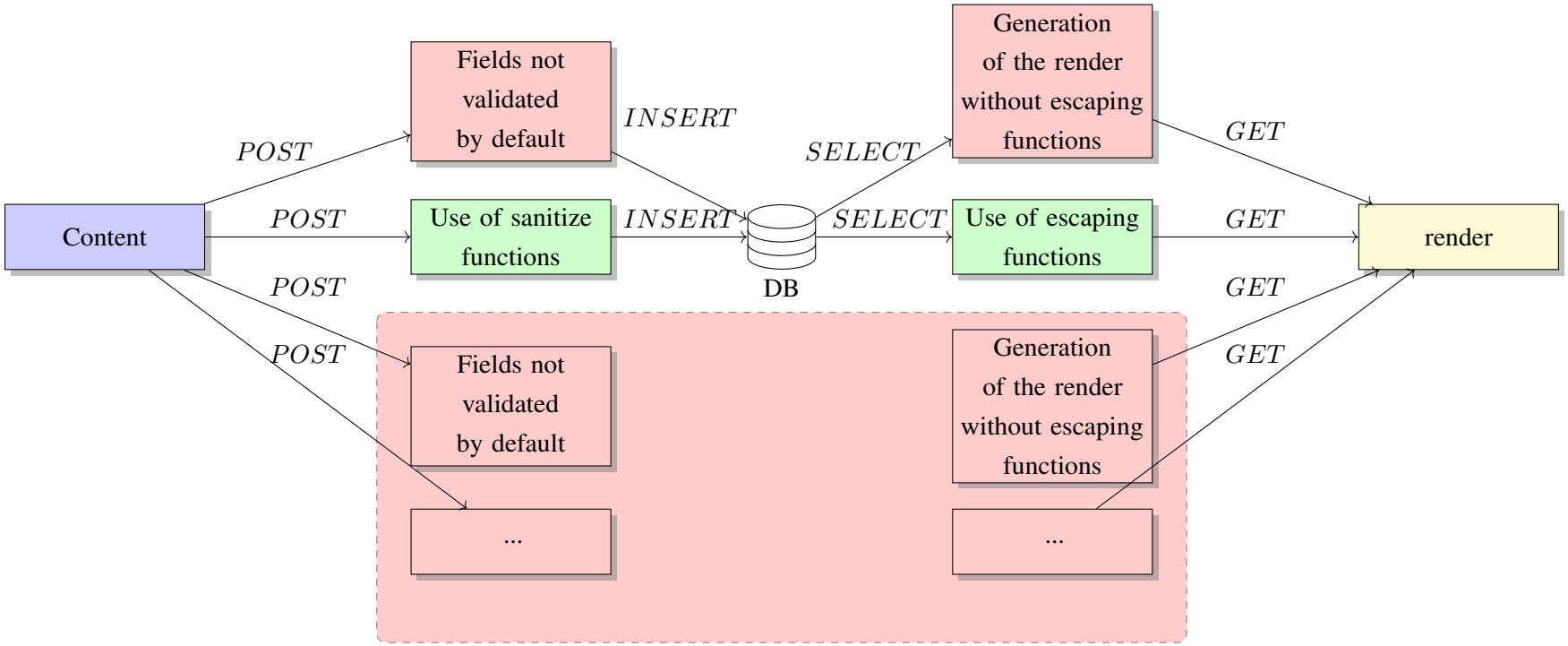

Figure 5-1: Additive effect of flaws due to lack of validation in Wordpress extensions and themes

6 Conclusion

The lack of use of sanitizing and escape functions in Wordpress extensions and themes adds many potential security holes. To check if an extension or a theme validates the data before displaying it, simply check the use of these functions when saving the data and when generating the rendering.

References

[1] Wordpress. Data sanitization/escaping, https://developer.wordpress.org/themes/theme-security/data-sanitization-escaping/.