ALGORITHME SOLUTIONS INC.

Les fonctions désinfectantes de Wordpress

Rédigé par :
M. Donavan Martin
Candidat à la profession d'ingénieur (oiq.qc.ca)

Version non-finale (public)
Sherbrooke – 28 mai 2021

Table des matières

1 Introduction

L'utilisation de Wordpress est très répandue. Aujourd'hui, n'importe qui peut créer en quelques cliques un site web personnalisé pour une modique somme, mais à quel prix ? Pour répondre à cette question, il est important de bien comprendre le cycle de vie des données dans Wordpress et ainsi éviter certains pièges. Cette publication est davantage pour les développeurs.es, mais nous avons suffisamment vulgarisé l'information pour être compris par la majorité des utilisateurs de Wordpress.

1.1 Cycle de vie des données pour les rendus dans Wordpress

À haut niveau, le système proposé par Wordpress est simple. En effet, ce gestionnaire de contenus propose des moyens efficaces de publier de l'information en fonctions de plusieurs niveaux de permissions utilisateur. Le cycle des données dans wordpress se résume en 3 actions :

— Ajout (création de contenue)

— Sauvegarde (base de données / fichiers)

— Rendu (affichage des données)

Le modèle à haut niveau du cycle de vie des données pour les rendus dans Wordpress est abordé par la figure 1-1.

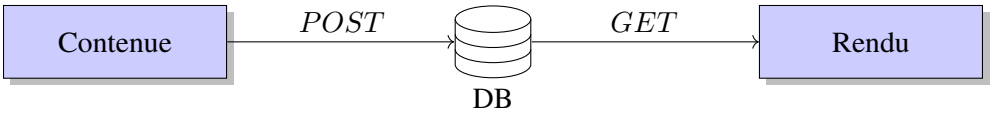

FIGURE 1-1 – Modèle à haut niveau du cycle des données pour les rendus dans Wordpress.

2 Les fonctions désinfectantes (sanitize_*)

Les fonctions désinfectantes servent à éliminer les caractères invalides. Elles permettent de sécuriser l'entrée des données dans la base de données. [1]

Voici la liste actuels :

— 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()

Cependant, rien n'empêche de modifier les données dans une base de données en utilisant directement des requêtes SQL par exemple. Donc bien que les fonctions désinfectantes soient utiles pour l'insertion de données dans la base de données, les fonctions désinfectantes ne suffisent pas.

3 Les fonctions d'échappement (esc_*)

Les fonctions d'échappement permettent d'assurer le rendu des données sur la page web de l'utilisateur. Du même principe que les fonctions désinfectantes, les fonctions d'échappement valident les données sur le serveur avant de les envoyer à l'utilisateur. [1]

Voici la liste actuels :

— esc_html()

— esc_url()

— esc_js()

— esc_attr()

— esc_textarea()

4 Propagation des données infectées

Une erreur courante chez les utilisateurs Wordpress est le fait de considérer les fonctionnalités du système sécuritaires par défaut. Or, ce n'est pas le cas. Par exemple, il suffit de modifier le titre d'un "Post" Wordpress par une balise HTML <script> pour injecter du code dans le navigateur client afin de valider l'absence d'utilisation de fonction d'échappement approprié. Voici un exemple simple montrant que le champ "titre" d'un "Post" Wordpress n'utilise pas de function désinfectante. La

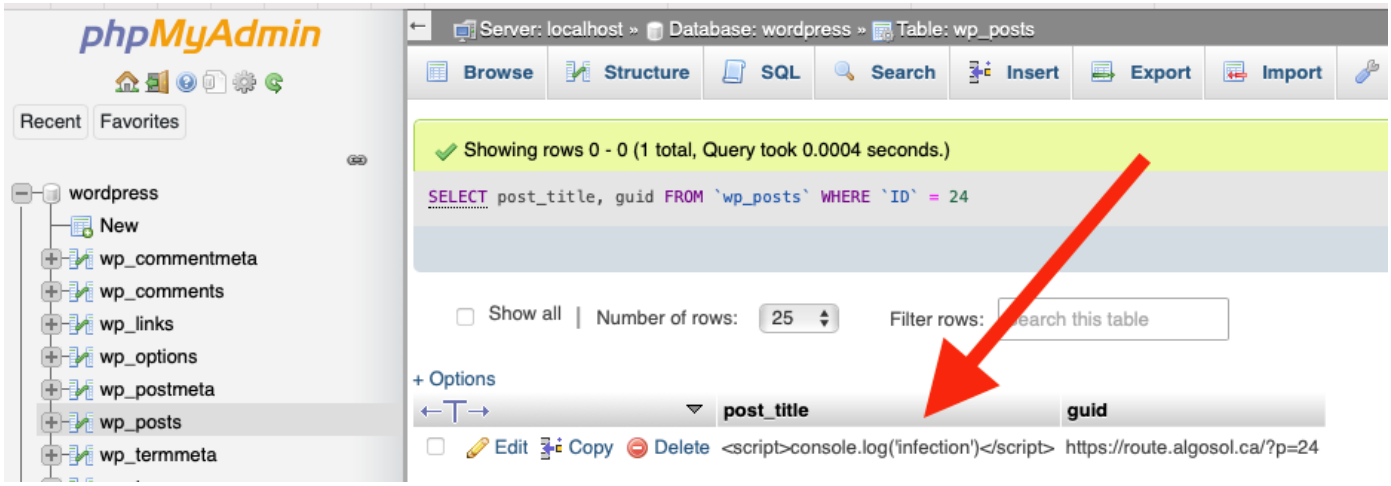

FIGURE 4-1 – Titre d'un post qui à été infecté dans la base de données

figure 4-1 montre le titre d'un post qui a été infecté dans la base de données. Lors des tests nous avons utilisé la dernière version disponible (Wordpress 5.7.2).

Par la suite, il suffit d'ouvrir la page web pour vérifier l'exécution du code. La figure 4-2 montre que Wordpress n'utilise pas de fonction d'échappement pour le rendu du titre de la page.

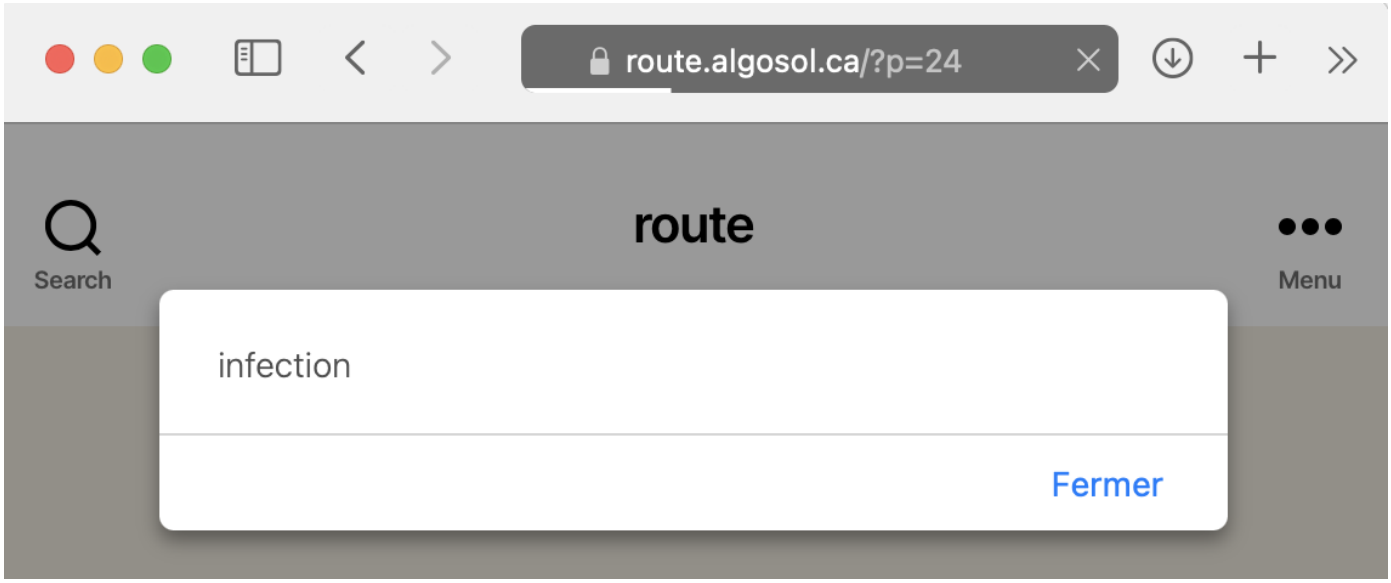

FIGURE 4-2 – Exécution du script injecté dans le titre

5 Effet additif d'ajout d'extensions et de thèmes

Par principe d'efficacité, l'ajout d'extensions Wordpress est aujourd'hui devenu un incontournable. En effet, les extensions permettent d'ajouter rapidement des fonctionnalités et des thèmes permettent d'afficher un rendu visuel à apparence professionnel. Cependant, très peu d'utilisateurs.es valident l'utilisation des fonctions désinfectantes et des fonctions d'échappement , car la majorité des utilisateurs considèrent les extensions et thèmes sécuritaires.

À ce propos, la figure 5-1 montre l'effet additif des failles en raison de l'absence de validation par défaut dans le cycle des données pour les rendus dans Wordpress. Ce schéma ajoute l'emplacement des fonctions désinfectantes et les fonctions d'échappement au schéma à haut niveau. La zone d'extensions et de thèmes est cadrée par une ligne en pointillé. Dans cette zone on remarque que la probabilité du nombre de failles en lien avec la validation des données est fonction du nombre d'extensions utilisées et du thème.

Il est donc important de limiter le nombre d'extensions actives sur la plateforme Wordpress pour réduire la probabilité des failles potentielles.

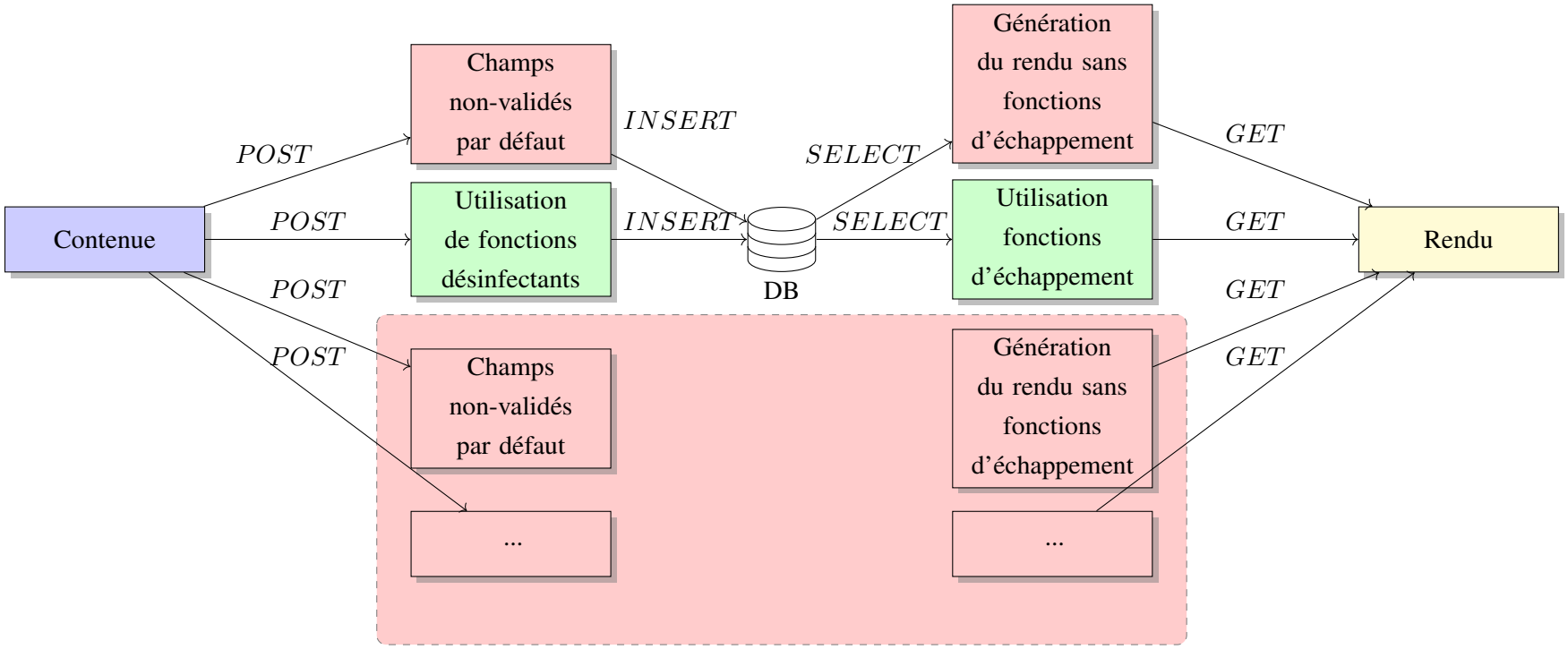

FIGURE 5-1 – Effet additif des failles en raison de l'absence de validation dans les extensions et thèmes Wordpress.

6 Conclusion

L'absence de l'utilisation de fonctions désinfectants et de fonctions d'échappement dans les extensions et les thèmes Wordpress ajoute de nombreuses failles potentielles de sécurité. Pour vérifier si une extension ou un thème valide les données avant de les affichées, il suffit de vérifier l'utilisation de ces fonctions lors de la sauvegarde des données et lors de la génération du rendu.

Références

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