Content Injection Vulnerability in WordPress 4.7.0 and 4.7.1

Madis Janno

Tartu 2017
Introduction to WordPress

WordPress is a content management system used for blogs and websites. It is free and open-source and has numerous plugins. 27.9% of websites use WordPress, and it has a content management system market share of 58.9%, making it by far the most popular content management system for websites and blogs.

Vulnerability

WordPress 4.7.0 included a new REST API, which was enabled by default. One of the REST endpoints allowed access to view, edit and delete posts. A bug allowed visitors to edit any post on the WordPress site in versions 4.7.0 and 4.7.1 and was fixed in version 4.7.2. The fix was initially undisclosed, and was disclosed a week later in order to allow websites to update before hackers became aware of the vulnerability.

Technical details

The exploit was mainly due to two problems working together.

The first problem was the way parameters were populated in ./wp-includes/rest-api/endpoints/class-wp-rest-posts-controller.php. The registered routes were done in such a way, that for example in case of request like “/wp-json/wp/v2/posts/1234” the id parameter would be filled out to “1234”. This was done in a way that would stop the exploit by only allowing numeric values, but the problem was, that regular get and post parameters would override the ones created by the route, meaning that in case of request to “/wp-json/wp/v2/posts/1234?id=12345helloworld” the id would not be “1234”, but “12345helloworld”.

The second problem was the way the id was used in the method update_item_permissions_check and update_item. If no post was found with the given id,
the permission check would return true, allowing the user to proceed to update_item. update_item also attempts to find the post, but first converts the id to an integer. Which means that an invalid id like “456ABC” would first pass the permission check due to a post not being found, and later be converted to “456” which could be a valid post, thus essentially skipping the permission check.

This combination of two problems allowed users to both send requests with an otherwise invalid id, and for the system to allow those id’s to skip the permission check which was meant to disallow editing by unauthorized users. Meaning users could edit any post and thus exploit vulnerabilities in plugins that would otherwise only be restricted to contributor roles, infect the site with SEO spam, inject ads, etc.

Fix

The vulnerability was discovered by Sucuri by one of the researchers Marc-Alexandre Montpas. He informed the WordPress development team and the exploit was fixed in WordPress 4.7.2. Several companies and websites were informed of the vulnerability prior to the completion of the fix. There was no indication that the vulnerability had been exploited in the wild and so it was decided that the inclusion of the fix would remain undisclosed for a short period of time. As of February 1, there had been no attempts to exploit the vulnerability in the wild. But later comments speak of large scale attacks against websites still using 4.7.0 or 4.7.1.

The fix was relatively simple, a new function get_post() in ./wp-includes/rest-api/endpoints/class-wp-rest-posts-controller.php to be used instead of the regular get_post(). The new function, among other things, calls the regular get_post() function with the parameter first converted to an integer, thus giving both update_item_permissions_check and update_item the same behaviour regarding id’s.
References

https://blog.sucuri.net/2017/02/content-injection-vulnerability-wordpress-rest-api.html

https://make.wordpress.org/core/2017/02/01/disclosure-of-additional-security-fix-in-wordpress-4-7-2/

https://wordpress.org/about/

https://core.trac.wordpress.org/changeset/39997/branches/4.7?old=39871&old_path=branches%2F4.7