MEASURING PLUGIN IMPACT ON WORDPRESS PERFORMANCE

Liuiza A., Poška M., Rimkutė I.
Kaunas University of Applied Sciences

Goal of the research: to create a method for measuring plugin impact on WordPress performacne
Research methods: empirical research, systematical and logical analysis and synthesis
Purpose: this article presents a method to measure WordPress performance

Introduction

In this day and age of technology, websites take evergrowing part of human life. Every company or freelancer needs to have a website, so there is a big need for tools that allow people to create and manage websites easily. One of the most popular such tools is WordPress (27.8 percent of top 1 million websites in the world use this software [1]). One of the reasons why WordPress became so popular is its plugin ecosystem of almost 50.000 of free plugins [2], that allows to easily extend its functionality.

But easy extendibility has a drawback, because not all WordPress plugins are of good quality. A badly coded plugin might take a big toll on websites performance, so it is important to know, what impact does particular plugin have on site's performance.

This research project aims to create a method to measure performance impact of WordPress plugins. As an example, the very popular „Jetpack" plugin [3] is being measured. This plugin was selected because it has a big user base and also a reputation for having a lot of impact on website performance.

Methodology

Dataset. If the tests were run on „Hello World" WordPress installation, they will not show much in terms of performance for modules like Related posts. So it was decided to try and build a decent set of data for the test site. A sample dataset was built with. For content, country, capital and mayor city descriptions from Wikipedia were taken. Some pages, some comments,

categories and tags were also created. Every post has a thumbnail, so there are a lot of media items in there, too. The final dataset includes:

- 500 posts with thumbnails;
- 20 pages;
- 50 comments;
- 48 tags;
- 3 categories;
- 500 images.

Metrics. Main backend-metrics were measured:

- Peak Memory Usage
- PHP Execution Time
- SQL Read Query Count
- SQL Write Query Count
- SQL Execution Time
- Local Script/Style Request Count
- Remote Sricpt/Style Request Count
- Remote HTTP GET Request Count
- Remote HTTP POST Request Count

Test system. A completely ordinary Digital Ocean droplet (1GB RAM, 1 CPU, 30GB SSD) was taken and a basic LEMP stack (Ubuntu 14.04.4 LTS, nginx 1.4.6, PHP 5.5.9, MariaDB 5.5.47) was set up. The latest WordPress version at the time (4.6.1) was installed, as well as the latest (4.3.2) version of Jetpack and the default Twenty Sixteen 1.2 theme.

Process. For a baseline, a plain WordPress installation (no plugins) was measured. Then, Jetpack plugin was activated and step by step, every Jetpack module was turned on individually. For every module, 9 pageviews were generated. The lowest and the highest values were discarded to eliminate possible fluke results. And an average was calculated from the remaining 7 runs.

Tools

At first, team was planning to use John Blackbourn's excellent Query Monitor plugin [4] and Pingdom's Website Speed Test tool [5] to measure performance, but that would have meant a lot of manual work and would not allow for fully testing both anonymous and logged-in state of the website.

So with a goal to automate the process as much as possible, a custom set of measuring tools was built in a form of a small and simple must-use plugin. It uses a lot of the same methods for measuring as Query Monitor

does, but it also allows to turn the measuring on/off via GET arguments and stores the results in CSV format for later analysis. One of the goals was to keep the footprint of the plugin as light as possible, it currently is only just over 100 lines long and does most of its work on the shutdown hook, when WordPress execution is already done.

This plugin uses SAVEQUERIES constant to tell $wpdb to log all queries it makes, PHP's memory_get_peak_usage(false) to get memory usage, and WordPress' built-in stop_timer() function to measure PHP execution time. $wp_styles and $wp_scripts global variables are used to determine and count what scripts and styles were loaded in a particular pageview. The only measurement that happens before shutdown hook is counting remote HTTP requests via HTTP API – for that the plugin hooks hooks into pre_http_request for that.

Results

Just installing and activating Jetpack will add 0.47 Mb of peak memory usage. Compared to 3.33 Mb usage of a plain WordPress installation that is a significant increase. Which gets even more significant if the Recommended modules (2.04 Mb) are turned on or every single module included with Jetpack (2.451 Mb) is activated.

There also is an increase in PHP execution time. On average, Jetpack adds ~0.076s to it. It seems very small in absolute terms, but relative to results plain WordPress on the same system (0.18s), it would be a significant change, too. But from scientific point, there is a big variance of these results (± 30%), so it falls within the margin of error.

Going in, it was expected that Jetpack would be adding quite a few database calls, But as it turns out, even turning on every single Jetpack module only adds 10 SQL read queries to the blog homepage. Again, it might seem a lot, compared to a plain WordPress, but in nature, WordPress home page can have anywhere from several dozen to 200+ SQL read queries, so it is not really that significant.

Activating all Jetpack modules will add 13 (11 internal, 2 external) scripts and 11 internal styles and that is quite a lot. But this is being done by only a handful of modules: Infinite Scroll, Gravatars, Photon, Likes).

The last number that was being measured was remote HTTP requests. As some parts of Jetpack are based on external API's, reseaerch team was expecting quite a few of those. But at least on the home page, no remote requests were happening.

Conclusions

1. Jetpack will add at least 0.5 Mb (and up to 2.5 Mb) of peak memory usage on home page of a WordPress site;

2. There is an effect on execution times, too, but a bigger sample size is needed for reliable comparison;

3. Only a handful of Jetpack modules load additional assets to WordPress home page;

4. Jetpack adds no remote HTTP requests on WordPress home page;

5. Measuring only the homepage is not enough. A lot of the modules do their dirty work in other places of the site, like single page or the admin dashboard.

References

1. W3Tech. Usage of content management systems for websites.[interactive] [retrieved 2017-03-01]. <https://w3techs.com/technologies/overview/content_management/all >.
2. WordPress.org plugin directory [interactive]. [retrieved 2017-03-01]. < https://wordpress.org/plugins/ >
3. Jetpack [interactive]. 2016 [retrieved 2017-03-01]. < https://jetpack.com/ >
4. Query Monitor [interactive]. 2016 [retrieved 2017-03-01]. < https://wordpress.org/plugins/query-monitor/ >
5. Pingdom Tools [interactive]. 2016 [retrieved 2017-03-01]. < https://tools.pingdom.com/ >