

Figure out why WordPress is slow



- [Installation](#)
 - [Step 1 - Profile the WordPress load process](#)
 - [Step 2 - Dive into a specific stage](#)
 - [Step 3 - Profile specific hooks](#)

`wp profile` is a WP-CLI command to help you quickly identify what's slow with WordPress. It's designed to work alongside Xdebug and New Relic because it's easy to deploy to any server that has WP-CLI. With `wp profile`, you gain quick visibility into key performance metrics (execution time, query count, cache hit/miss ratio, etc.) to guide further debugging.

Installation

First, install the `wp profile` package if you haven't already:

```
$ wp package install wp-cli/profile-command
```

Step 1 - Profile the WordPress load process

Dealing with a slow WordPress install you've never worked with before? Run `wp profile stage` to see metrics for each stage of the WordPress load process. Include the `--url=<url>` argument to mock the request as a specific URL.

```
$ wp profile stage --url=runcommand.io
+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+-----+
```

stage	time	query_time	query_count	cache_ratio	cache_hits	cache_misses	hook_time	hook_count	request_time	request_count
bootstrap	0.7597s	0.0052s	14	93.21%	357	26	0.3328s	2717	0s	0
main_query	0.0131s	0.0004s	3	94.29%	33	2	0.0065s	78	0s	0
template	0.7041s	0.0192s	147	92.16%	2350	200	0.6982s	6130	0s	0
total (3)	1.477s	0.0248s	164	93.22%	2740	228	1.0375s	8925	0s	0

When WordPress handles a request from a browser, it's essentially executing as one long PHP script. `wp profile stage` breaks the script into three stages:

- **bootstrap** is where WordPress is setting itself up, loading plugins and the main theme, and firing the `init` hook.
- **main_query** is how WordPress transforms the request (e.g. `/2016/10/21/moms-birthday/`) into the primary `WP_Query`.
- **template** is where WordPress determines which theme template to render based on the main query, and renders it.

Step 2 - Dive into a specific stage

In the example from above, bootstrap seems a bit slow, so let's dive into it further. Run `wp profile stage bootstrap` to dive into higher fidelity mode of a given stage. Use the `--spotlight` flag to filter out the zero-ish results.

```
$ wp profile stage bootstrap --url=runcommand.io --spotlight
```

hook	callback_count	time	query_time	query_count	cache_ratio	cache_hits	cache_misses	request_time	request_count
muplugins_loaded:before		0.1644s	0.0017s	1					

40%	2	3	0s	0	
muplugins_loaded	2		0.0005s	0s	0
50%	1	1	0s	0	
plugins_loaded:before			0.1771s	0.0008s	6
77.63%	59	17	0s	0	
plugins_loaded	14		0.0887s	0s	0
100%	104	0	0s	0	
after_setup_theme:before			0.043s	0s	0
100%	26	0	0s	0	
init	82		0.1569s	0.0018s	7
96.88%	155	5	0s	0	
wp_loaded:after			0.027s	0s	0
	0	0	0s	0	
+-----+-----+-----+-----+-----+					
+-----+-----+-----+-----+-----+					
total (7)	98		0.6575s	0.0043s	14
77.42%	347	26	0s	0	
+-----+-----+-----+-----+-----+					
+-----+-----+-----+-----+-----+					

Each stage is further segmented by `wp profile` based on its primary actions. For the bootstrap stage, the primary actions include `muplugins_loaded`, `plugins_loaded`, and `init`. You can also see some intermediate actions like `plugins_loaded:before` and `wp_loaded:after`. These intermediate actions correspond to script execution before (or after) actual WordPress hooks. They're pseudo hooks in a sense.

Step 3 - Profile specific hooks

When you've found a specific hook you'd like to assess, run `wp profile hook <hook>`. Include the `--fields=<fields>` argument to limit output to certain fields.

```
$ wp profile hook plugins_loaded --url=runcommand.io --fields=callback,time,location
+-----+-----+-----+
+-----+
| callback                                | time    |
location                                |
+-----+-----+-----+
+-----+
| wp_maybe_load_widgets()                | 0.0046s | wp-
includes/functions.php:3501              |
```

wp_maybe_load_embeds()	0.0003s	wp-
includes/embed.php:162		
VaultPress_Hotfixes->protect_jetpack_402_from_oembed_xss()	0s	
vaultpress/class.vaultpress-hotfixes.php:124		
_wp_customize_include()	0s	wp-
includes/theme.php:2052		
EasyRecipePlus->pluginsLoaded()	0.0013s	
easyrecipeplus/lib/EasyRecipePlus.php:125		
Gamajo\GenesisHeaderNav\genesis_header_nav_i18n()	0.0007s	genesis-header-
nav/genesis-header-nav.php:61		
DS_Public_Post_Preview::init()	0.0001s	public-post-
preview/public-post-preview.php:52		
wpseo_load_textdomain()	0.0004s	wordpress-seo-
premium/wp-seo-main.php:222		
load_yoast_notifications()	0.0016s	wordpress-seo-
premium/wp-seo-main.php:381		
wpseo_init()	0.0329s	wordpress-seo-
premium/wp-seo-main.php:240		
wpseo_premium_init()	0.0019s	wordpress-seo-
premium/wp-seo-premium.php:79		
wpseo_frontend_init()	0.0007s	wordpress-seo-
premium/wp-seo-main.php:274		
Black_Studio_TinyMCE_Plugin->load_compatibility()	0.0016s	black-studio-tinymce-
widget/black-studio-tinymce-widget.php:206		
Jetpack::load_modules()	0.0564s	
jetpack/class.jetpack.php:1672		
+-----+		
+-----+		
total (14)	0.1026s	
+-----+		
+-----+		

There you have it! We've discovered that `wpseo_init()` and `Jetpack::load_modules()` are collectively contributing ~100ms to every page load.

With `wp profile`, discovering why WordPress is slow becomes the easy part.

📄 #2

📄 cocobook-sync 📄 2026-10-28 11:01:14 UTC

📄 cocobook-sync 📄 2026-10-28 11:32:49 UTC