Shared Externally

OpenCensus Metrics for PHP

About the PHP runtime

PHP is an odd one in the sense that a standard “PHP server” or “PHP Application binary” does not exist. PHP is a scripting runtime which can be bundled into web servers (Apache+mod_php), command line scripts (php-cli), cgi servers (php-fpm) and even embedded into C/C++ applications (php_embed). Most important implication of this is that we can’t make guarantees on the lifetime of a PHP process and even about which PHP process handles what request (as typically multiple PHP processes work together to serve PHP requests and which process handles which request is undetermined).

In a typical FPM or mod_php setting there will be many PHP processes on a single server or container scaling up and down given qps and request concurrency.

Inside PHP’s userland there is no concept of threads so basically all happens on the critical path or after the critical path (response provided to requestor but script continues to run in background). This might be ok for small tasks but on a global level in high performance sites this is a non starter.

PHP Daemon App

To come up with a solution that works for most deployment types we want to provide a simple sidecar application which will ingest raw metrics through unix sockets (or named pipes on Windows) from local PHP processes and route them forward to an OpenCensus Agent on the local network using its gRPC service. By not placing this logic into a PHP extension we only need one gRPC connection per container, php server, etc. and memory footprint of each PHP process will be lower as it doesn’t have to include gRPC for each one of them.

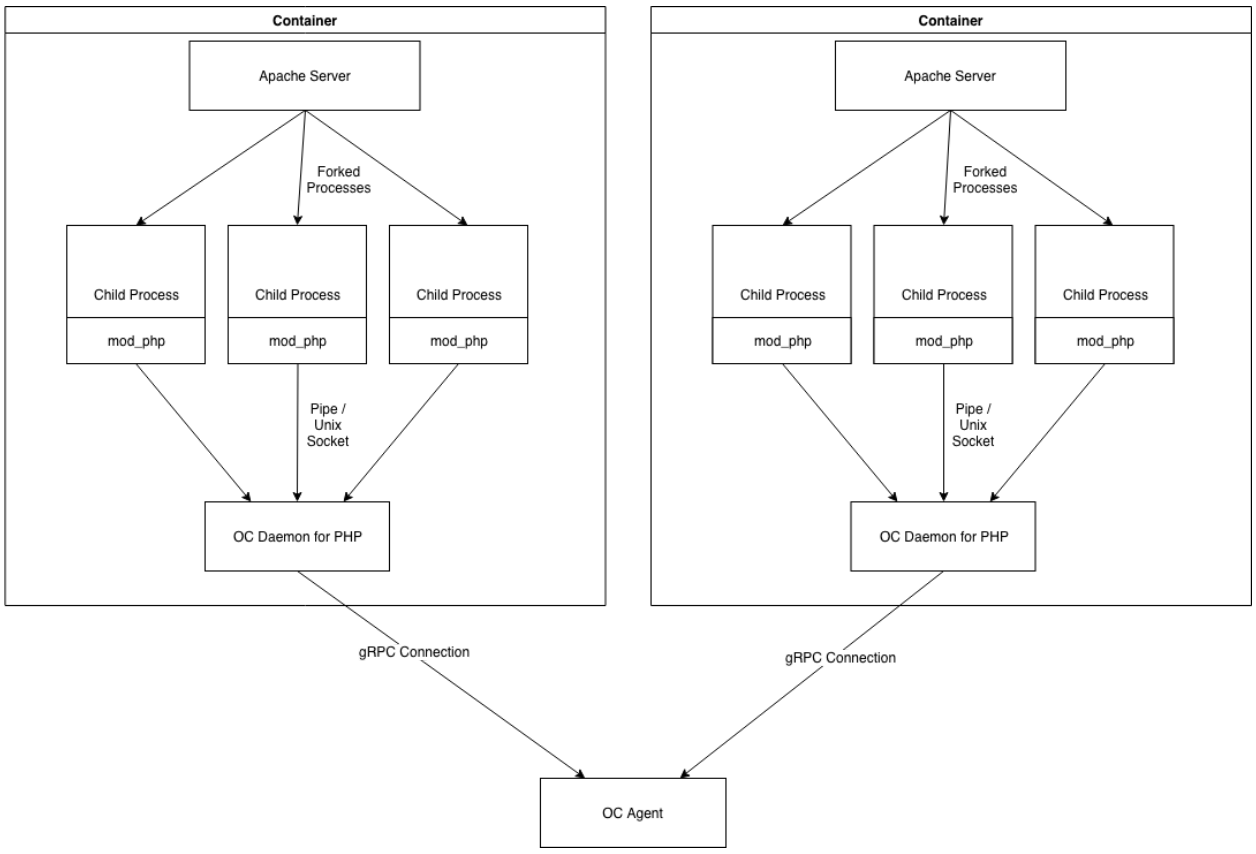
The Daemon will be written in Go as it will be easy to deploy for consumers given its cross platform no dependencies single binary nature.

If we can keep the messages sent over the unix sockets / named pipes simple and not overly specific we can even reuse this daemon for solutions other than PHP.

PHP Metrics API

By having a daemon app we can minimize the time spent on sending the raw metrics data out of process so a PHP userland API can work for a lot of deployments. If even more efficiency and/or performance is needed the existing OpenCensus PHP7 tracing extension can be enhanced to also handle the metrics API (overriding the userland one where it makes sense). Since a PHP extension can take processing out of band from the critical path and in more optimized code, it will even show less impact on request duration.

So the proposed solution in case of Apache+mod_php would look like this:



Components Involved

Given the above outline we would have the following items to work on:

- OC Metrics API (PHP Userland code)
- OC Daemon for PHP (Go application)
- OC Agent gRPC Metrics API
- OC Metrics API (low level PHP7 extension implementation)
- Examples
    - Plain/Vanilla PHP
    - Common PHP Frameworks
- Blog post(s)

Since we want to move forward quickly, we should first focus on building out the Metrics API for PHP userland and the OC Daemon for PHP. Initial support would be for Linux, Mac OS X, etc. We can add Windows support later if there is demand for it. Most larger scale PHP deployments are non Windows anyway.

In parallel we should work on adding support for ingesting metrics into the OC Agent project. This saves us having to write aggregation logic and implement exporters on the PHP Daemon end; keeping it lean and mean.

Next step would be to enhance the existing PHP7 extension to support handling the metrics API. This will provide performance gains for those running a modern PHP stack and able to add PHP extensions (using Pecl) to their deployment.

Examples are not to be underestimated. Current documentation for OC Tracing on PHP is minimal in the sense that the API is code documented but unclear to novices how to use except for the most basic example of adding a bare bones span.

Non Goals (for now)

Shared hosting environments. Most shared hosting environments do not allow for custom PHP extensions to be loaded. For any performance critical PHP applications we expect customers to run their own non shared PHP environments and expect them to install the OC PHP7 extension for improved tracing and metrics handling efficiency.