testing encyclopedias in production

effie mouzeli • giuseppe lavagetto
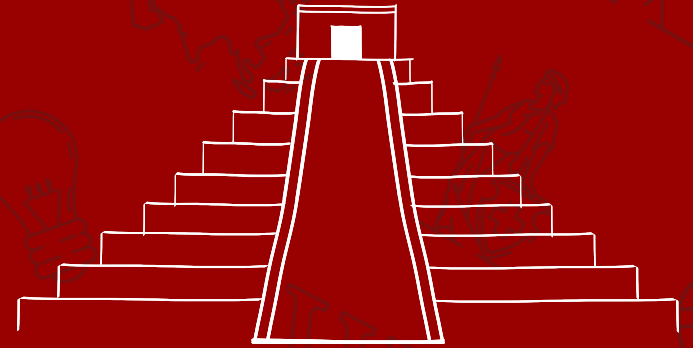
SREcon20 Americas
@Joe0blivian • @manjiki

WIKIMEDIA
FOUNDATION

About

Did you know...

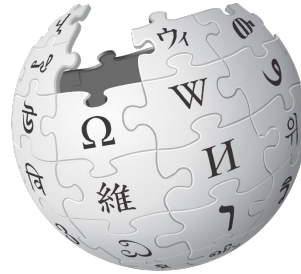- ... the **Wikipedia infrastructure** is run by the **Wikimedia Foundation,** an American nonprofit charitable organisation?

- ... and we are ~430 people?

- ... and have no affiliation with other Wiki* websites?

- ... all content is managed by volunteers?

- ... we support 304 languages?

- ... Wikipedia hosts some really really weird articles?

- ... but can't be read in China?
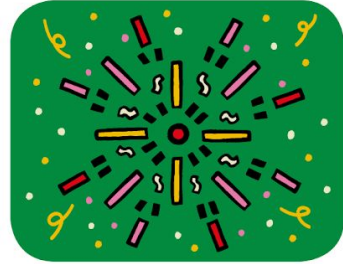
WIKIPEDIA 20

Wikimedia Projects

Production Overview

@Joe0blivian • @manjiki

WIKIMEDIA
FOUNDATION

Wikimedia Infrastructure

- **Open source software**
- 2 Primary Data Centres
- 3 Caching Points of Presence
- ~22 billion pageviews per month*
- ~300k new editors per month
- ~1300 bare metal servers

ulsfo codfw eqiad esams eqsin

WIKIMEDIA
FOUNDATION

* https://stats.wikimedia.org/#/all-projects

Production Overview (2017)
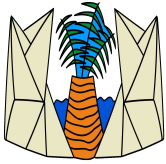
The Internet → Varnish → webserver Apache

edge cache

Apache → HHVM / MW [🌼]

application

8
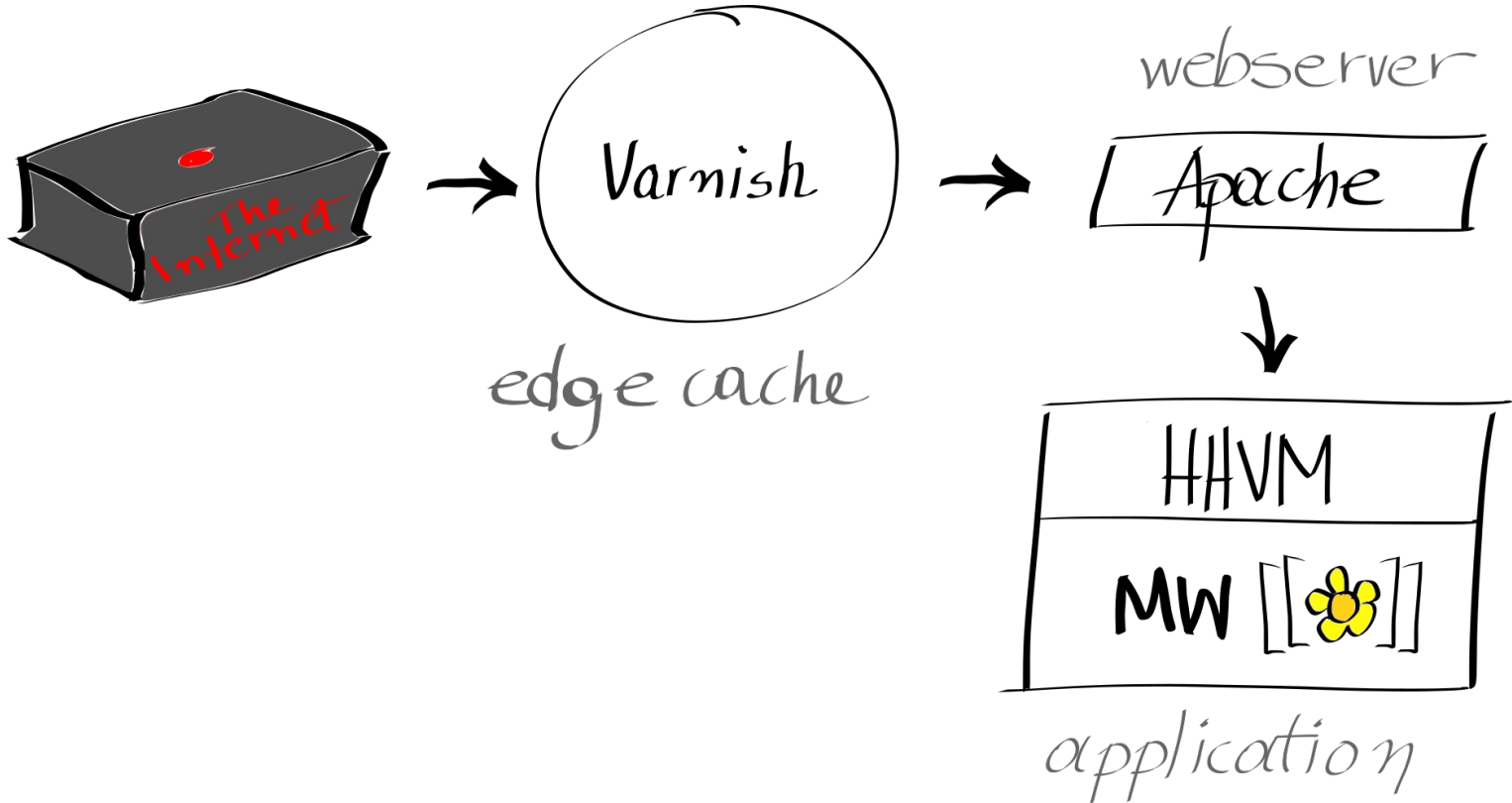
Edge Caches (2017)

- Varnish frontend (text+upload)
  - in memory

- Varnish backend (text+upload)
  - local stores

Varnish: Reverse HTTP caching proxy

Text (rw): static objects eg. HTML, CSS

Upload (ro): media like images, videos

WIKIMEDIA
FOUNDATION

MediaWiki [[🌼]]

- Our core application
- PHP, Apache, MySQL*
- **App** servers cluster (Web)
- **API** cluster
- **Jobrunners** cluster
- **Scheduled** tasks (crons)

MediaWiki is a **free** server-based software, licensed under the **GNU GPL.**

It is an **extremely powerful**, scalable software, and a feature-rich wiki implementation that uses **PHP** to process and display data stored in a database, such as **MySQL.**

WIKIMEDIA
FOUNDATION

* true story

HHVM (2017)

- Hip Hop Virtual Machine
  - Supports PHP5 and Hack
  - JIT compilation
  - Performant

  - Reduced CPU usage by 70%
  - Reduced latency by 30%

HHVM is PHP5/Hack execution engine developed by Facebook.

We were very happy with it since migrating to it 2014…

September 2017

Max Wang

The Future of HHVM

Posted September 18, 2017

Several months ago, PHP officially announced the end-of-life for PHP5.

The HHVM team is happy about the direction PHP has taken with PHP7, and we're proud of the role we've played in pushing the language and runtime to where they are today. Since the PHP community is finally saying goodbye to PHP5, we've decided to do so as well.

Our next LTS release, 3.24, will be cut about four months from now and will receive support for one year thereafter. It will also be the **last HHVM release that commits to PHP5 support**. This aligns with PHP's own timeline of sunsetting PHP5 at the end of 2018. HHVM users should note that while we won't drop support for every PHP5-specific quirk immediately, they will all be up for removal after the 3.24 release cut—and compatibility features like ext_zend_compat will eventually be deprecated and deleted.

It's clear, however, that PHP7 represents a substantive departure from PHP5. It changes a number of behaviors, some of which are not backward-compatible. At Facebook, meanwhile, we've used HHVM for years almost exclusively to run Hack. Hack had already addressed many of PHP5's shortcomings that PHP7 also fixes (as well as others that it does not), though Hack's solutions don't always match PHP7's.

PHP7 is charting a new course away from PHP5, and we want to do the same, via a renewed focus on Hack. Consequently, **HHVM will not aim to target PHP7**. The HHVM team believes that we have a clear path toward making Hack a fantastic language for web development, untethered from its PHP origins. We'd do ourselves and our users a disservice by positioning HHVM as an uncommon, less well-documented, less compatible PHP7 runtime.

We want to make HHVM and Hack a much better—and even more performant—developer experience. Our team has lots of features, libraries, and performance opportunities in the pipeline, and we want to take the rest of this post to share them with our community.

13

PHP-FPM

* PHP FastCGI Process Manager
    * PHP7 Support
    * Opcode Caching
    * Supposedly catched up with HHVM

PHP-FPM is a community project and is PHP-FastCGI implementation with additional features.

WIKIMEDIA
FOUNDATION

[Wikitech-l] HHVM vs. Zend divergence

Tim Starling tstarling at wikimedia.org
Wed Sep 20 07:05:12 UTC 2017

- Previous message: [[Wikitech-l] HHVM vs. Zend divergence](#)
- Next message: [[Wikitech-l] HHVM vs. Zend divergence](#)
- **Messages sorted by:** [date] [thread] [subject] [author]

```
On 19/09/17 10:13, Tim Starling wrote:
> I'll run a benchmark

I upgraded the test wiki container on my laptop from Ubuntu 14.04 to
16.04, which also necessitated a platform switch from schroot to
systemd-nspawn. The benchmark is thus approximately native performance
on a Core i5 4210U @ 1.7 GHz. I killed the usual CPU hogs so that
everything was quiet in top. Then I ran benchmarkParse.php on a copy
of the [[Australia]] article, including templates, with --loops 3.

The results were

PHP 7.0:     1.59 seconds
HHVM 3.21:   1.75 seconds

So PHP 7 was significantly faster on this test.

Note that I ran HHVM with JIT enabled; total wall clock time including
compilation and warmup was about 75 seconds, compared to 13 seconds
for PHP 7.

The test wiki has Scribunto with LuaStandalone. Debug logging was
disabled for this test.

-- Tim Starling
```

15

Warming Up

WIKIMEDIA
FOUNDATION

Main Challenges

- Functionality
  - Code coverage is not optimal - **expect dragons!**
  - Resource usage changes might make some pages un-renderable

- Performance
  - Will PHP-FPM be as performant at our **traffic level**?
  - Has our **code** been unconsciously **optimised** for HHVM?

- Observability
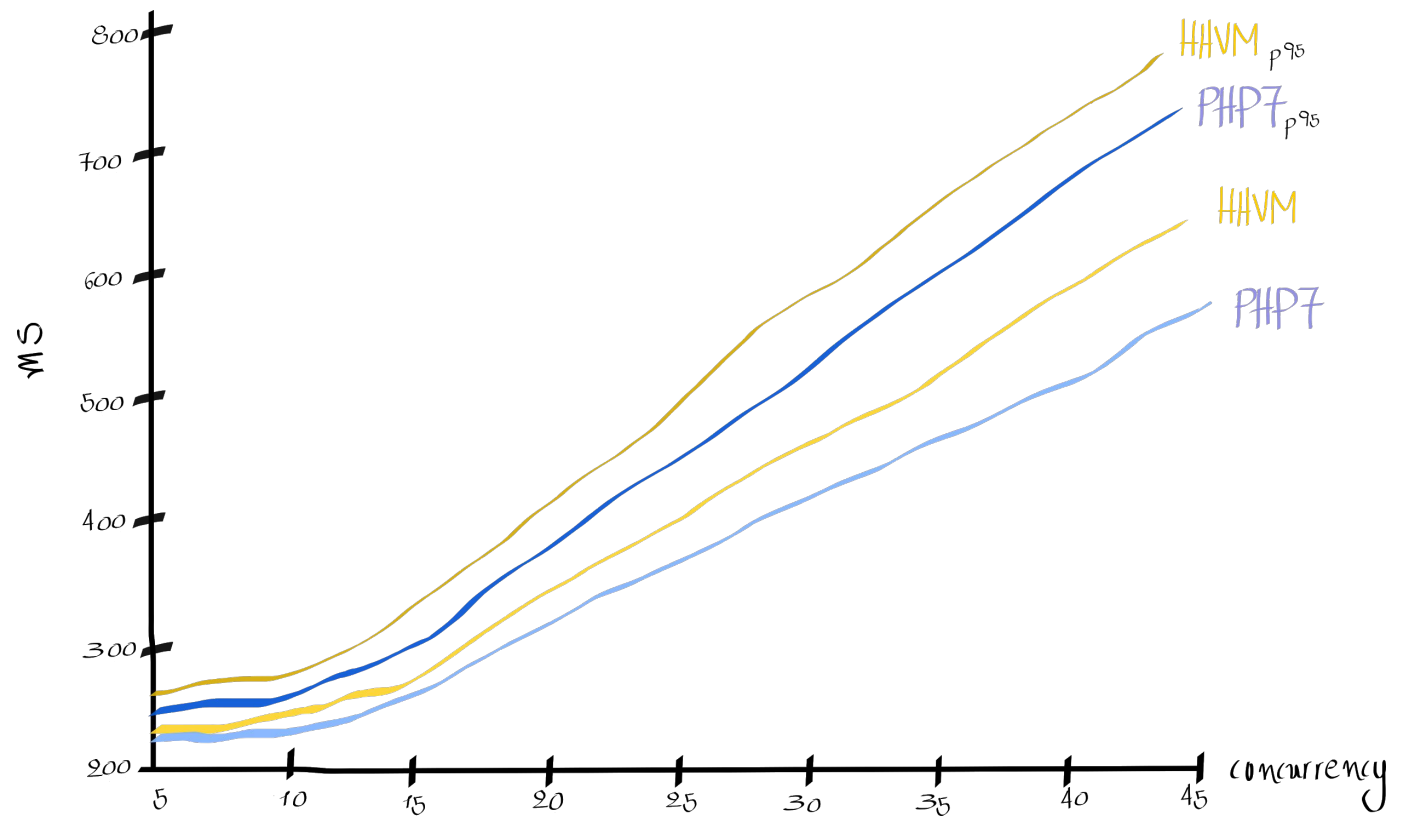 - PHP-FPM is **not as observable** as HHVM
  - Can we build comparably-good **metrics**?
  - No production ready **profiler**

WIKIMEDIA
FOUNDATION
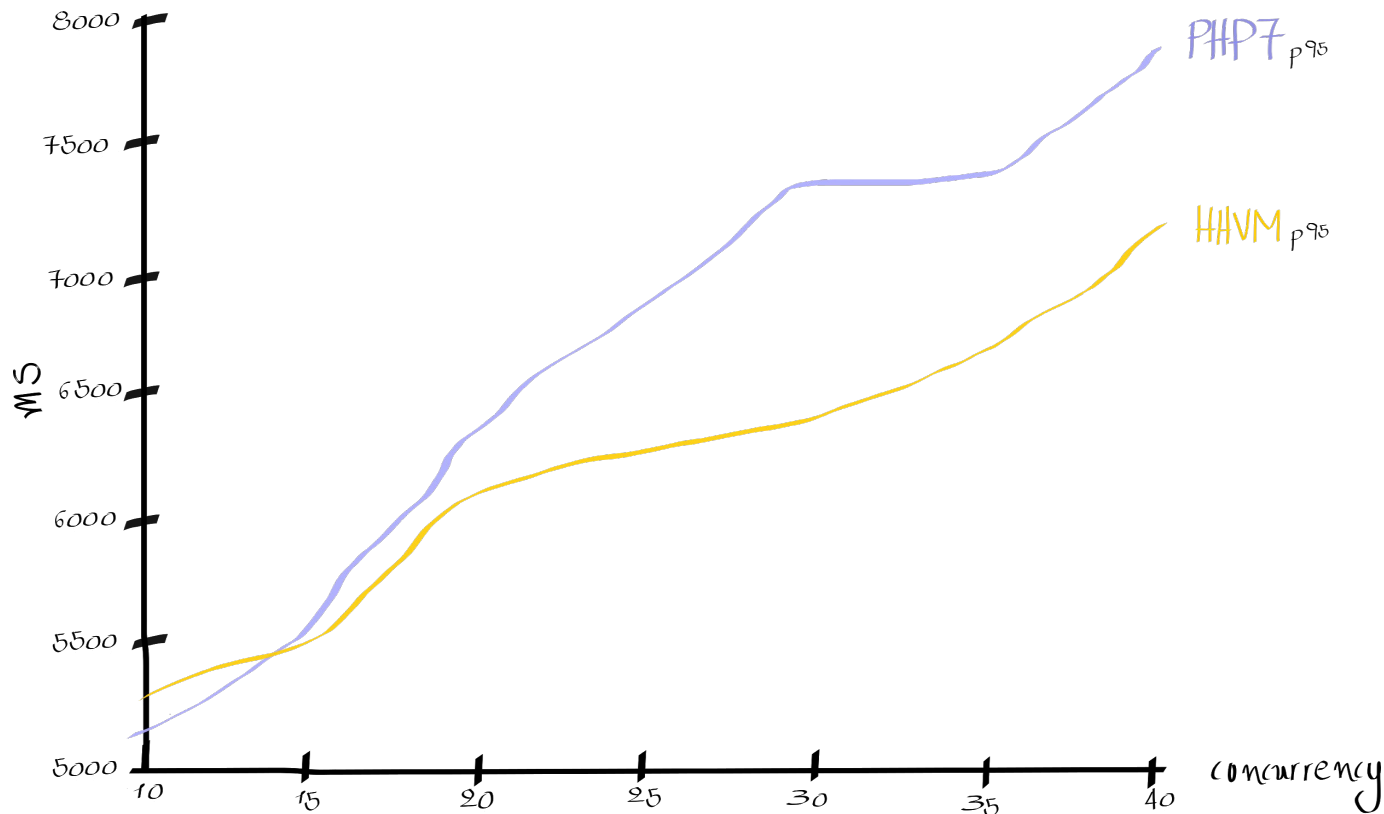
Setting it up

* Set a measurable goal
  * 100% of traffic eventually migrated
  * Performance should stay the same.
  * Get rid of HHVM

* Install PHP7
  * Co-exist with HHVM in all clusters
  * Route traffic when the PHP7 cookie is present

* .. and start measuring!
  * How many errors are generated by php-fpm?
  * What's the latency?
  * How much traffic are we pushing?
  * Collect profiling samples from php-fpm (excimer)

@Joe0blivian • @manjiki

WIKIMEDIA
FOUNDATION

Barack Obama (with cache)

Australia p95 (no cache)

Line chart. Y-axis labeled "MS" ranging from 5000 to 8000. X-axis labeled "concurrency" ranging from 10 to 40. Two lines: PHP7 p95 (purple) and HHVM p95 (yellow).

20

Testing in Production

@Joe0blivian • @manjiki

WIKIMEDIA FOUNDATION

CC BY-SA 4.0 User:Peellden
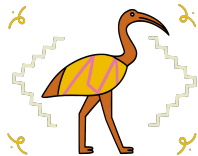
We test in production, like everyone!

- ✴ **Why?**
  - ✴ Unreplicatable traffic
  - ✴ Users are more effective in breaking production

- ✴ **How?**
  - ✴ Canarying
  - ✴ A/B testing
  - ✴ Phased rollouts
  - ✴ [[♡]]Volunteers

WIKIMEDIA
FOUNDATION

Rules of the game

- Minimum blast radius
- Easy to switch
- Easy to debug
- Initially users should be able to choose

Have a PHP7 cookie!

How?

* `PHP_ENGINE=php7`
* Manual tests
    * Send traffic to debug servers
* Wikimedia browser extension
    * choosing PHP engine from a menu
    * send traffic to debug servers
* Opt-in users
    * Enabling beta features

WikimediaDebug is a set of tools for debugging and profiling MediaWiki web requests in a production environment.

Have a PHP7 cookie!

What does it do?

✳ Cache slotting in varnish

```
// Detect client cookie indicating to use PHP7
unset req.http.X-Seven;
 if (req.http.Cookie "(^|;\s*)PHP_ENGINE=php7(;|$)"){
   set req.http.X-Seven = "1";
  }
```
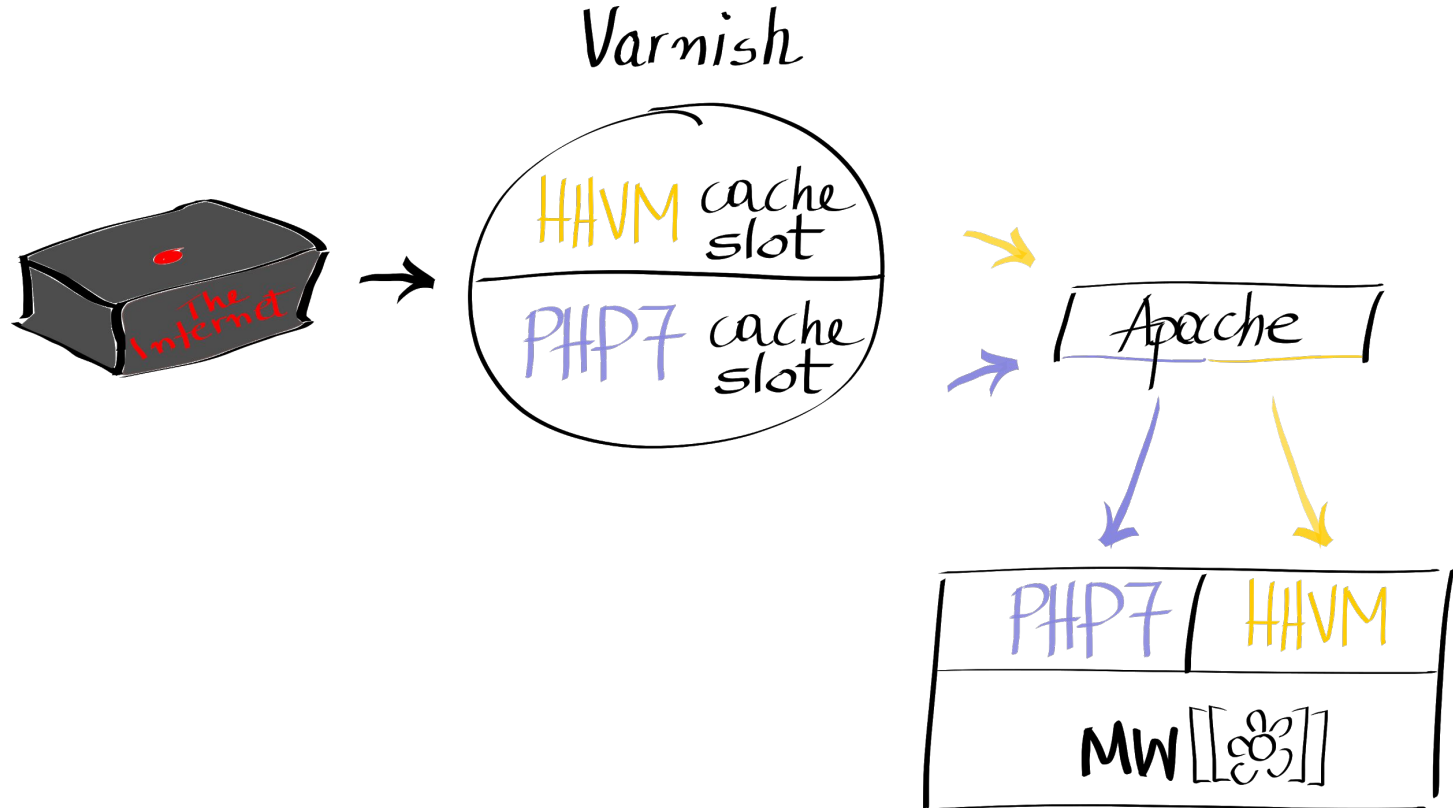
✳ Apache routing to PHP-FPM

```
SetEnvIf Cookie "PHP_ENGINE=php7" backend=php7
```

@Joe0blivian • @manjiki

Cache slotting: divide the cache using the `Vary: X-Seven` header, so a PHP7 user will not view an HHVM rendered page.

WIKIMEDIA
F O U N D A T I O N

Transitional Production

Rolling out

@Joe0blivian • @manjiki

WIKIMEDIA
FOUNDATION

CC BY-SA 2.0 User:Benebiankie

Anonymous users (that's you!)

* 43 servers

* Not all users accept cookies

* Control the percentage PHP7 users

* Gradual traffic increase

Issues:

* Performance hit when `max_children` is reached

* Memory corruptions

WIKIMEDIA
FOUNDATION

A winding road

```
* c8c932f21 Beta Features: Add the new PHP7 beta feature to the whitelist
* 779e2257a Set wgWMEPhp7SamplingRate to 0
* 5a270bbeb Direct 0.1% of anonymous users to php7
* 88984d4d2 Send 1% of anonymous users to PHP7.2
* 8afada66e Send 5% of anonymous users to PHP7.2
* 1027b78de Disable the PHP7 beta feature
* 12e7e067f Switch off php7 for investigation of production instabilities
* afc97c0bd Revert "Switch off php7 for investigation of production instabilities"
* cee99d4ca Move 10% of traffic to php7
* 4ffc48ff5 Send 20% of anonymous users to PHP7.2
* 1b3990ef7 Send 33.3% of anonymous users to PHP7.2
* 7efa56c1f Revert "Send 33.3% of anonymous users to PHP7.2"
* 559c8afb1 33.3% of anonymous users via PHP7.2
* fa81b83d7 50% of anonymous users via PHP7.2
* 2723f44f1 Enable coredump for some mysterious php7.2 failure
```

A winding road

```
commit 11d6db5d7e4bbcf61b9f6f54c61b93a824732fcd
Author: Effie Mouzeli <effie@wikimedia.org>
Date:   Tue Sep 17 10:49:55 2019 +0300

    100% of anonymous users via PHP7.2

    Of the 5 stages of migrating to PHP7, I hope this is Acceptance.

    Bug: T219150
    Change-Id: I20c0b5046030cc1574fe84c2ab4d9d73ec030fa9
```

API users

- 47 Servers

- A few API clients supported cookies

- Introduced `php7_only` feature flag

- Converted servers to PHP7

  - Each server serves ~2% of API traffic

Issue:

While the rollout was mostly without surprises, it doesn't allow to ensure a consistent experience for your users.

WIKIMEDIA
FOUNDATION

The long tail: jobs

- Asynchronous and scheduled (cron) job execution
- Lower visibility on hidden issues
- Job-by-job migration
- Enabled `php7_only` on all

Issue: None (that we know of)

WIKIMEDIA
FOUNDATION

Cleaning up

* Mediawiki code

* Config management code

* Remove transitional code

* Polish dashboards

* Server re-installation

WIKIMEDIA
FOUNDATION

Choosing our Battles

@Joe0blivian • @manjiki

WIKIMEDIA
FOUNDATION

The bugs that haunt us

Confession

We forward-ported a bug in PHP's unicode tables.

```
$ hhvm --php - r 'echo mb_strtoupper("dž\n");'
dž
$ php7.2 -r 'echo mb_strtoupper("dž\n");'
DŽ
```

Every Article gets capitalized when building its URL.

`mb_strtoupper` is called on the URL's first unicode character.

HTTP connection pooling
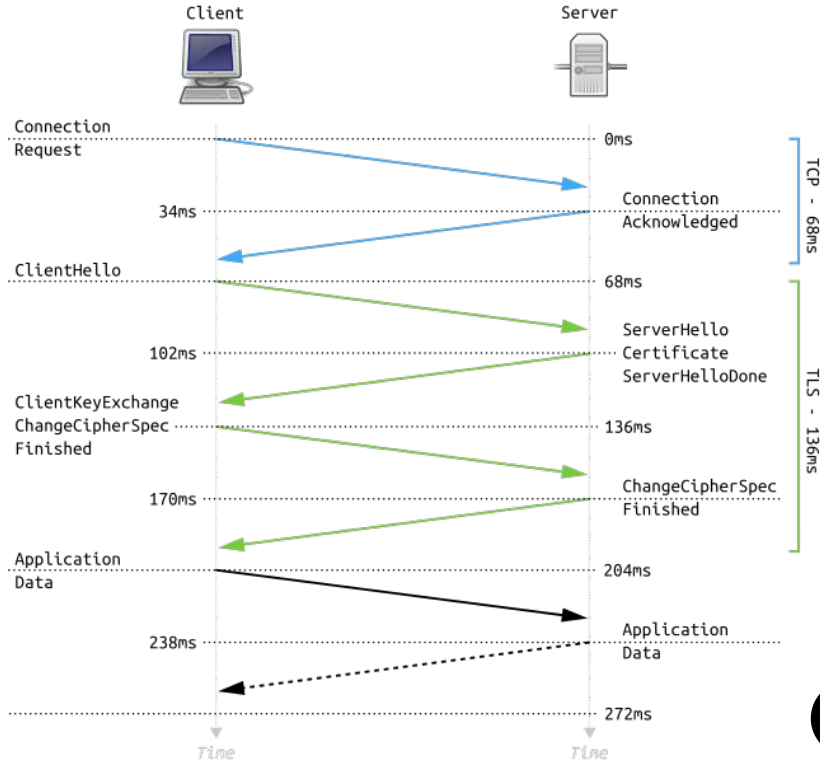
Client		Server	
Connection Request		0ms	TCP - 68ms
	34ms	Connection Acknowledged	
ClientHello		68ms	TLS - 136ms
	102ms	ServerHello Certificate ServerHelloDone	
ClientKeyExchange ChangeCipherSpec Finished		136ms	
	170ms	ChangeCipherSpec Finished	
Application Data		204ms	
	238ms	Application Data	
		272ms	
Time		Time	

HTTPS is used for **cross datacentre** application communication.

Renegotiating a TLS connection across a laggy link is **very expensive**.

HTTP connection pooling

Issue: php-fpm does not support connection pooling for service-to-service communication

Solution: Introduced Envoy, improved RPC performance over HHVM.

(Blog post at **https://w.wiki/mDE**)

HTTPS is used for **cross datacentre** application communication.

Renegotiating a TLS connection across a laggy link is **very expensive**.

Envoy is a high-performance open source edge and service proxy.

WIKIMEDIA
FOUNDATION

Runtime profiling

Issue: we lack the ability to collect sampled profiling data and to set wall-clock timeouts

Solution: `php-excimer`, a PHP extension that allows to sample stack traces at regular intervals, and also to set wall-clock time limits for PHP script execution.
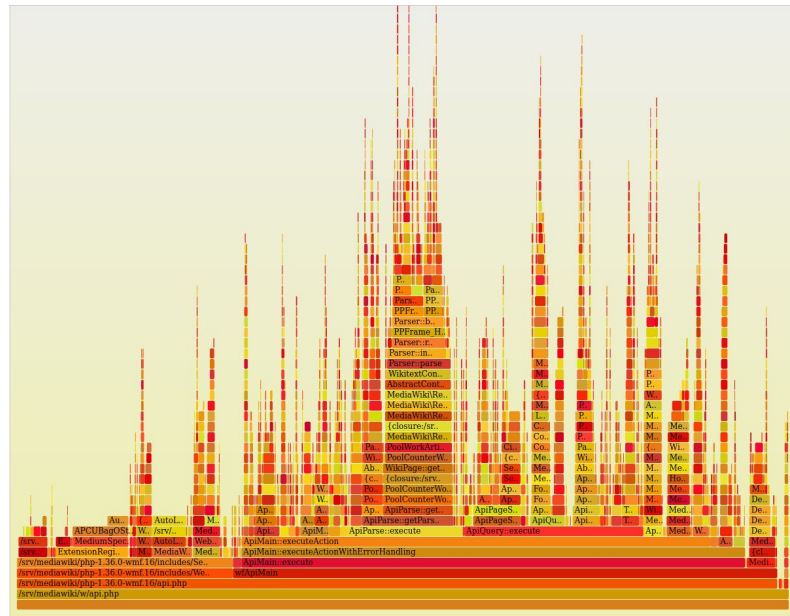
Future

@Joe0blivian • @manjiki

WIKIMEDIA
FOUNDATION

Migrating MediaWiki to Kubernetes

Why #1?

- ☀ Kubernetes is designed for 19-year-old monoliths ;)

- ☀ For microservices, Kubernetes has served us well

WIKIMEDIA
FOUNDATION

Migrating MediaWiki to Kubernetes

Why #2?

✸ Repay our technical debt

✸ Unify how we deploy code

✸ Enhance development experience

✸ Elasticity and flexibility

TL;DR

Thank you for testing encyclopedias in production!