Job Queues with Gearman

SoFloPHP Nov. 2014

Michael Moussa | @michaelmoussa

http://github.com/michaelmoussa/soflophp-gearman

Boca: https://joind.in/12749

Miami: https://joind.in/12750

About Me

- PHP developer for 15 years
- Lead Developer, ZAM Network
  - lolking.net, wowhead.com, destinydb.com
- Zend Certified PHP Engineer & ZF2 Certified Architect

In a nutshell

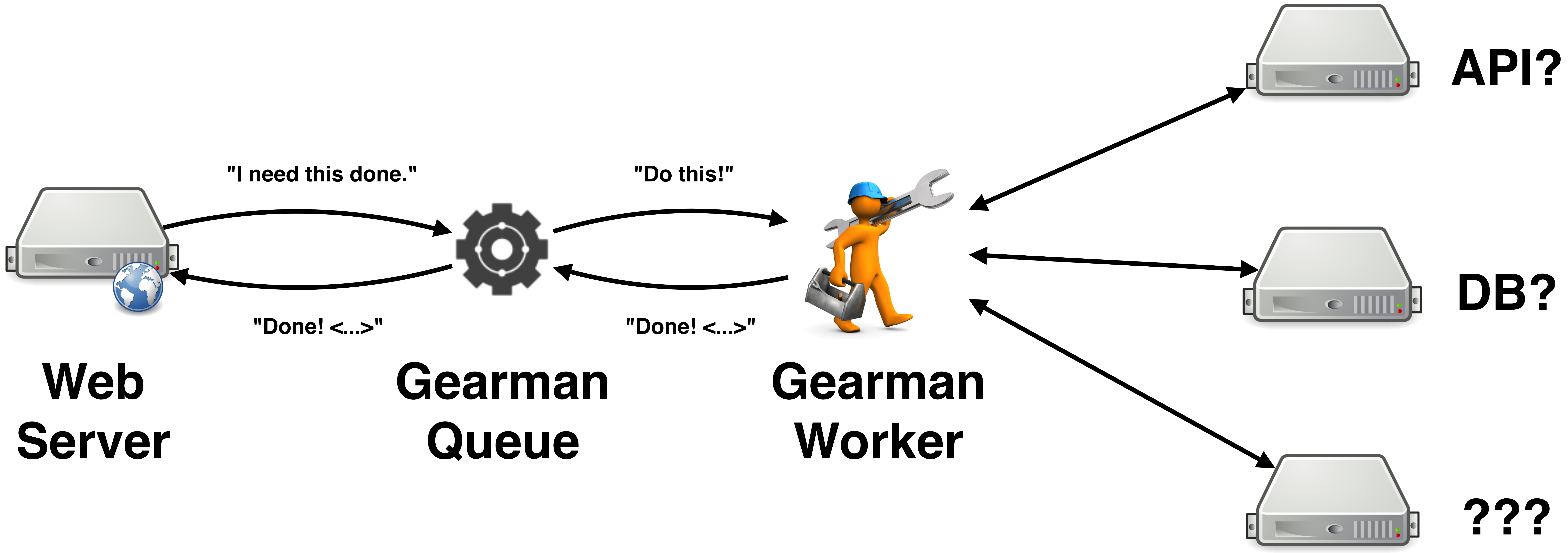
- What is Gearman?
- Why bother?
- Getting Started
- PHP API Overview
- Examples
- Unit Testing

“Gearman is a generic application framework for farming out work to multiple machines or processes.”

–http://php.net/manual/en/intro.gearman.php

But what does that mean?

Brief Overview



OK... but what problem are we trying to solve?

DEMO TIME!

Demo 1 - "Baseline"

```
$ ab -n 10000 -c 250 "http://192.168.133.71/demo/1.php"
...
Benchmarking 192.168.133.71 (be patient)
...
Requests per second:    620.39 [#/sec] (mean)
Time per request:       402.971 [ms] (mean)
...
```

Don't forget!

CPU Usage: ~13%

Demo 2 - "Doing work on the web"

```
$ ab -n 10000 -c 250 "http://192.168.133.71/demo/2.php"
...
Benchmarking 192.168.133.71 (be patient)
...
Requests per second:    57.22 [#/sec] (mean)
Time per request:       4368.977 [ms] (mean)
...
```

~10x worse!

CPU Usage: ~90%

Your boss

or sysadmin

Gearman can help me with this?

OK - where do I start?

Getting Started

Linux (Ubuntu)

```
$ apt-get install -y apache2 php5 php5-cli php5-dev \
      libgearman-dev gearman-job-server
$ pecl install gearman
$ echo "extension=gearman.so" > /etc/php5/apache2/conf.d/gearman.ini
$ echo "extension=gearman.so" > /etc/php5/cli/conf.d/gearman.ini
```

Linux (Other distros), OSX, Windows, etc...

http://gearman.org/getting-started/

PHP API Summary

- GearmanClient
- GearmanJob
- GearmanTask
- GearmanWorker
- GearmanException

GearmanClient

Connects to the job server and makes requests

```
$client = new \GearmanClient();
$client->addServer($host, $port);

// Send request for some work to be done at
// Normal priority.
$client->doNormal('foo', '{"hello": "world"}');
```

GearmanJob

Used to represent a unit of work that needs to be done, and to communicate back to the Gearman server.

```
// Grab the "workload", i.e. parameters, and do
// some work with them
$result = doSomethingWith($job->workload());

// Send back some indication of what happened
$job->send<_____>(...);
```

GearmanTask

Used in the client when receiving communication from server via a callback.

```
$client->setCompleteCallback(
    function (\GearmanTask $task) {
        doSomethingWith($task->data());
    }
);
$client->addTask(...);
$client->addTask(...);
$client->addTask(...);
$client->runTasks();
```

GearmanWorker

Used to register work functions and wait for jobs.

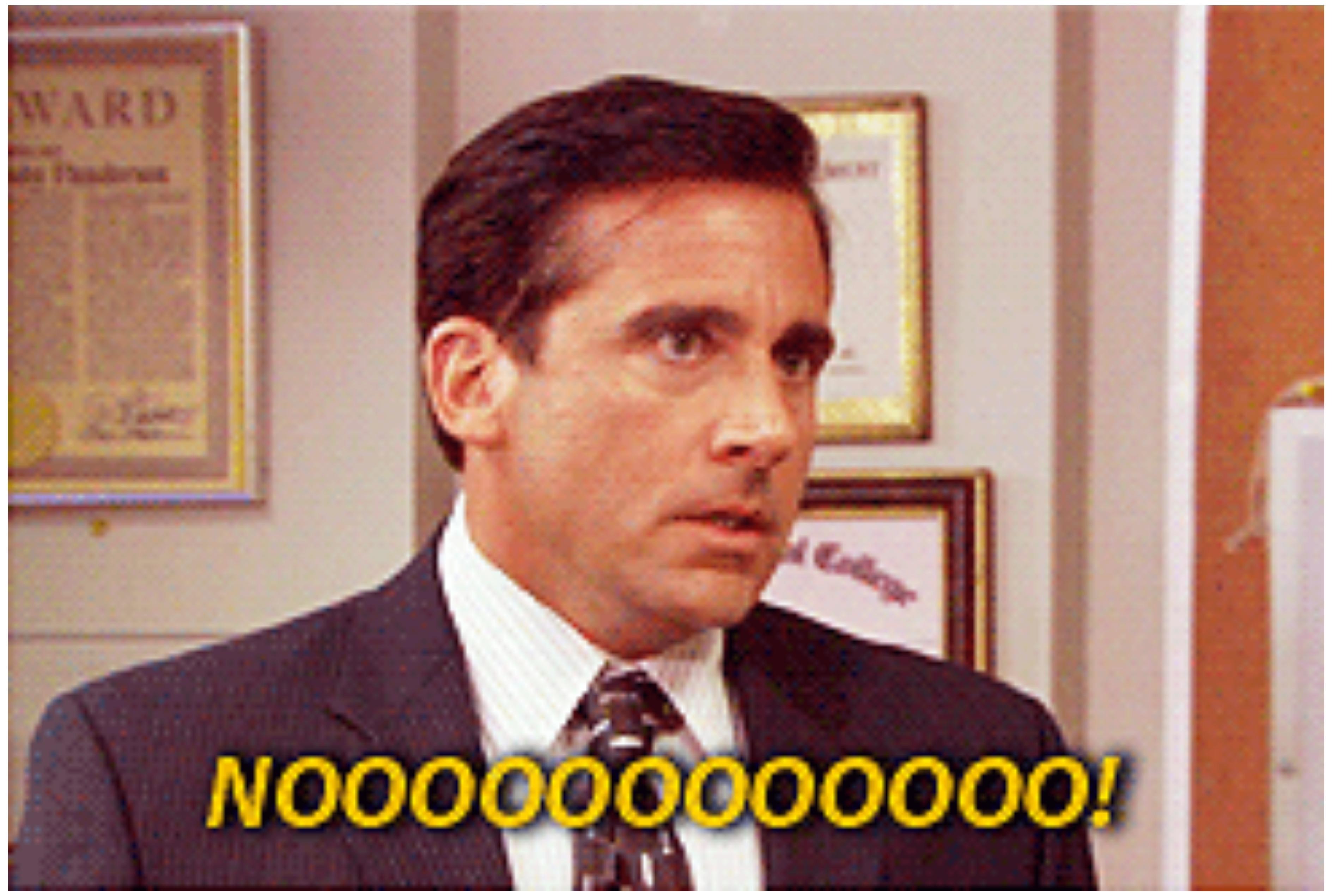
```php
$worker = new \Worker();
$worker->addServer($host, $port);

// Grab "foo" jobs and call "bar" with them.
$worker->addFunction('foo', 'bar');

// Never stop working!
while ($worker->work());
```

GearmanException

Used when something goes horribly wrong.



Don't worry... that won't happen to you.

Now that we've got all of that straight...

DEMO TIME!

Demo 3 - "Foreground worker"

```
$ ab -n 10000 -c 250 "http://192.168.133.71/demo/3.php"
...
Benchmarking 192.168.133.71 (be patient)
...
Requests per second:    0.47 [#/sec] (mean)
Time per request:       526542.888 [ms] (mean)
...
```

MANY times worse!
I stopped after ~30 requests

But look, web CPU is OK!

Web CPU Usage: ~1%
Worker CPU Usage: ~99%

Demo 4 - "Foreground workerS"

```
$ ab -n 10000 -c 250 "http://192.168.133.71/demo/4.php"
...
Benchmarking 192.168.133.71 (be patient)
...
Requests per second:    23.65 [#/sec] (mean)
Time per request:       10569.333 [ms] (mean)
...
```

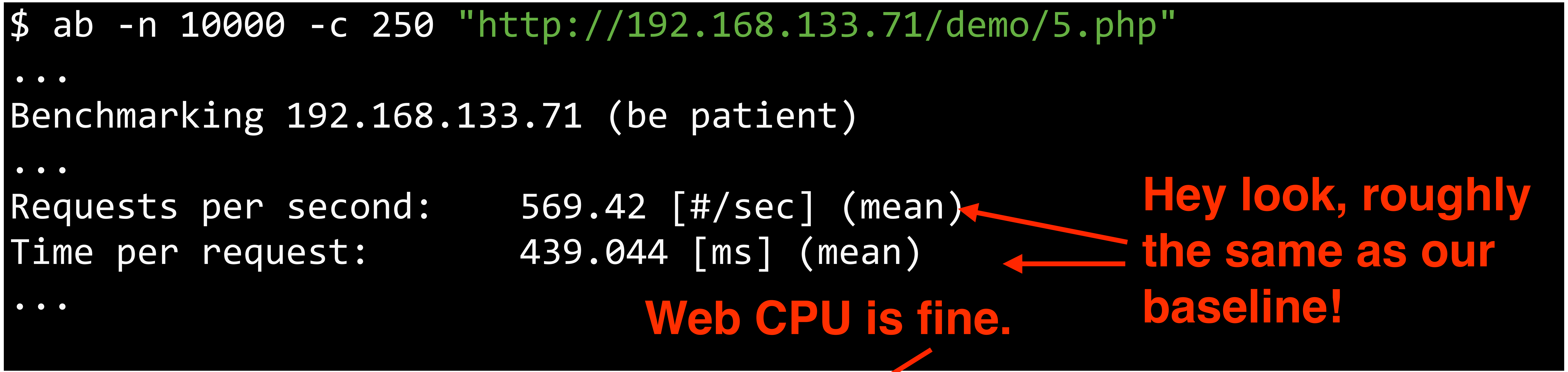
Better than demo #3, but still terrible.

Web CPU still OK.

Web CPU Usage: ~1%

Worker CPU Usage: ~99%

Demo 5 - "Background workerS"



```
$ ab -n 10000 -c 250 "http://192.168.133.71/demo/5.php"
...
Benchmarking 192.168.133.71 (be patient)
...
Requests per second:    569.42 [#/sec] (mean)
Time per request:       439.044 [ms] (mean)
...
```

Hey look, roughly the same as our baseline!

Web CPU is fine.

Web CPU Usage: ~13%

Worker CPU Usage: ~99%

We'll talk about this later.

Quick Recap

- Don't have your web application do expensive work unless the site *requires* it and the user *expects* it to be slow.
- Run multiple workers to distribute work.
  - http://supervisord.org/
- Use background jobs whenever possible.
- Gearman is a tool, not magic. Be smart!

Wait, what about that high worker CPU?

- It's not *ideal*, but don't worry too much about CPU alone.
- Concern yourself more with *load average*.
    - http://blog.scoutapp.com/articles/2009/07/31/understanding-load-averages
- Scale out by adding more worker servers if necessary.
- It was just a demo. Your site's traffic will be much more reasonable.
- Ask your sysadmin

Tips, Tricks, and Caveats

- Use the `$uniqueId` parameter when possible.
  - e.g. `md5($functionName . $workload);`
  - Prevents queueing if an identical job is pending
- Prioritization does not work intuitively.
  - Only takes effect if functionName is the same.
  - Forget about "functionName". Pretend it doesn't exist. Think of it as **QUEUE** name!
- Don't forget - the worker and client talk to each other via the server.
  - Keep your messages light. Don't give the Gearman job server more work to do than it needs.

Testing

Mocks, right?

```
$job = $this->getMock('GearmanJob');
$job->expects($this->once())
    ->method('functionName')
    ->will($this->returnValue('foobar'));
$job->expects($this->once())
    ->method('workload')
    ->will($this->returnValue('{"foo": "bar"}'));
```

No. Please don't do that.

Consider this instead

```
class TestGearmanJob extends \GearmanJob
{
    public $functionName;
    public $workload;

    public function functionName()
    {
        return $this->functionName;
    }

    public function workload()
    {
        return $this->workload;
    }
}

$job = new TestGearmanJob();
$job->functionName = 'foobar';
$job->workload = '{"foo": "bar"}';
```

Create once in
your test directory

Now you can avoid
a ton of copy & paste

Don't forget!

Boca: https://joind.in/12749

Miami: https://joind.in/12750

So... any questions?