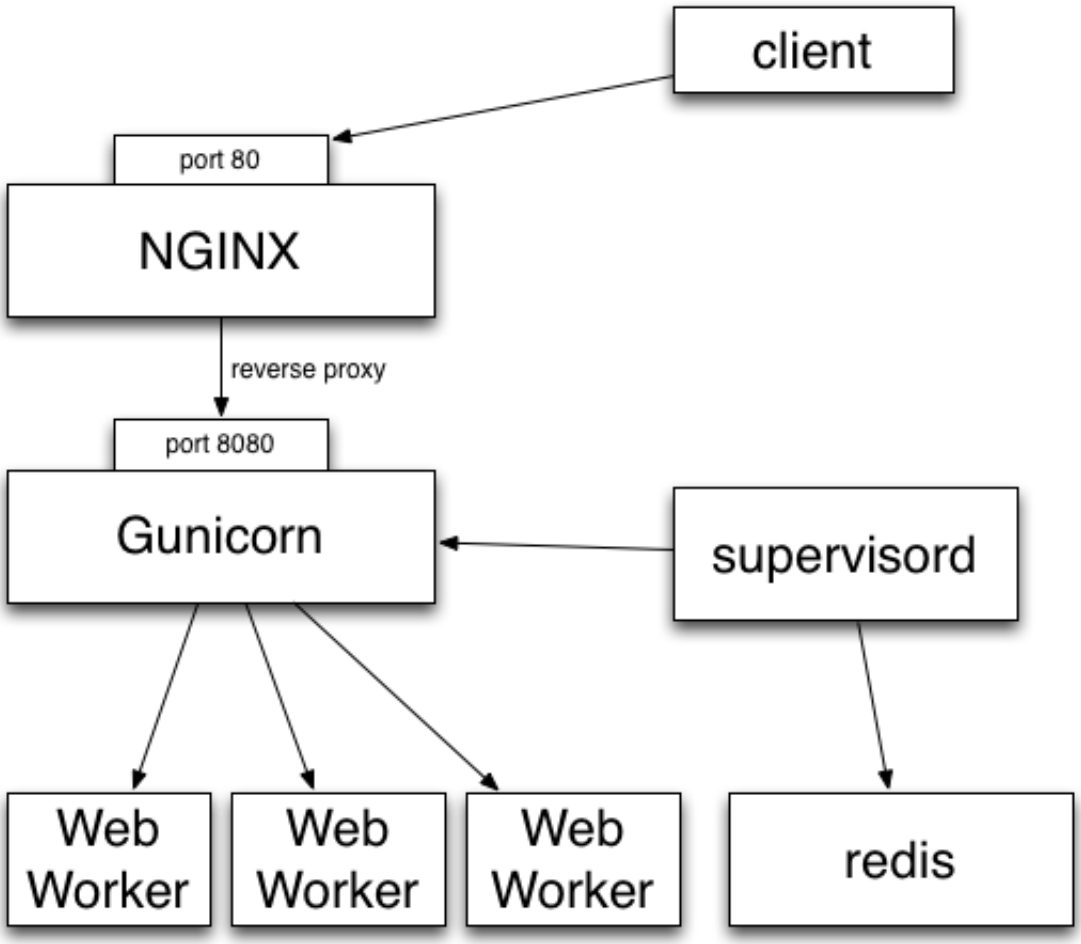
Web服这个屎盆子不够装了——记inetd的现代版Mozilla Circus

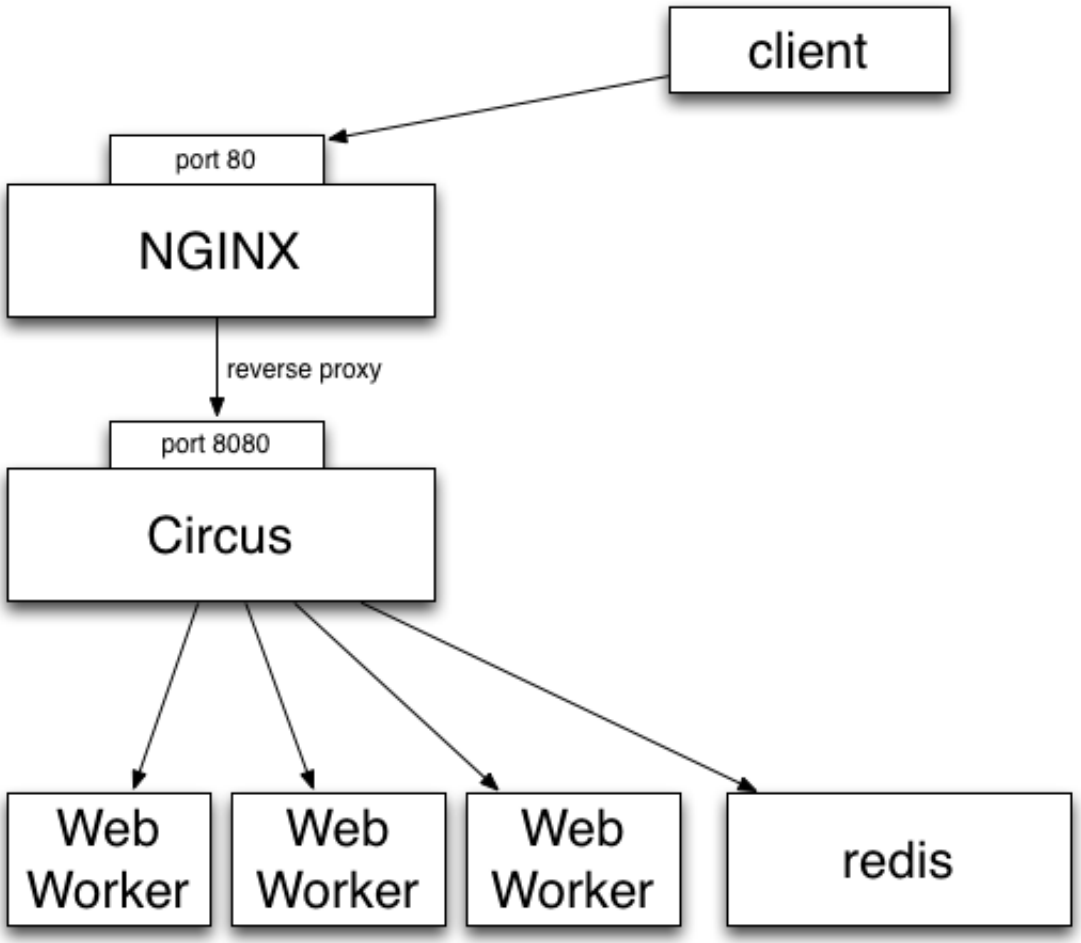
今天瞄了下Mozilla Circus这个项目，黑有意思。

简单的说，目前Web服务器基本模型都是一个总瓢把子Nginx，后边一大堆苦力worker进程。比如Python世界的Gunicorn是这个样子的粗暴简单：



worker基本都是日出而耕，日落而息。（顺便一喷，Supervisord这些都弱爆了。djb大牛的runit才是王道）

要做到聪明的运维，和可小可大硬又黑的HA scaling，就需要一个高级一点的worker进程管理器于是——哐当当，Mozilla的Circus闪亮登场！



中文介绍在这里Mongrel2&Circul = web栈的完全控制

我觉得这项目很不错，至少有了Web Console码农不用再去砸400多元的机械键盘去 `kill -HUP`和 `cat pidfile` 啪啪啪了

但是Circus只这样就太大材小用了。都进程了，那么socket、fd也一并托管了吧。以后业务逻辑就只管去轮fd就是了。这就是einhorn项目的思路。

同时我突然觉得Mongrel2这个项目有点2，居然是用json把http完整序列化一遍再pubsub分发到处理点。。。。这效率。。

Circus的让我想起uWSGI。大部分核心特性都是重合的。uWSGI 1.3其实也支持Circus直接绑定fd了。uWSGI甚至支持adaptive process spawning，连Heroku那种手动设定worker数量的机制都弱爆了么。。

uWSGI是个好坑，我以前准备详细写写的。uWSGI还有一种绝世内功——统一、灵巧的signal机制达到集群能力。

所以这个uWSGI、circus的轮子核心还得看这个控制信道和数据信道的设计是否良好分离。uWSGI还提供了一些奇技淫巧，比如类似Redis/Memcache的in-process key-value store，跨语言的RPC机制，一个timer和cron。（随便喷：只能master进程启动的时候初始化timer和cron。。。wtf？）

再进一步思考：客户端海量请求的合理分离，这只是水平切割，那么垂直切割呢？比如，在两个地方的服务器集群，同时渲染部分模板，然后通过circus输出为最终版本……当然这个太YY了。

这些都是Circus有希望解决的。

看到这里，我突然想起了一个老物——inetd

历史真的是个轮回

#python #web #stack #wsgi

2 notes Mar 12th, 2013