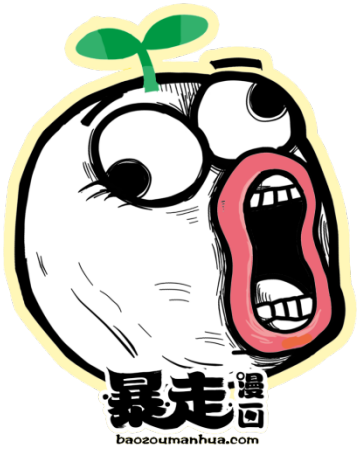
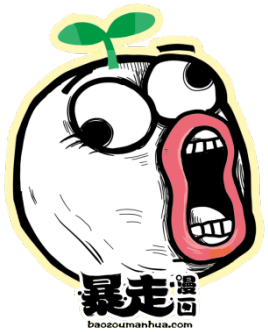


[redacted] 程序员如何一个人打造日PV百万的网站架构

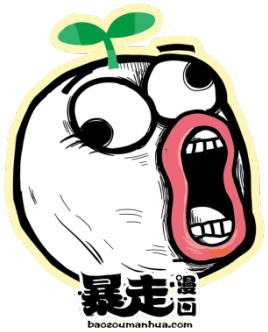
曹力 @ShiningRay

本人经历

- 2008年~2011年维护过糗事百科
- 2011年~2012年创办过博聆网
- 2012年~2013年暴走漫画



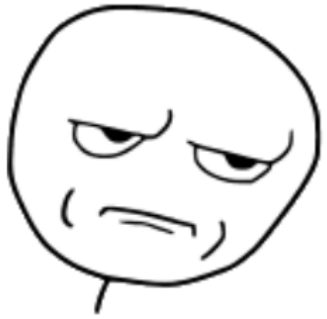
深受广大无节操朋友的喜爱



 的特征

穷！

勒紧裤带买个一两台服务器



- CPU吃紧（4核不错了）
- 内存吃紧（8G高端啊）
- 磁盘IO吃紧（RAID真奢侈）

野心大！

要赚大钱！
要逆袭有木有！

智商情商有限

为何选用Ruby/Rails？

因为别的我不会啊！

用户说：We don't care!

我们也有闪光点

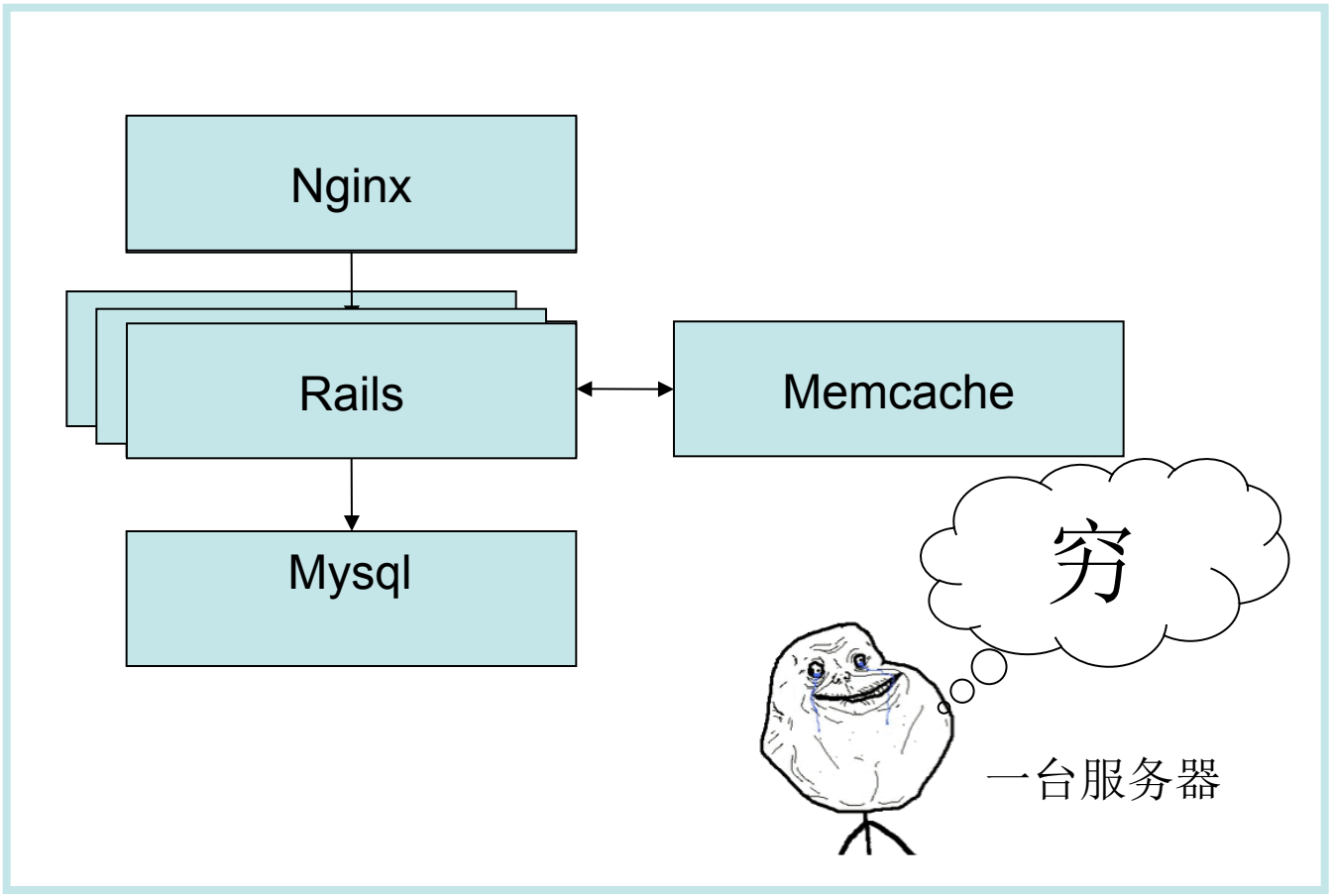
- 勤奋
- 坚持
- 有那么一点点聪明

Let's go !

回顾应用场景

- 功能类似Blog、留言板
- 用户以浏览为主
- 同一时刻大部分用户看到的内容大体一致
- 有一定的交互（投票，留言，私信）
- 需要SEO

初始的架构



初始的架构

勉强支持了每日1~10w PV的请求

目标每天100wPV

怎么破！？

难点

- Rails在大并发下效率差
- 每个页面都需要显示用户信息
- 每个文章的顶埋数量变化非常快

根据长期观察

- 50%~80%访客是不登陆的
- 首页等几个页面占了50%以上访问量
- 页面上80%内容是不变的，剩下的主要为用户相关信息

根据长期观察

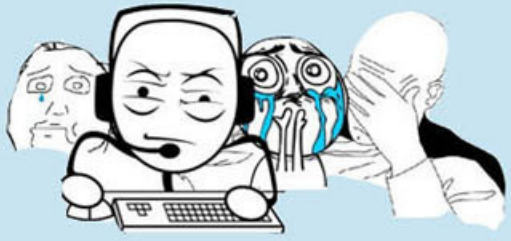
- 用户关注的核心是内容，这部分是变化少
- 未登录用户交互更少，也不关注数据的准确性

暴走漫画
baozoumanhua.com
因为你而存在的原创恶搞漫画社区

有变化的部分

审核组全体成员

致各位暴民的一封信



在历尽寒暑和暴民的蹂躏摧残下，审核组特意对暴走的稿子做出了以下绝逼不要的分类。大家感受下！

审核组全体成员致全体暴民的一封信

王尼玛

9158 -1337

350 收藏

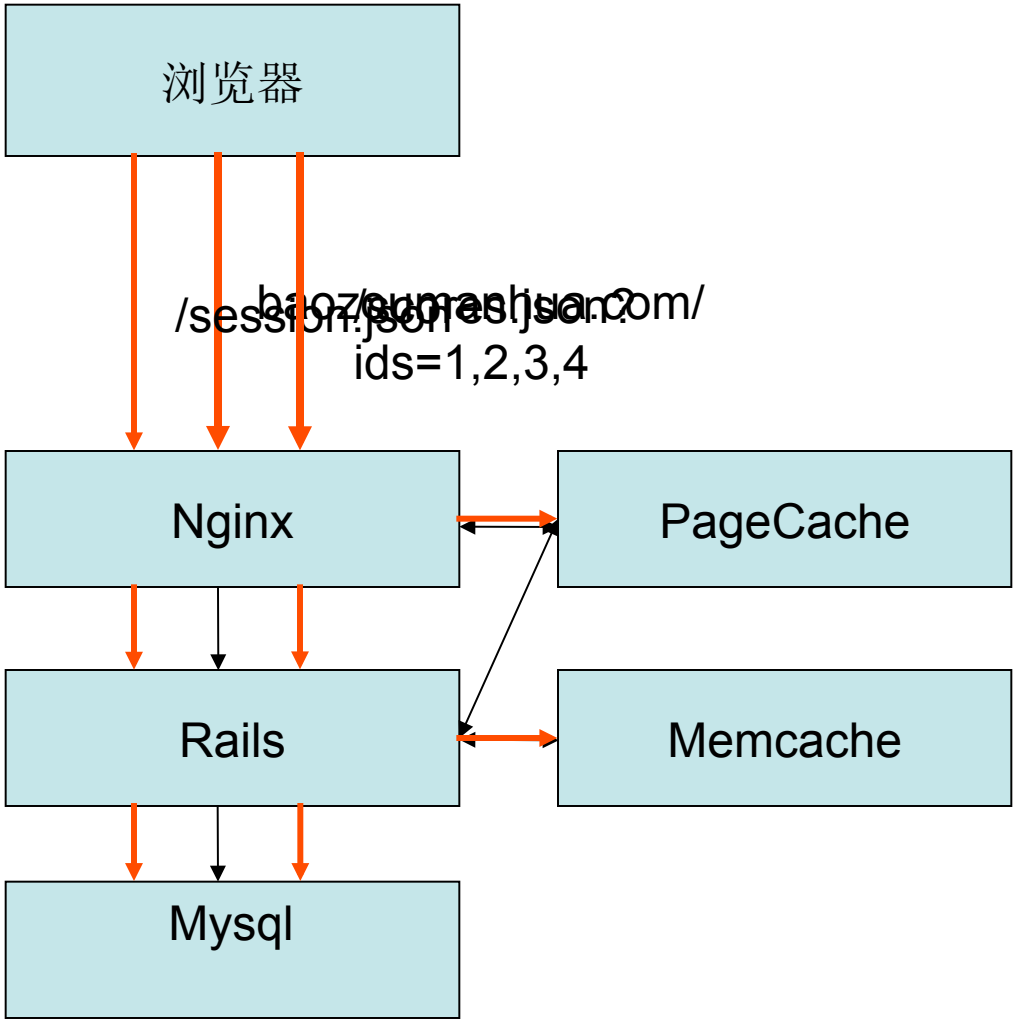
169条评论

暴走漫画活动 查看全部活动 »

寻找书中的你

策略

- 未登录用户直接返回缓存内容
- 分离页面上的静态内容和动态内容
  - 先载入相对不常变的缓存的内容
  - 然后加载经常变化的内容


浏览器
/session.json?test.json?
ids=1,2,3,4
Nginx
PageCache
Rails
Memcache
Mysql

异步载入优势

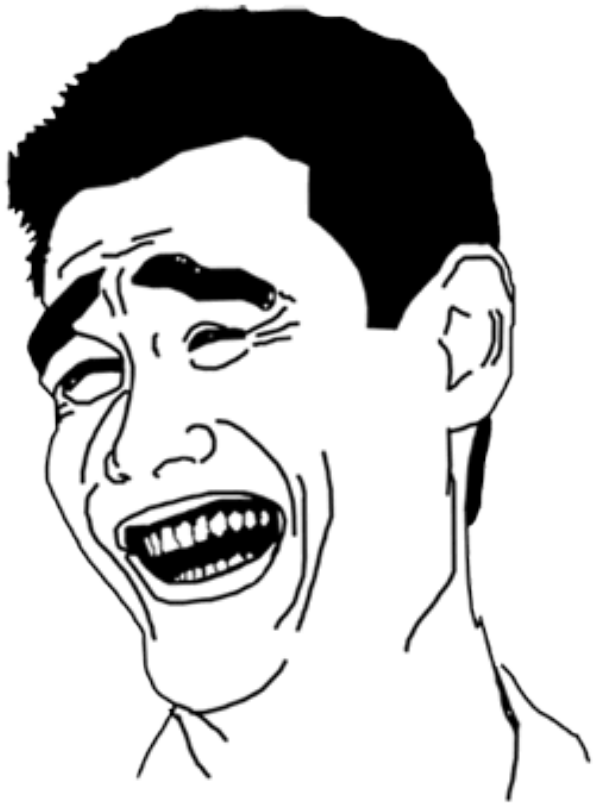
- 最快速地让用户看到他们最希望看到的页面内容
- 子请求不进行模板渲染
- 子请求仍然可以进行缓存
- 子请求可以和HTTP API放在一起实现（可单独进行优化，如使用其他框架或语言开发）

简单估算

首页等页面可缓存的页面占80%访问量=80w PV

- 即，实际只有20w的请求是需要单独处理的完整页面
- 80%的用户是非登录用户，80w实时数据请求可以被（短时）缓存。
- 20w用户信息的请求（亦可缓存）

一台服务器一天100wPV
挑战成功



计算有错误？

不要在意这种细节

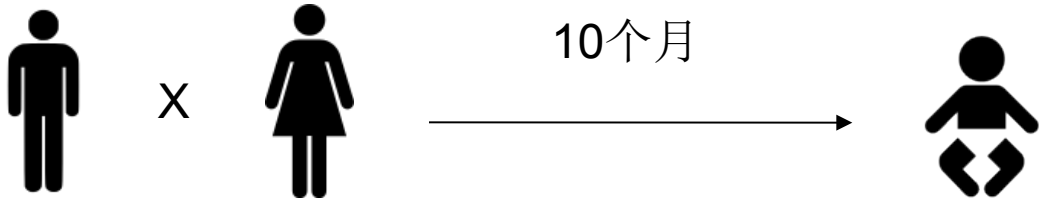
有一天高帅富王尼玛找到我

曹力，我们来搞一票大的吧

目标
日访问量1000万PV！

高帅富的开发方式

我眼中的软件开发

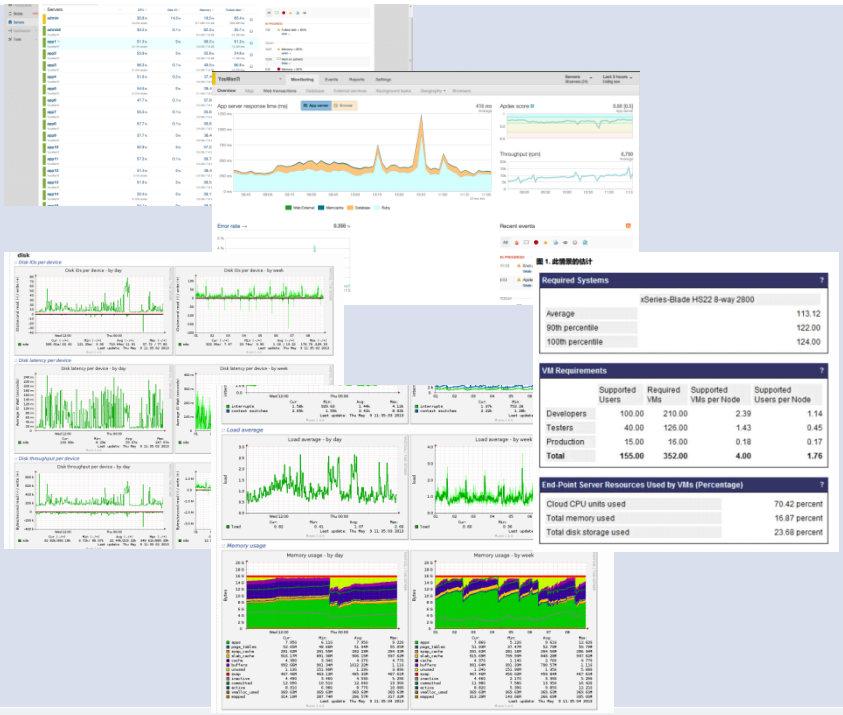


老板眼中的软件开发

系统性能不够？！

多买几台服务器！

我眼中的容量规划



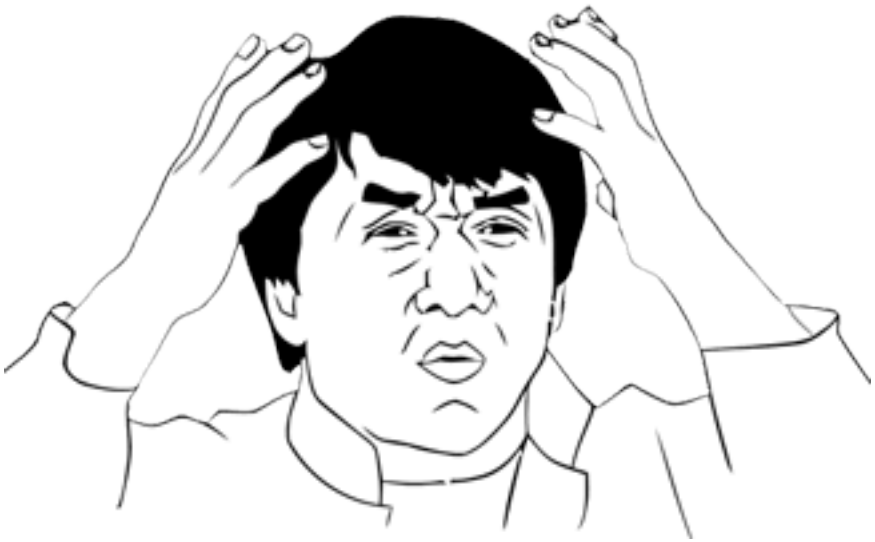
老板眼中的容量规划

1台服务器=100w PV

10台 = 10 * 100w PV = 1000w PV

100台 = 1亿 PV

1亿 PV = $$$$$$$$$....



这不是扯淡么

Thanks to Open Source Software

Thanks to Cloud computing

有了云服务
现在1人顶过去10人
一口气弄10台服务器
腰不酸腿不疼

妈妈再也不用担心我的Scalability

一些准备工作

监控所有服务的状态

服务器的CPU 内存 磁盘IO

MySQL slowlog

各个页面的效率

一些准备工作

优化系统参数sysctl.conf
优化Ruby虚拟机的参数
优化MySQL参数
优化磁盘参数、文件系统

And more ...

Challenge1. 多台Rails服务器与缓存

caches_page的问题

- 只能使用文件系统
- 多机共享则必须使用NFS
- NFS需要进行较多的配置、挂载
- 如果不进行定期清理，文件数量会不断增加
- 如果在文件数量很多，遍历目录进行清理消耗时间太长
- 优势：省内存。

文件系统→Memcache

Memcache的优势

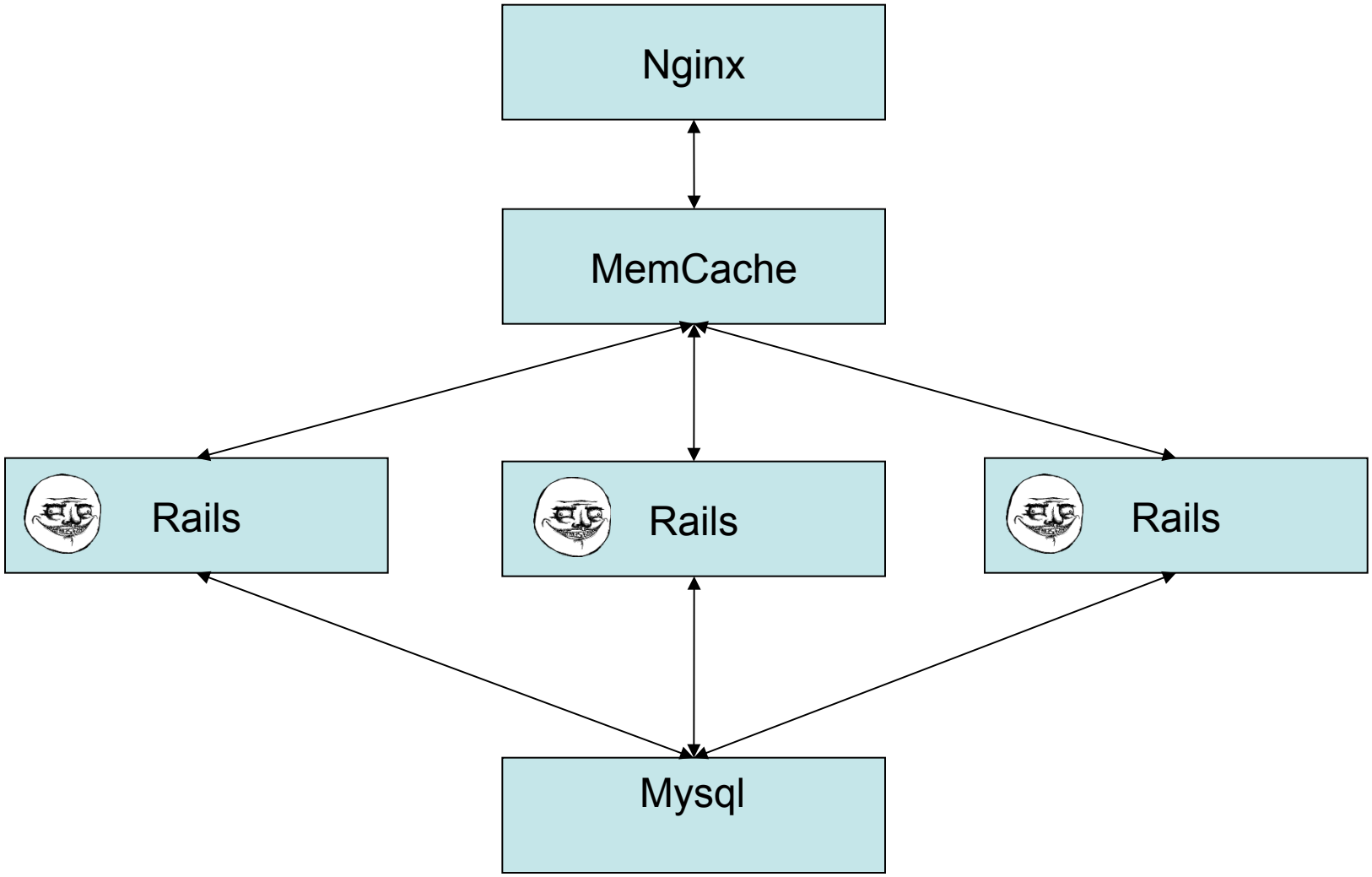
- 配置简单
- nginx内建对memcache后端的支持
- 自动失效

SuperCache

https://github.com/ShiningRay/super_cache

```
super_caches_page :index
```

use `Rails.cache`


Nginx
MemCache
Rails
Rails
Rails
Mysql

Challenge2. 缓存的横向扩展

nginx+多台memcache的问题

- nginx不支持使用一致性哈希来选择memcache后端（可以使用第三方模块）

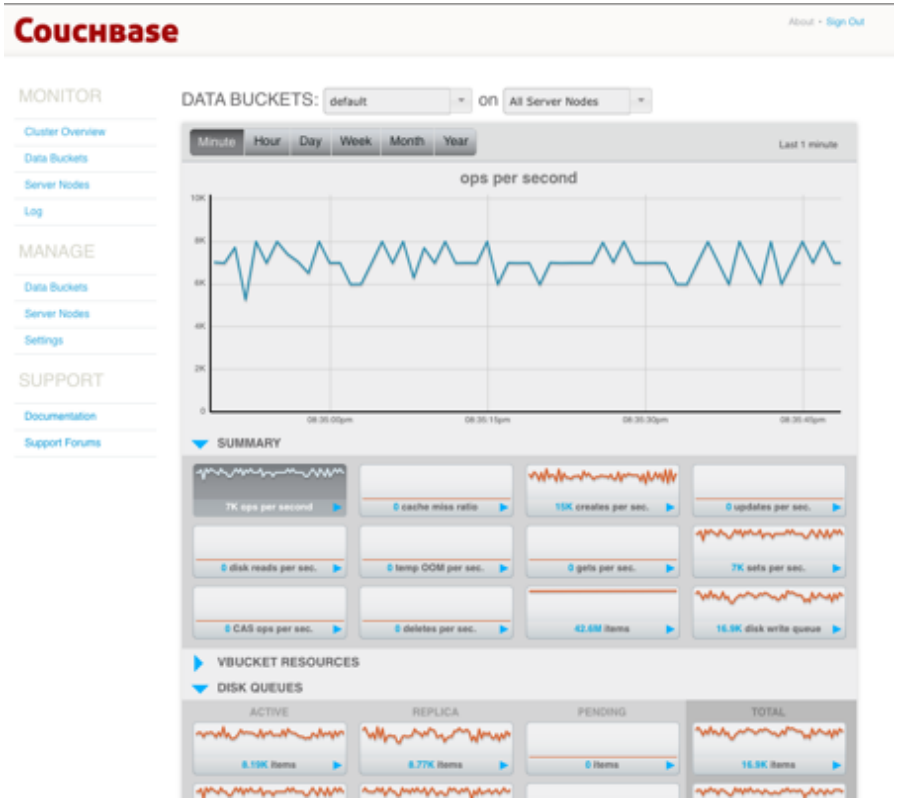
Membase to rescue

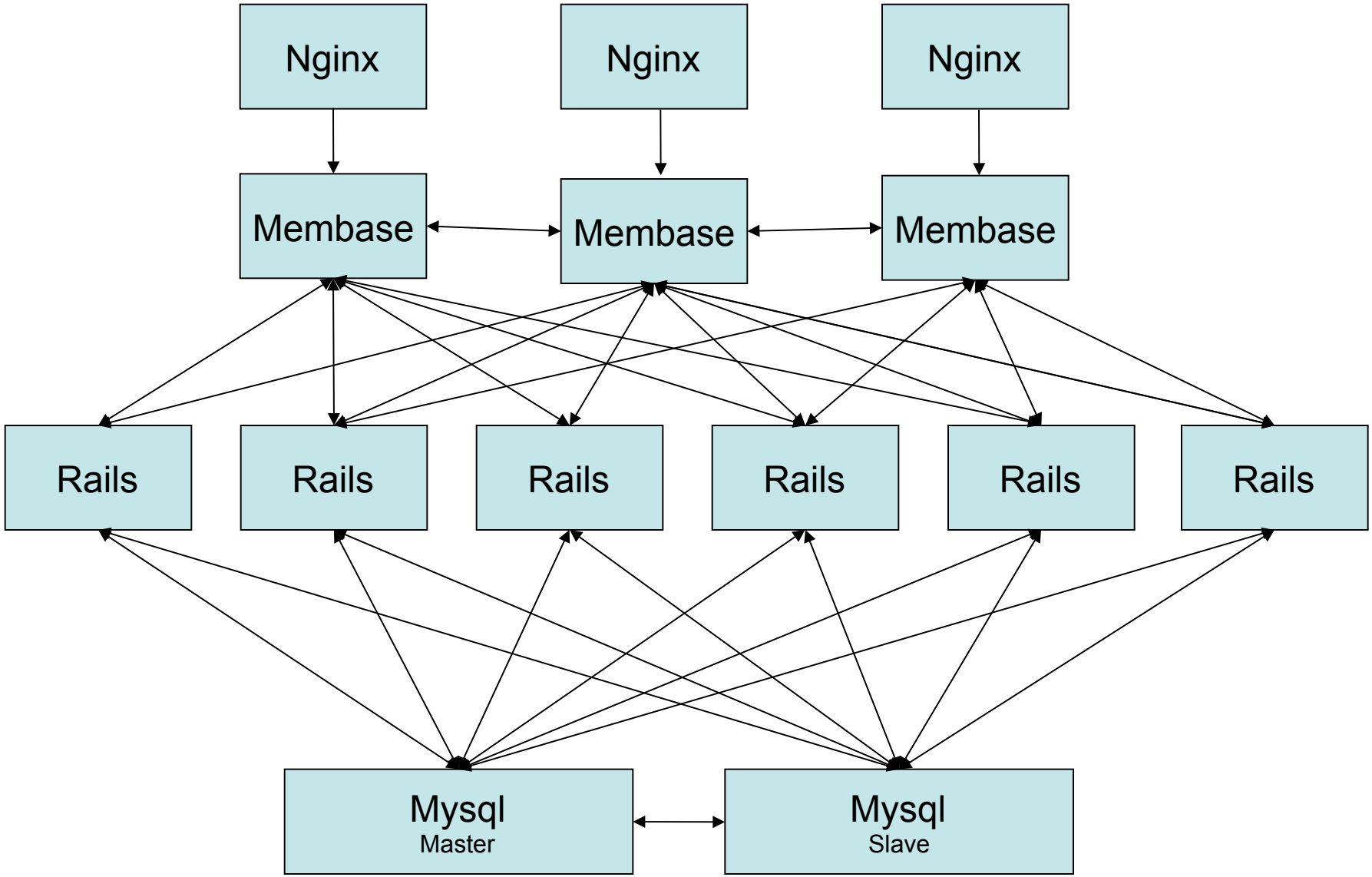
COUCHBASE

Membase特点

- 完全兼容Memcached协议
- 横向扩展性极强
- 任意节点可以读取到全部数据
- GUI操作简便
- 高可用性，自动故障转移

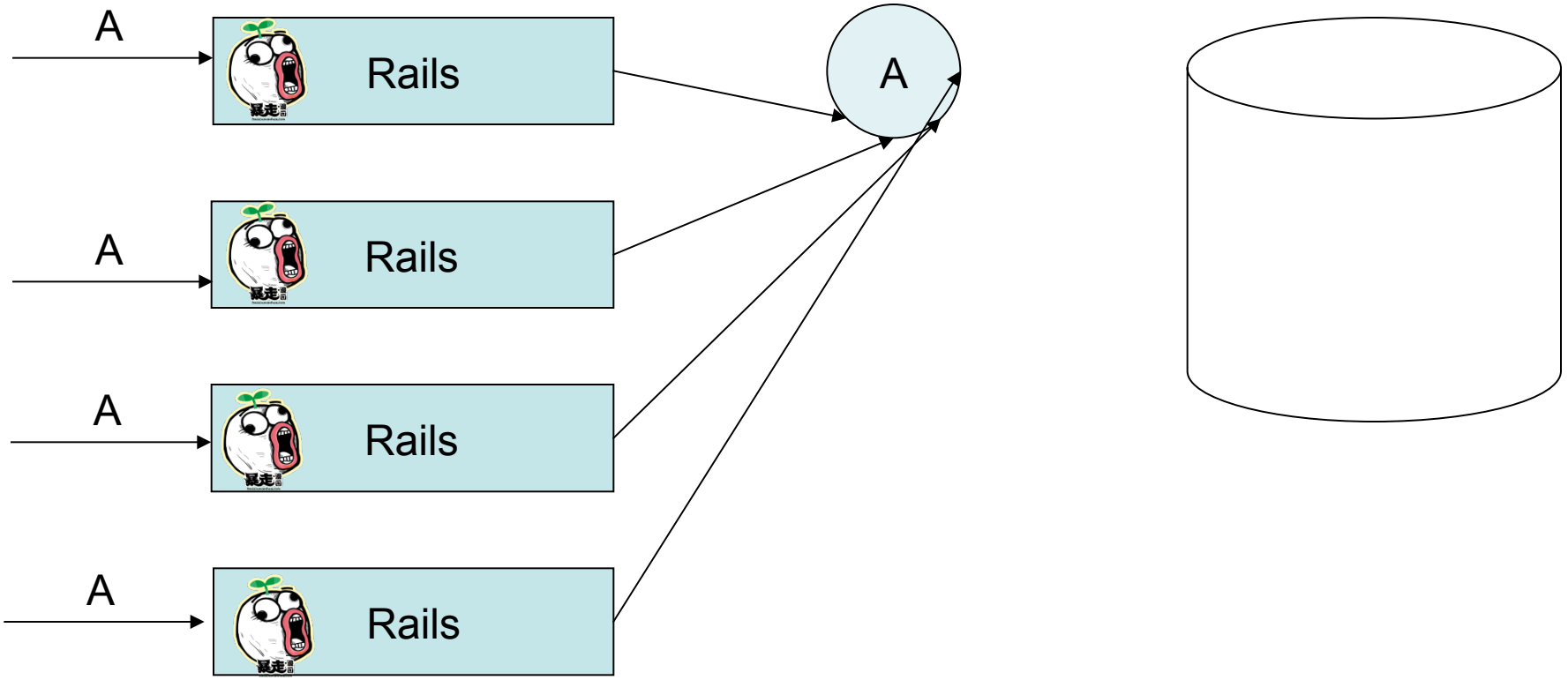
Couchbase/Membase 控制台

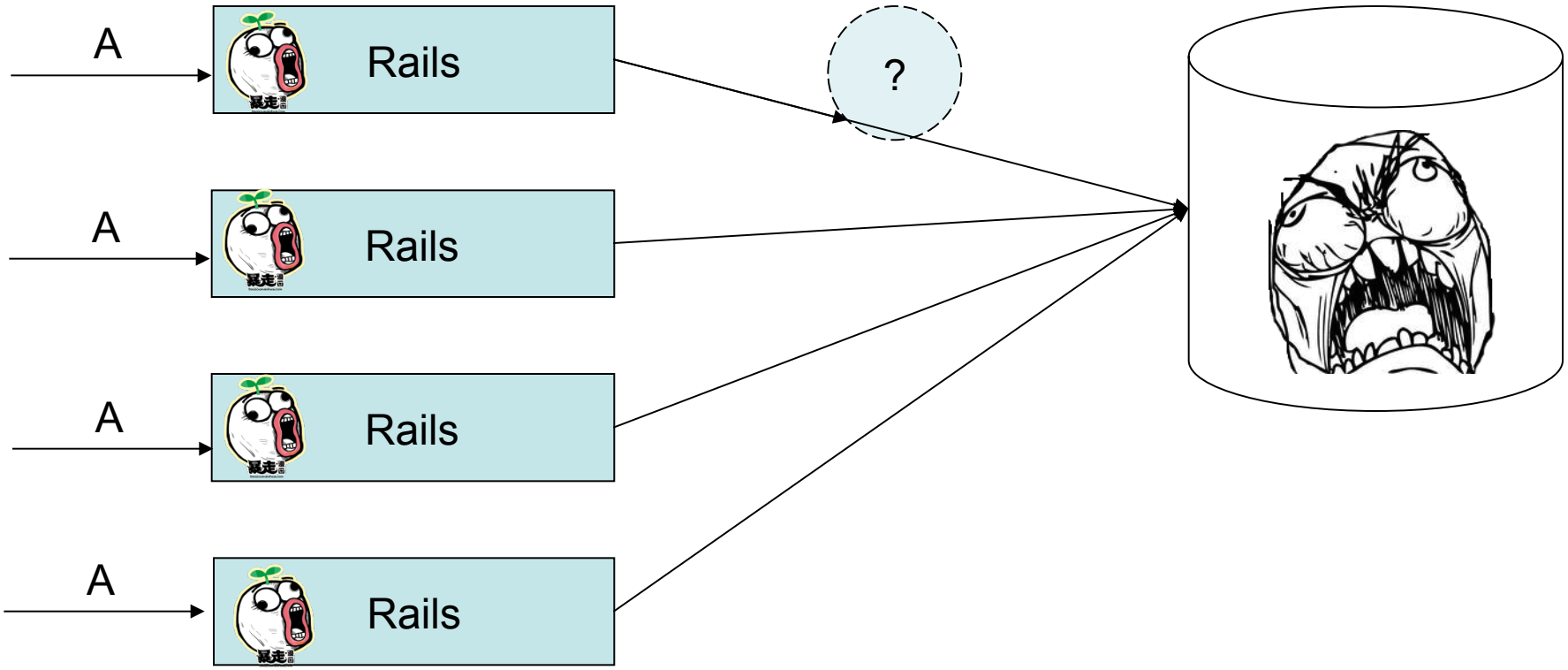



Nginx
Nginx
Nginx
Membase
Membase
Membase
Rails
Rails
Rails
Rails
Rails
Rails
Mysql
Master
Mysql
Slave

Challenge3. Dog Pile Effect

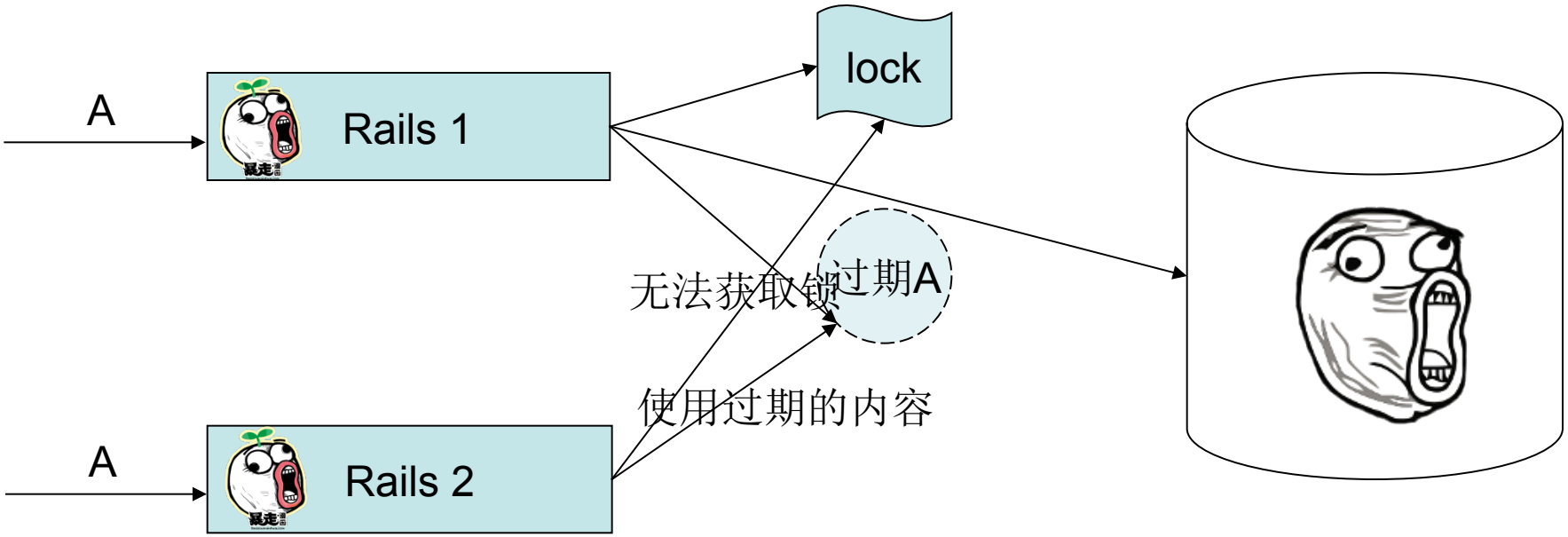
What's Dog Pile Effect


A
Rails
A
Rails
A
Rails
A
Rails
A


A
Rails
A
Rails
A
Rails
A
Rails
?

解决方案

- WriteThrough：不失效缓存，当更新数据后直接更新缓存（适合Model层）
- Lock：当缓存对象过期时，只有获得锁的进程才能更新缓存



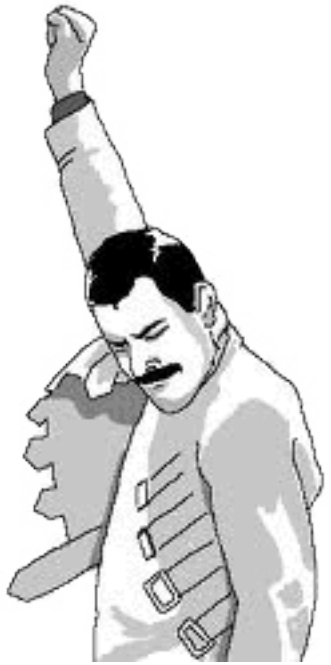
Lock 机制

Distributed Lock实现方式

- Memcache的原子操作
- Redis的原子操作

SuperCache

```
super_caches_page :index, :lock => true
```

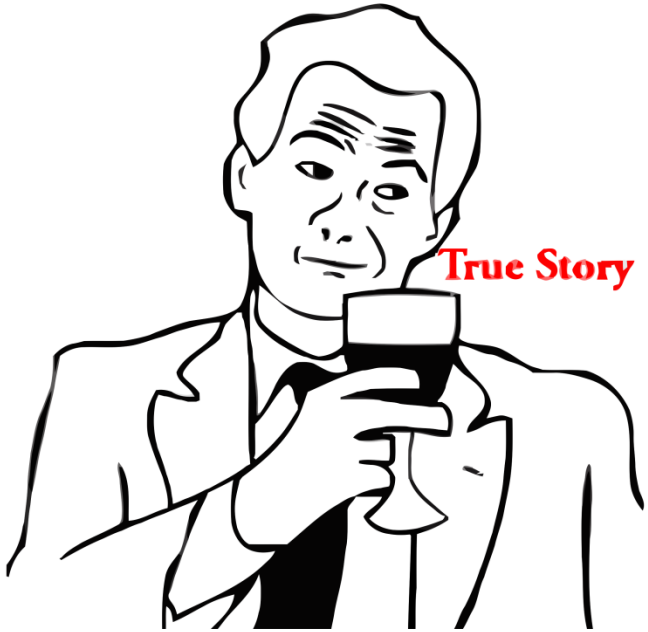


日1000万PV达成！


妥妥的

总结

- 使用80/20原则进行缓存
- 使用云服务减少开发量
- Membase替换Memcache
- 使用锁来防止Dog Pile Effect



谢谢观赏

http://baozoumanhua.com