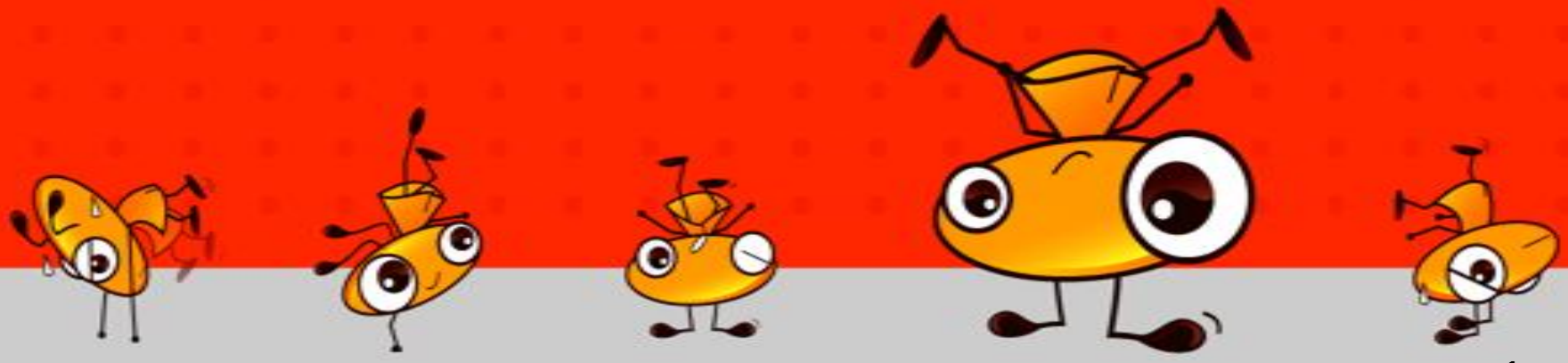
淘宝网
Taobao.com

Alibaba Developer Conference

Detail静态化

（淘宝动态系统的静态化改造实践）

—— 淘宝君山



花名：君山
真名：许令波
博客：http://xulingbo.net
微博：http://weibo.com/u/1855869382
简介：2009加入淘宝，一直在做Detail系统的性能优化方面的工作，研究过Cassandra、开发过一个sketch模板引擎、给developerworks 投稿,出版《深入分析Java Web技术内幕》。
部门：交交易中心-导购-商品详情
所在团队：小凡、济城、大仁、常彬、旭天、君山

Detail系统（http://item.taobao.com）

Detail系统是个什么样的系统—业务上

- ✓ 占有整个淘宝90%的PV
- ✓ 超过70%的成交是从Detail页发起的
- ✓ 淘宝最重要的系统之一、是买家达成购买意向的关键路径
- ✓ 唯一详细商品展示信息的地方
- ✓ 卖家可以随意装修和编辑商品的描述

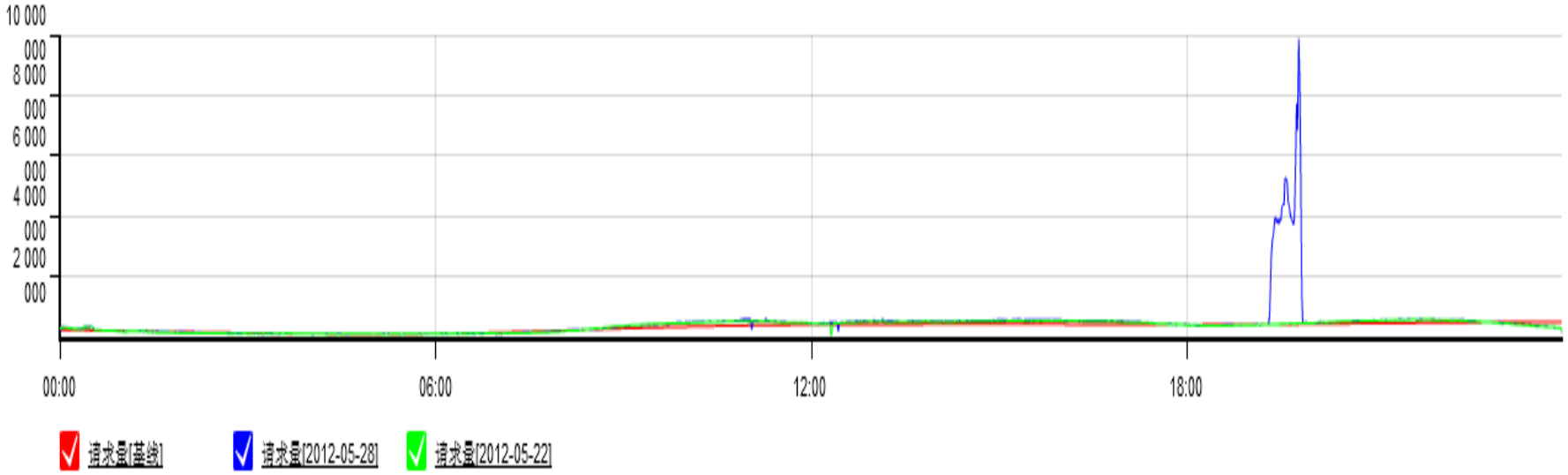
Detail系统性能数据

- ✓ 淘宝访问量最高的系统
  - PV：5亿/每秒1.8w
  - UV：3100万（其中直接登录用户1700万）
- ✓ 淘宝单个应用占用最多机器的系统
  - 300台vm
- ✓ 动态系统中性能最好的系统
  - 最大350QPS
  - RT30ms
- ✓ 对外峰值带宽：4.03Gbps
  - 平均页面压缩前108k、压缩后28k

面临的挑战

面临的挑战

- 网站经常受到攻击
- 双11/双12的大型促销活动
- 秒杀/活动等突发流量冲击
- 各种爬虫频繁抓取数



Detail系统的优化历程

- 前端优化
  - ✓ assets合并
  - ✓ 整合页面中inline的js\css
  - ✓ combo合并css、js文件
  - ✓ BigRender（使用textarea，控制浏览器渲染节奏）
- 服务端优化
  - ✓ 将iframe改为jsonp调用
  - ✓ 建立异步系统（JS触发加载）
  - ✓ Velocity模板优化（sketch框架）
  - ✓ 逐步去除DB依赖（各个C走Tair缓存）
- 网络优化
  - ✓ TCP初始拥塞窗口优化
  - ✓ 去除空格、TAB压缩字符串，减少页面大小
- 终极优化（静态化）

主要内容

淘宝网 Taobao.com

- 什么是静态化
- 为什么要静态化
- 系统改造方案选择
- 动态系统如何改造

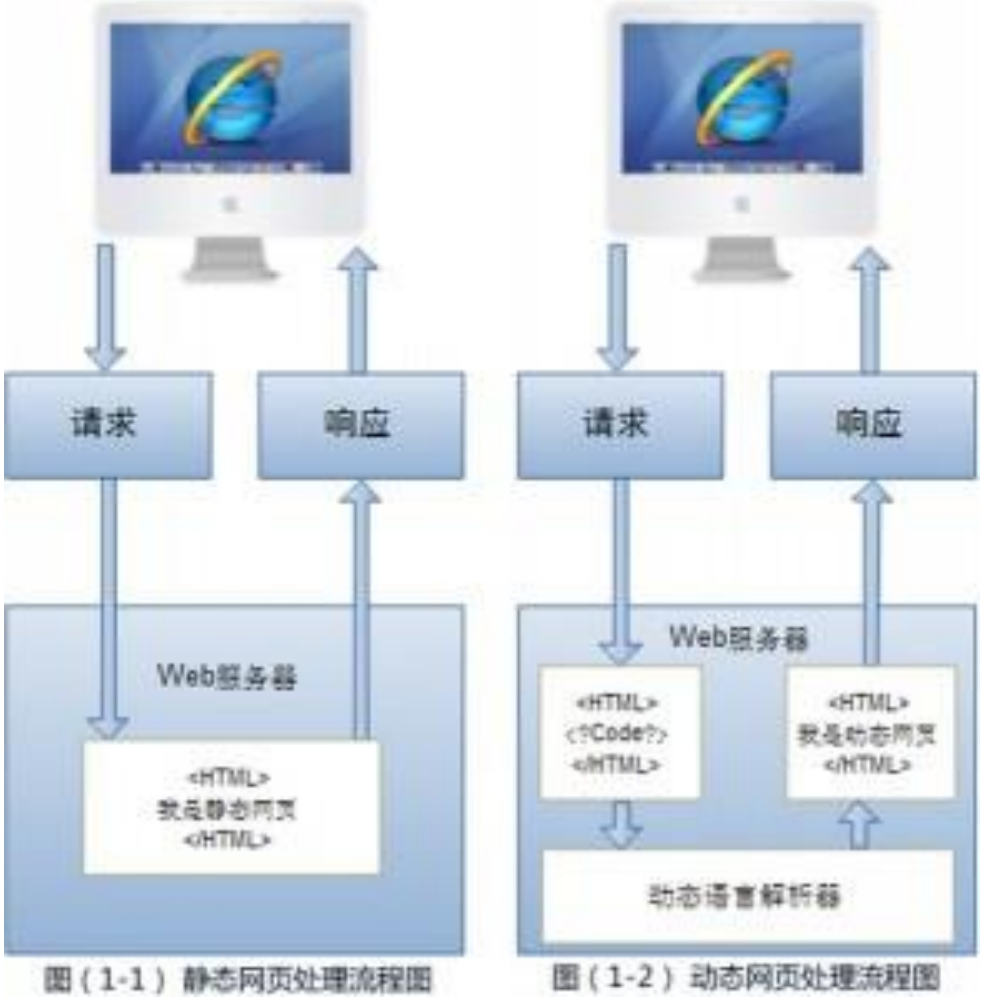
  要让页面与浏览者无关、与时间无关、与地域无关、页面内容如何动静分离？
- 页面如何缓存

  页面怎么缓存？用什么软件缓存？是在HTTP层做缓存还是在应用层做缓存？
- 如何失效

  是整体失效还是局部失效？主动失效or被动失效？失效策略如何选择，如何保持数据一致性

什么是静态化

- 一个页面的URL固定，不同的URL表示不同的内容，让返回的请求之和URL相关
- 去掉页面中浏览者因素
- 去掉页面中的时间因素
- 去掉页面中地域因素
- 去除Cookie等私有数据



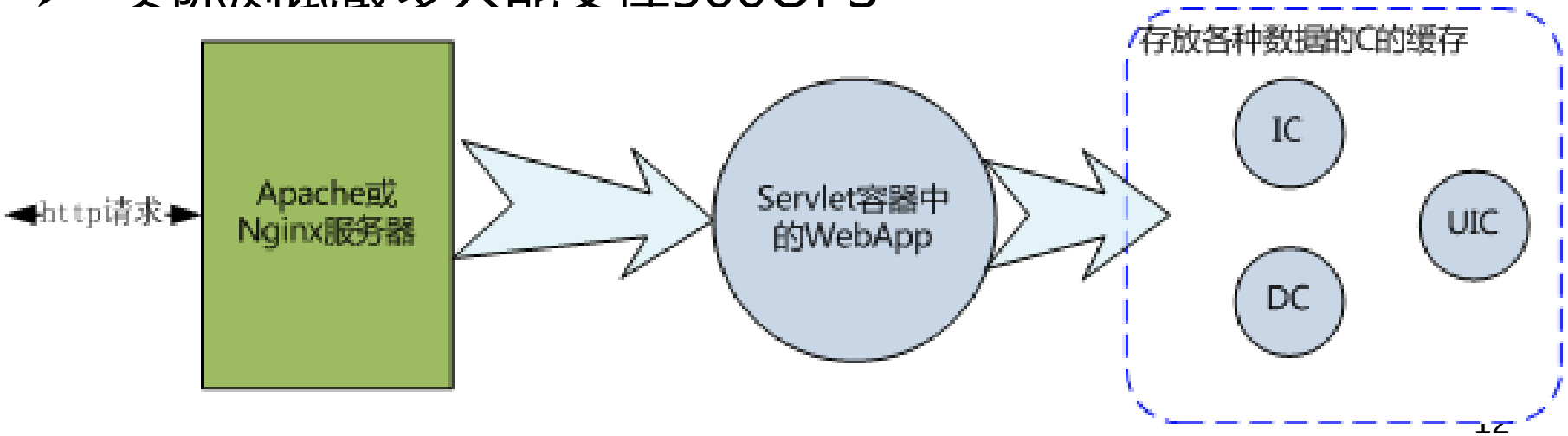
图（1-1）静态网页处理流程图　　图（1-2）动态网页处理流程图

为什么要静态化

- 动态系统的QPS有瓶颈
  1. Java
     - CPU运算（Java字符串查找、替换、拼接、锁、压缩、解压缩等）
     - 元数据获取的网络开销
  2. Web服务器
     - 模块过滤（atpanel pv、acookie打点、简繁转换）
     - 大HTML(100K以上)的Gzip压缩
- 淘宝的突发流量
  1. 攻击
  2. 秒杀
  3. 双11/双12（21w/QPS）
- 性能已经不能再得到大的提升了

当前的Detail架构面临的技术挑战

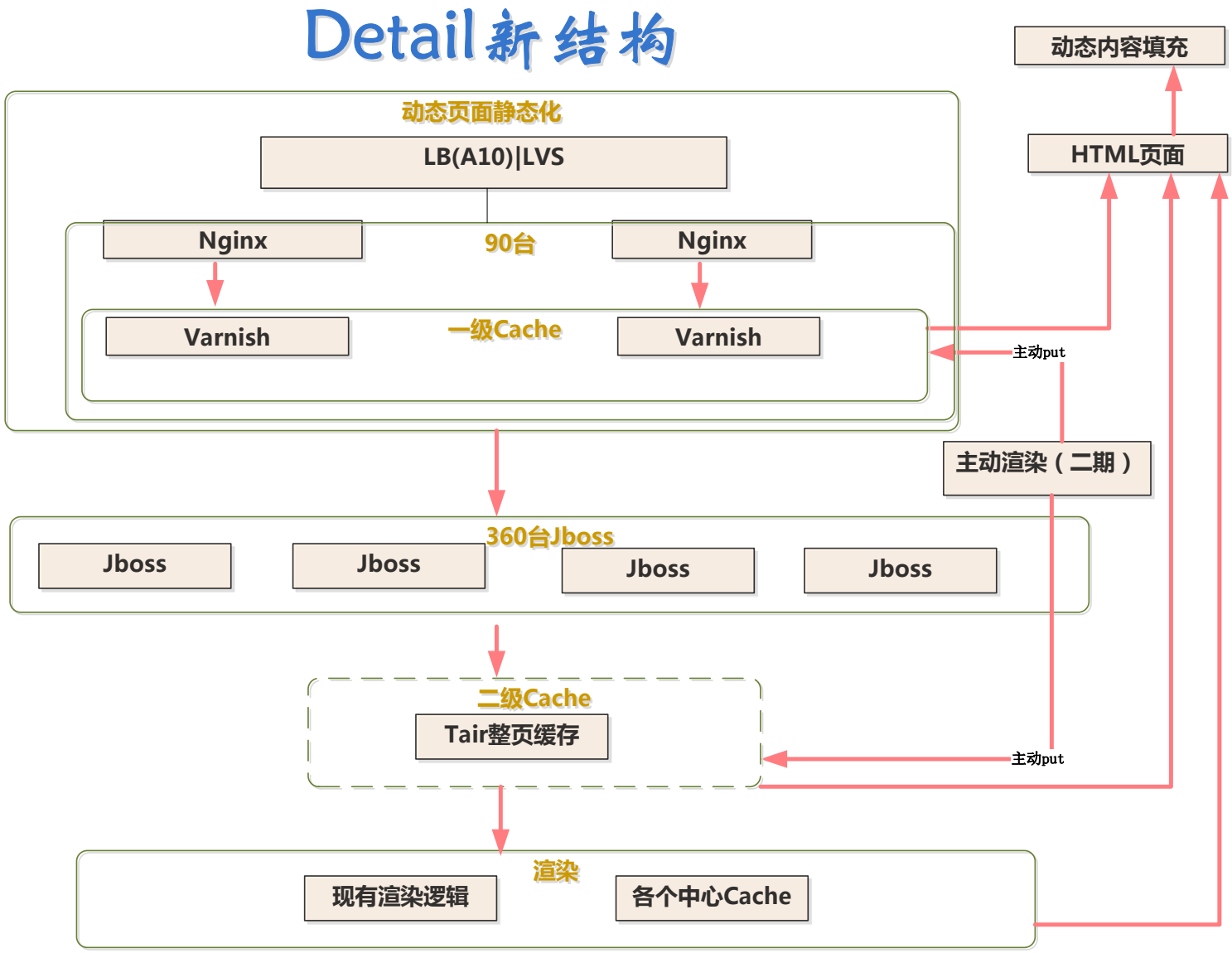
- 当前的Detail架构已经达到瓶颈
  - ✓ 能使用缓存的地方已经使用了
  - ✓ 服务端的CPU瓶颈能优化的基本已经优化（模板、压缩）
- Java系统的性能瓶颈
  - ✓ 不擅长处理大量连接（NIO事件驱动较弱）
  - ✓ 网络数据传输较差（数据需要在内核和用户态直接切换）
  - ✓ Servlet容器也有性能瓶颈
- 实际测试最多只能支撑500OPS



Detail必须要有新的架构

- 改变缓存方式
  - ✓ 直接缓存Http
- 改变缓存的地方
  - ✓ 直接基于Web服务器换
  - ✓ 屏蔽业务逻辑（值根据URL）
- 基本原则
  - ✓ 保证易维护性
  - ✓ 没有单点
  - ✓ 缓存足够大

Detail新架构方案选择

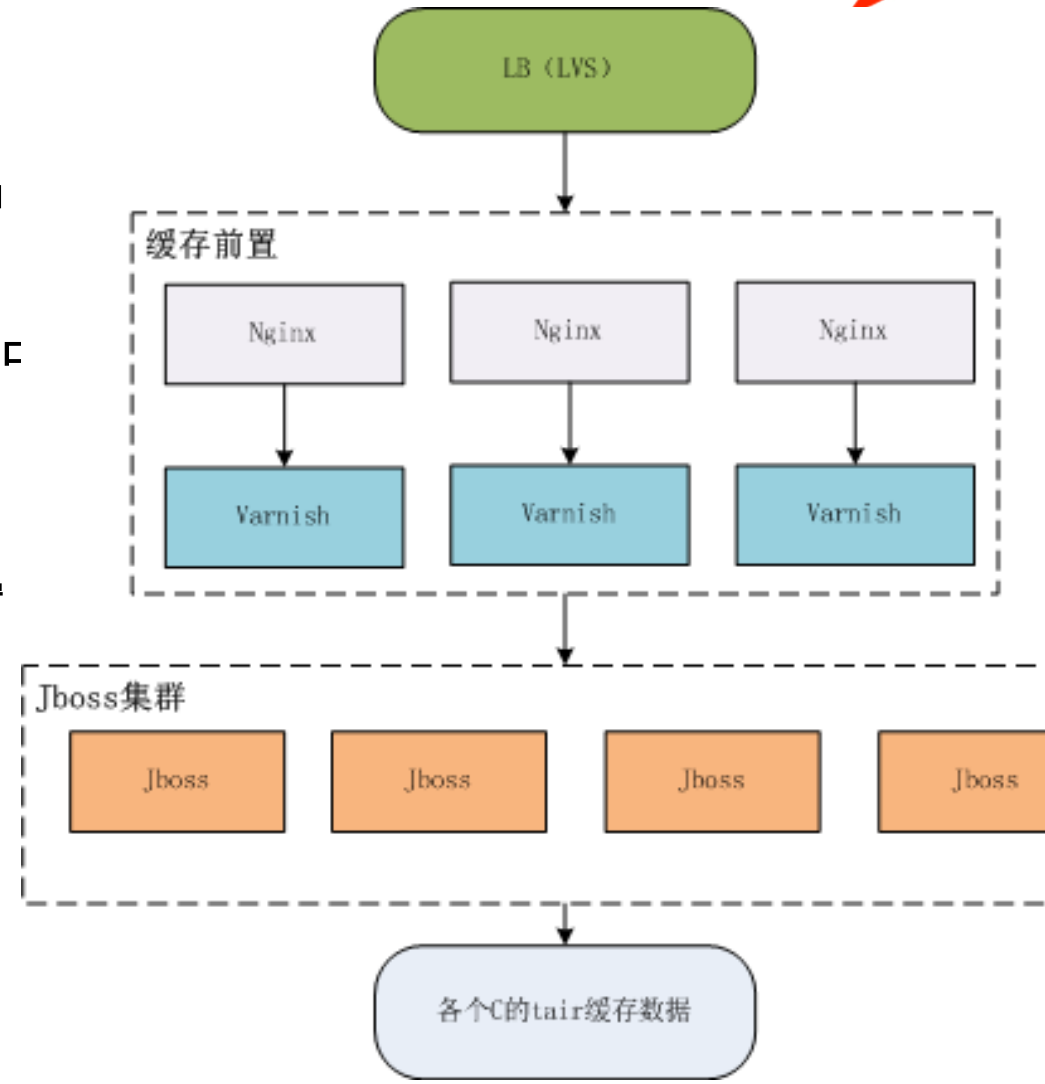


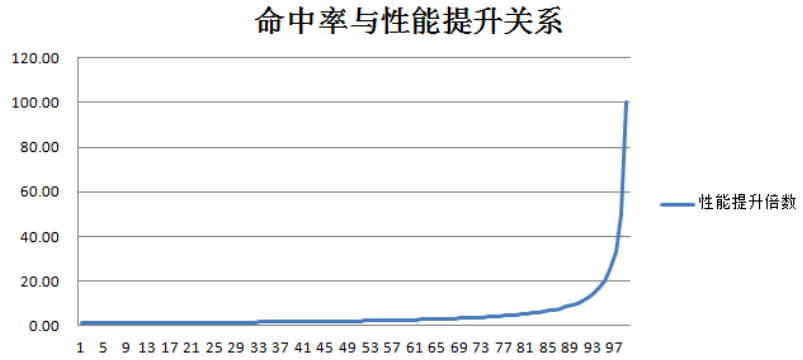
方案选择的关键因素

- 是否一致性Hash分组？
- 是否使用ESI？
- 是否使用物理机？
- 谁来压缩?
- 网卡选择

改造方案一 Nginx+Varnish前置（20台）

- 优点
  - ✓ 与Java应用隔离，方便PE维护
  - ✓ 大内存，缓存集中管理
  - ✓ 命令率高（性能提升=1/(1-命
- 缺点
  - ✓ 内部交换网络成为瓶颈
  - ✓ CPU也会成为瓶颈（Gzip压缩
  - ✓ 缓存服务器的网卡也会是瓶颈
  - ✓ 机器少风险较大

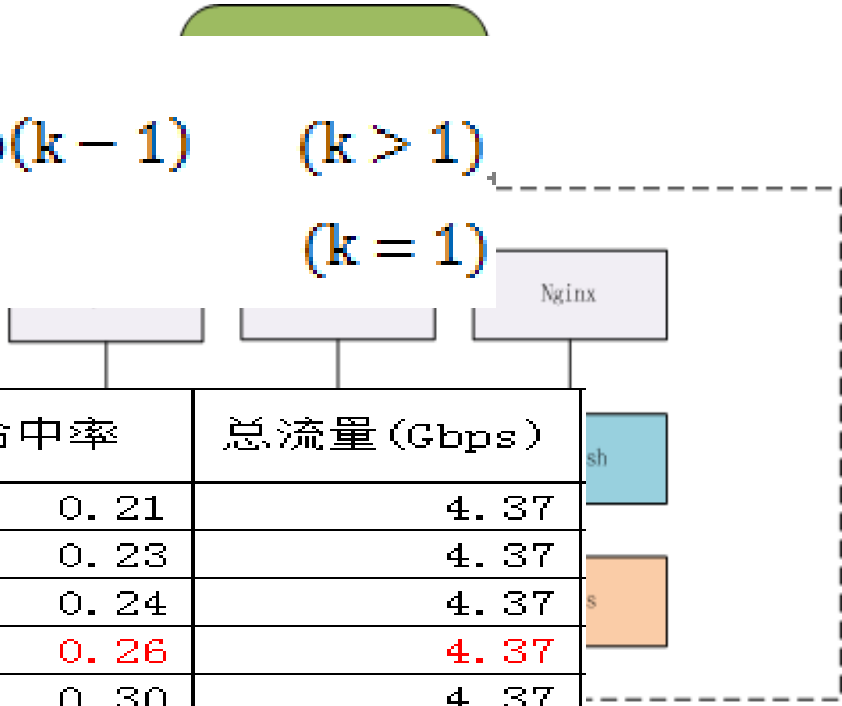




改造方案二 Nginx+Varnish+Jboss（vm）

- 优点
  - ✓ 没有网络瓶颈
  - ✓ 机器增加，t
  - ✓ 机器数增多故
- 缺点
  - ✓ 机器增加，
  - ✓ 缓存分散，
  - ✓ Varnish和J

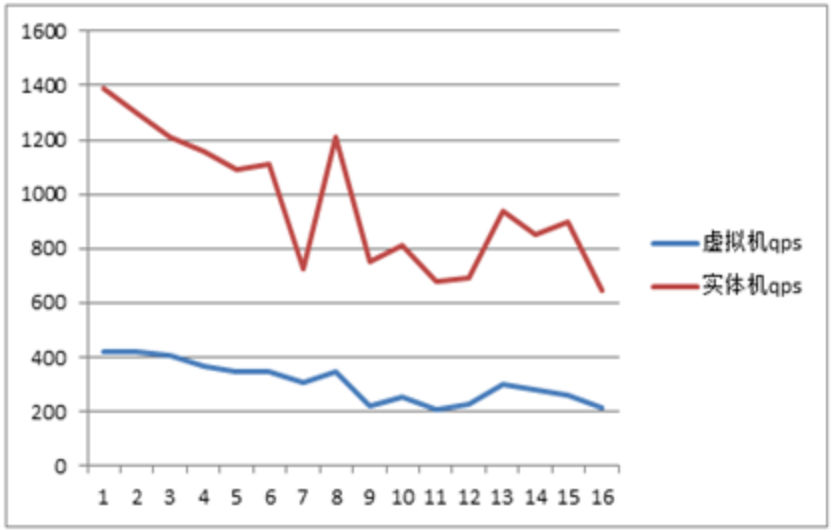
$$\begin{cases} p(k) = \frac{1}{n} + \frac{(n-1)}{n} * p(k-1) & (k > 1) \\ p(1) = 0 & (k = 1) \end{cases}$$



集群QPS	机器数	命中率	总流量(Gbps)
18200	120	0.21	4.37
18200	90	0.23	4.37
18200	80	0.24	4.37
18200	60	0.26	4.37
18200	40	0.30	4.37
18200	30	0.33	4.37
18200	16	0.41	4.37
18200	10	0.47	4.37
18200	6	0.55	4.37
18200	4	0.60	4.37
18200	2	0.69	4.37
18200	1	0.76	4.37

改造方案三 Nginx+Varnish+Jboss(实体)

- 实体机和虚拟机对比
  - ✓ 实体机是虚拟机的3倍
- 中庸之美
  - ✓ 既没有网络瓶颈也能使用大内存
  - ✓ 减少Varnish机器，提升命中率
  - ✓ 提升命中率，能减少Gzip压缩
  - ✓ 减少CC失效压力



- 临时方案（双12）
  - ✓ 让Varnish压缩，解除CPU瓶颈（采用Java实现Beacon打点）
  - ✓ 采用CSI渲染动态数据（目前采用Varnish的ESI填充动态数据）

系统的改造点（动态系统静态化）

- URL是否单一（get参数）
- 去点post表单
- Cookie不影响业务逻辑

图（1-1）静态网页处理流程图

图（1-2）动态网页处理流程图

系统的改造点（动态内容异步化）

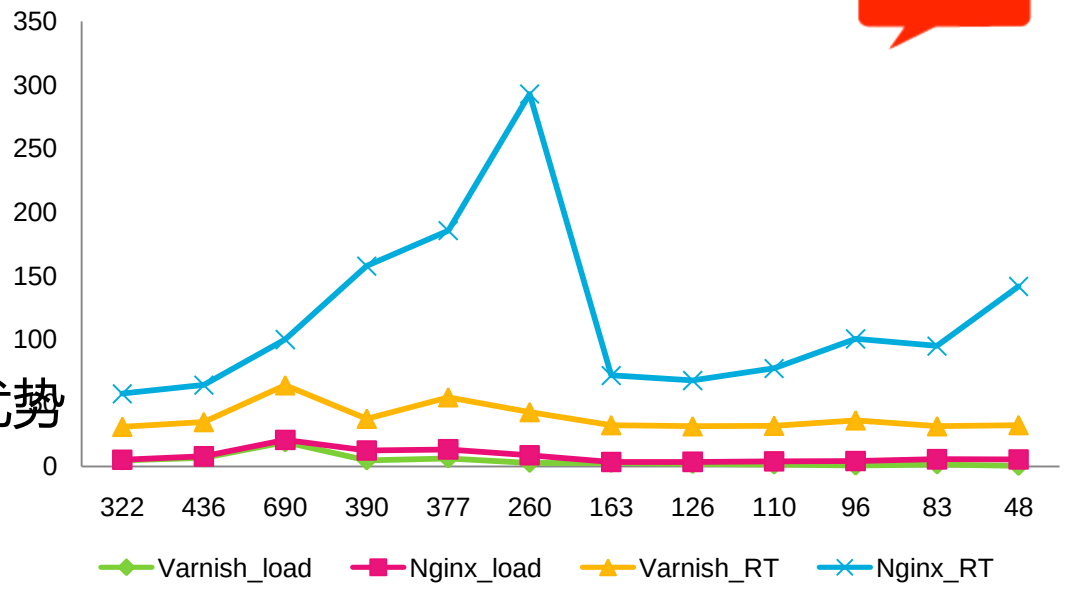
- URL固定（/item.htm?id=xxxx）
- 去掉浏览者相关
  - ✓ Atpanel打点
  - ✓ 用户id等Cookie信息
- 去掉时间相关
  - ✓ 定时上架
  - ✓ 秒杀
- 去掉地域相关
  - ✓ 运费模板

系统的改造点（动态内容格式化）

- ViewData（浏览者相关的数据）
- ItemData（商品相关的数据）

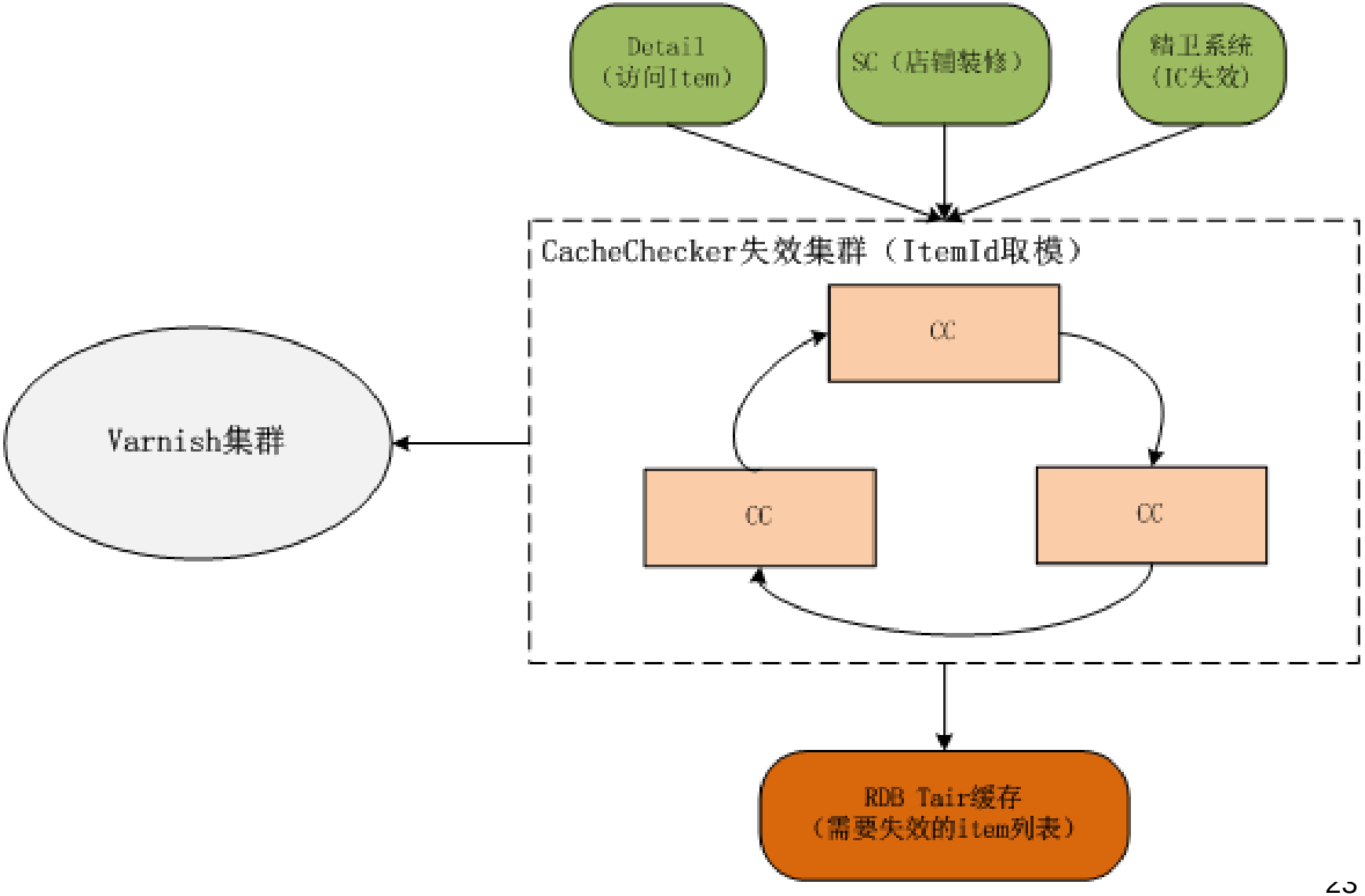
```
window.TB.Detail.common.data.idata={
  item:{
    id:1,//item Id
    status:0
  },
  seller:{
    id:1,//sellerId
    status:0
  },
  shop:{
    id:1,//id>0 has shop
    eshop:true//是否是旺铺
  },
  //This Field need To be Injected by Ajax
  viewer:{
    id:1,
    buySum:2//买家信用
  }
}
```

缓存选择Varnish



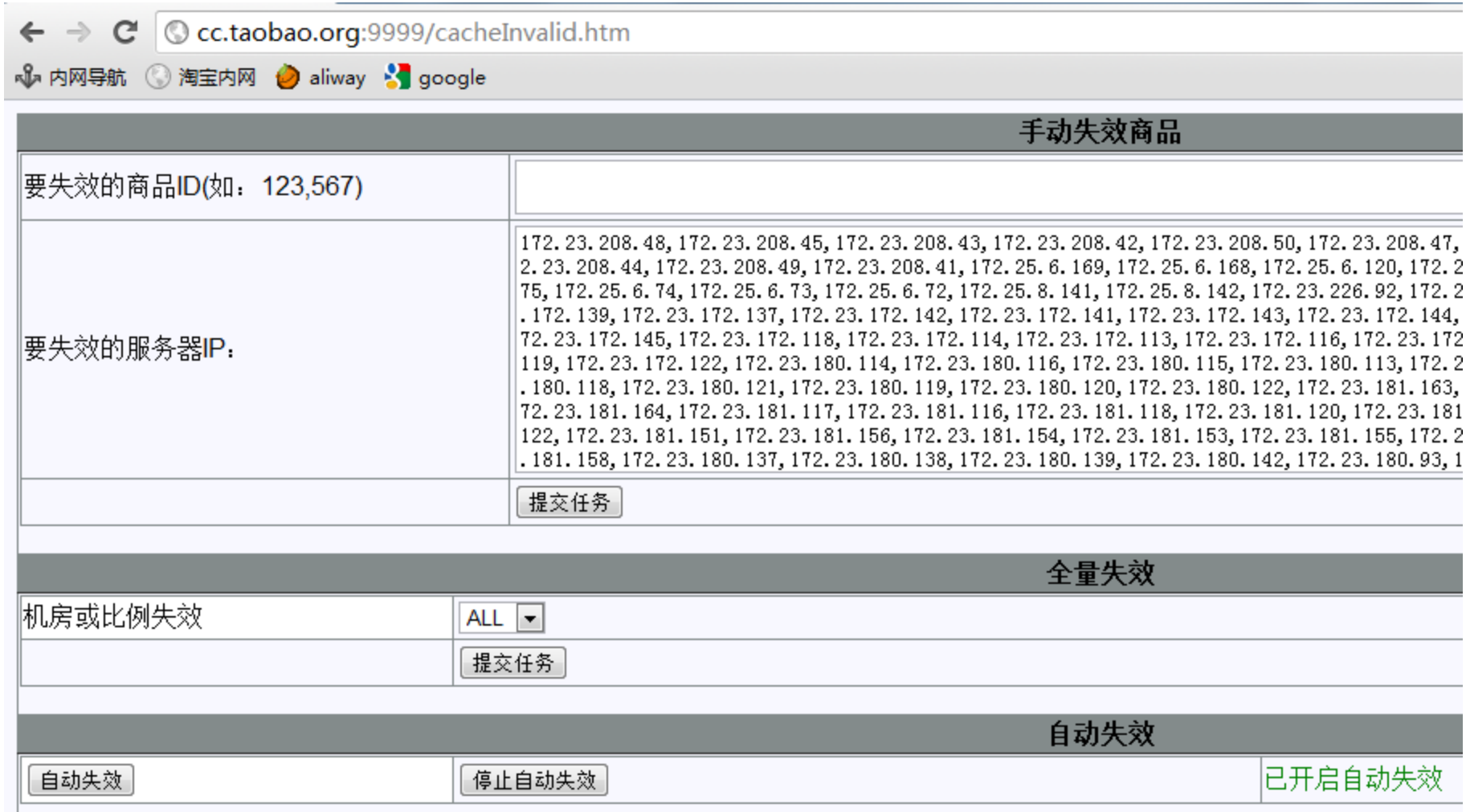
- 为什么没选择Nginx
  - ✓ Nginx使用硬盘缓存
  - ✓ RT和Load波动
  - ✓ 不稳定，不能发挥缓存优势
- 优点
  - ✓ 与其他缓存软件相比，直接在Http层做缓存，不需要额外的协议解析，和复杂的逻辑
  - ✓ 与其他Http层软件相比，Varnish性能更好，使用方便、维护简单
  - ✓ 支持ESI
  - ✓ 有大型网站都在使用（Facebook 静态资源）
- 缺点
  - ✓ 淘宝没有运维经验

失效方案——结构


Detail
（访问Item）
SC（店铺装修）
精卫系统
（IC失效）
CacheChecker失效集群（ItemId取模）
CC
CC
CC
Varnish集群
RDB Tair缓存
（需要失效的item列表）

失效方案——方式

- 主动失效
  - ✓ Cc监控



失效方案——技术难点

- 每秒要失效的数据量大
  - ✓ IC每秒4000，SC没有50
  - ✓ 延时、去重、批量失效
- 要失效的Varnish多
  - ✓ 当前有260台，那么每秒要发送104w/s的请求（20台CC）
  - ✓ 采用基于Keepalive的长连接（单台6w/s的http请求）
- CC性能调优
  - ✓ Mina内存泄露
  - ✓ 多线程操作load不稳定
- Varnish调优
  - ✓ 修改Error的长连接机制
  - ✓ 大内存写磁盘问题
  - ✓ Stat-out问题

失效方案——失效Varnish缓存

- Purge
  - ✓ 走Http协议，性能非常好
  - ✓ 不能做正则匹配，失效方式单一
  - ✓ 改造了Varnish的keeplive机制

```
String line = "PURGE /item.htm?id=" + itemid + " HTTP/1.1\nHost:\n\n\n";
```

- Ban
  - ✓ 可以使用正则表达时匹配缓存的key
  - ✓ 性能差

```
"ban.url /*$\n";
```

```
if (req.request == "PURGE") {

    if (client.ip ~ local) {
        return (lookup);
    } else {
        error 405 "Not allowed.";
    }
}
```

```
[junshan@detail188029.cm4 ~]$ curl  -X PURGE localhost:8888/item.htm?id=12643598611 -I
HTTP/1.1 200 Purged.
keeplive: true
```

Varnish机器数与命中率的关系

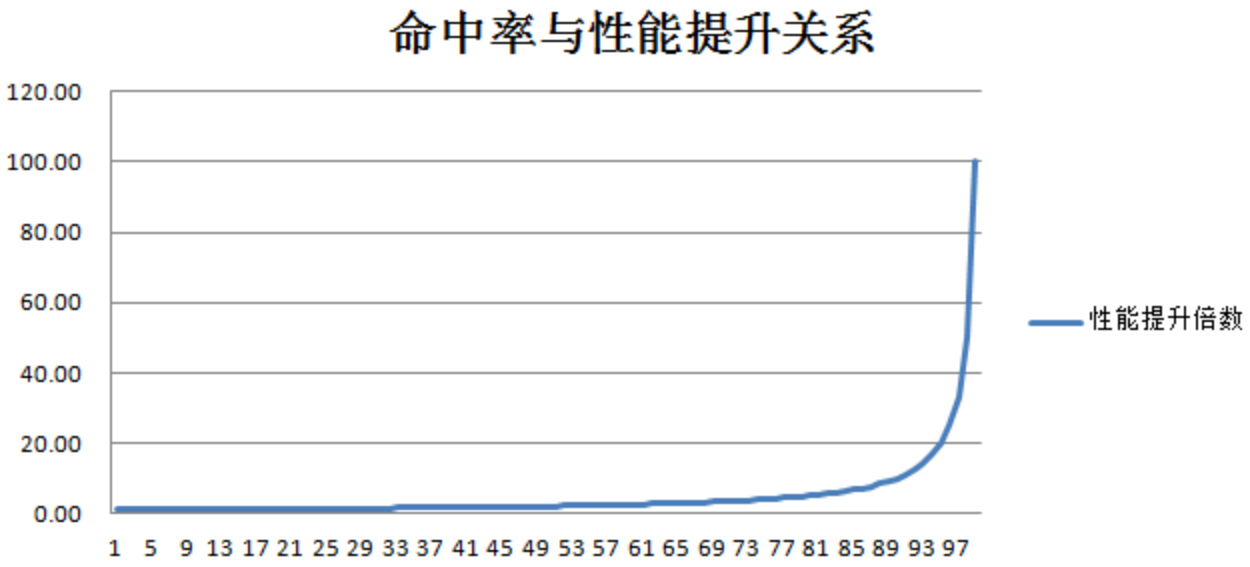
- 提供缓存的机器数和每台机器的命中率有直接关系
- 机器数与命中率关系

$$\begin{cases} p(k) = \frac{1}{n} + \frac{(n-1)}{n} * p(k-1) & (k > 1) \\ p(1) = 0 & (k = 1) \end{cases}$$

集群 QPS	机器数	命中率	总流量(Gbps)
18200	120	0.21	4.37
18200	90	0.23	4.37
18200	80	0.24	4.37
18200	60	0.26	4.37
18200	40	0.30	4.37
18200	30	0.33	4.37
18200	16	0.41	4.37
18200	10	0.47	4.37
18200	6	0.55	4.37
18200	4	0.60	4.37
18200	2	0.69	4.37
18200	1	0.76	4.37

命中率与性能提升的关系

- 性能提升=1/(1-命中率)
- 命中率达到50%，则当前的性能为之前的2倍
- 命中率达到90%，当前性能为之前的10倍
- 命中率达到99%，性能为之前的100倍
- 缓存的最大性能受制约于缓存之后的性能



谁做压缩

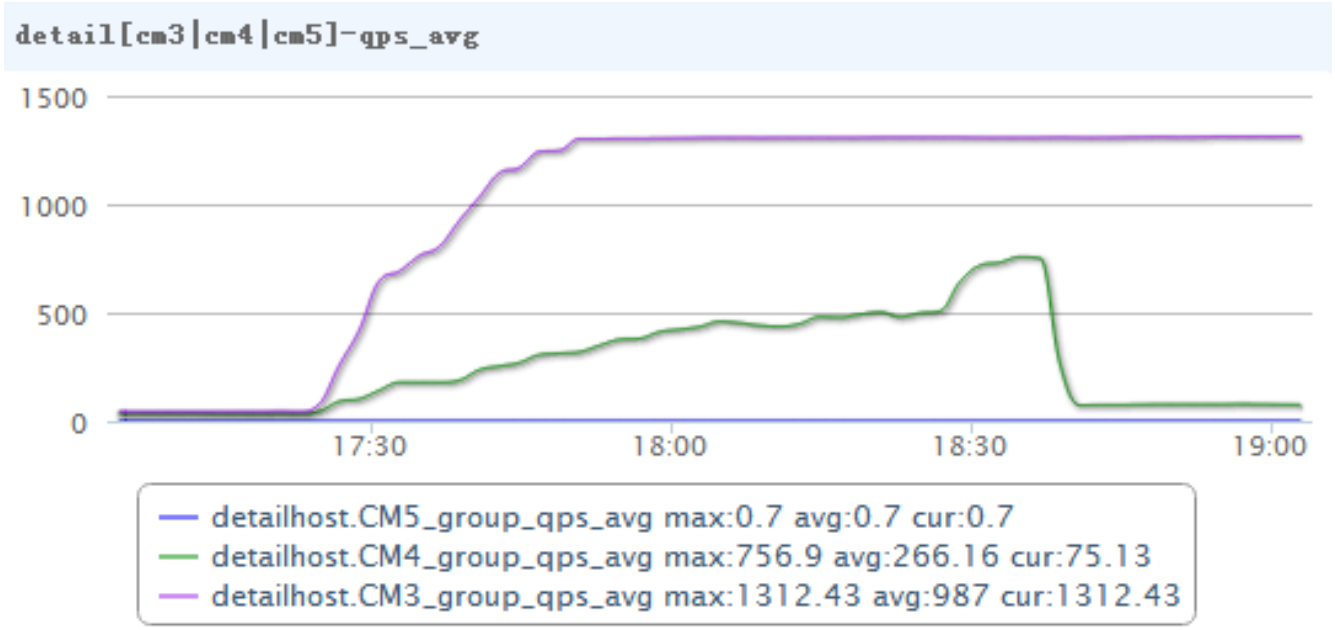
- Jboss做压缩
  - 内部流量可以减少
  - Java压缩不能充分发挥多CPU的优势，性能较差
  - 加长RT，影响正常用户请求
  - 前端Nginx不能做动态插入
- Varnish做压缩
  - Varnish到Jboss会存在内部流量
  - Varnish做ESI解析时需要解压
  - 前端Nginx不能做动态插入
- Nginx/Apache做压缩
  - 增加内部网络流量
  - 可以做动态插入

静态化成果

- 一次实际攻击

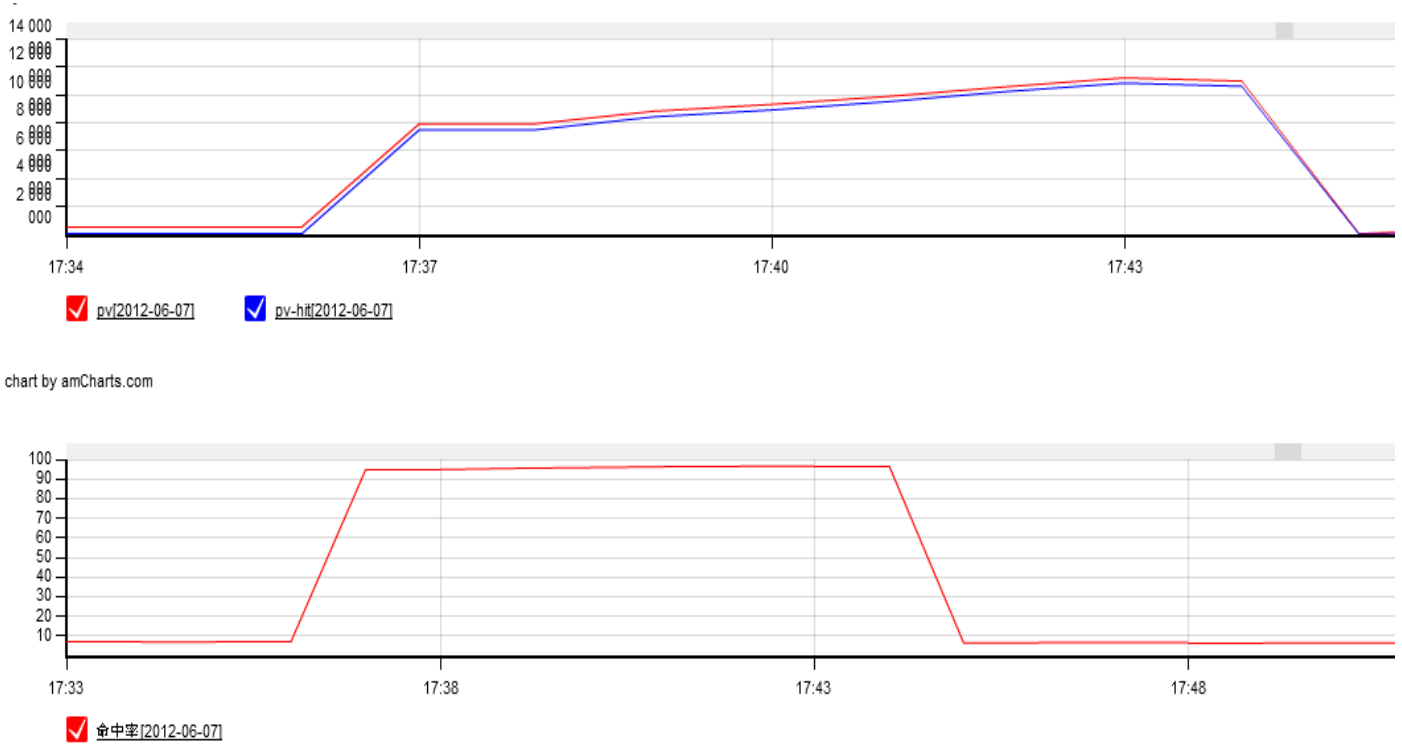
```
"GET http://voli.taobao.com//" 302 260 "http://www.baidu.com/"
"Baiduspider+(+http://www.baidu.com/search/spider.htm)" 220.178.34.190
```

- QPS是平时的20倍（20w/s）



静态化成果

- Varnish命中率非常高
- 响应时间下降
- Load正常



静态化成果

- 这次攻击如果不是静态化，Detail肯定挂
- TMD生效要一个小时
- PE手工操作至少要10分钟
- 如果按照10分钟计算，淘宝要损失多少money
- 而这次攻击，我们可以淡定的去吃晚饭（ ：

感谢所有项目组成员

开发：小凡、大仁、常彬、君山、济城
前端：渐飞、释然
功能测试：红泪、雪倾、纯纯、鸾伽、周叶林
性能测试：静枫、悟石
Varnish&Nginx：文景、叔度
精卫：齐昊、朔海、誓嘉、银时
Tair：仇天、宗岱
PE：普智、胜衣、空智
scm：红巾
SC：铁威、单通
IC：杜琨、逐流

淘宝网
Taobao.com

THANK YOU