Tumblr：150亿月浏览量背后的架构挑战（上）

2012-02-14 18:24 | 3812次阅读 | 【已有8条评论】发表评论
来源：High Scalability | 作者：Todd Hoff | 收藏到我的网摘

导读：和许多新兴的网站一样，著名的轻博客服务Tumblr在急速发展中面临了系统架构的瓶颈。每天5亿次浏览量，峰值每秒4万次请求，每天3TB新的数据存储，超过1000台服务器，这样的情况下如何保证老系统平稳运行，平稳过渡到新的系统，Tumblr正面临巨大的挑战。近日，HighScalability网站的Todd Hoff采访了该公司的分布式系统工程师Blake Matheny，撰文系统介绍了网站的架构，内容很有价值。我们也非常希望国内的公司和团队多做类似分享，贡献于社区的同时，更能提升自身的江湖地位，对招聘、业务发展都好处多多。欢迎通过@CSDN云计算的微博向我们投稿。

以下为译文的第一部分。第二部分点这里。（括号内小号字为CSDN编辑所注）：

Tumblr每月页面浏览量超过150亿次，已经成为火爆的博客社区。用户也许喜欢它的简约、美丽，对用户体验的强烈关注，或是友好而忙碌的沟通方式，总之，它深得人们的喜爱。

每月超过30%的增长当然不可能没有挑战，其中可靠性问题尤为艰巨。每天5亿次浏览量，峰值每秒4万次请求，每天3TB新的数据存储，并运行于超过1000台服务器上，所有这些帮助Tumblr实现巨大的经营规模。

创业公司迈向成功，都要迈过危险的迅速发展期这道门槛。寻找人才，不断改造基础架构，维护旧的架构，同时要面对逐月大增的流量，而且曾经只有4位工程师。这意味着必须艰难地选择应该做什么，不该做什么。这就是Tumblr的状况。好在现在已经有20位工程师了，可以有精力解决问题，并开发一些有意思的解决方案。

Tumblr最开始是非常典型的LAMP应用。目前正在向分布式服务模型演进，该模型基于Scala、HBase、Redis（著名开源K-V存储方案）、Kafka（Apache项目，出自LinkedIn的分布式发布-订阅消息系统）、Finagle（由Twitter开源的容错、协议中立的RPC系统），此外还有一个有趣的基于Cell的架构，用来支持Dashboard（CSDN注：Tumblr富有特色的用户界面，类似于微博的时间轴）。

Tumblr目前的最大问题是如何改造为一个大规模网站。系统架构正在从LAMP演进为最先进的技术组合，同时团队也要从小的创业型发展为全副武装、随时待命的正规开发团队，不断创造出新的功能和基础设施。下面就是Blake Matheny对Tumblr系统架构情况的介绍。

网站地址

http://www.tumblr.com/

主要数据

- 每天5亿次PV（页面访问量）
- 每月超过150亿PV
- 约20名工程师
- 峰值请求每秒近4万次
- 每天超过1TB数据进入Hadoop集群
- MySQL/HBase/Redis/memcache每天生成若干TB数据
- 每月增长30%

本周热点排行 更多
01 37signal创始人：SaaS开头容易收获难
02 黑客的方式：一切皆可自动化
03 Tumblr：150亿月浏览量背后的架构挑战（上）
04 NASA的大型电脑时代告终了，敬礼！
05 超级计算机模拟显示核弹可引爆碰撞地球小行星
06 免费云存储的秘密：贡献多余的存储
07 UNIX/Linux网络术语
08 揭秘Dell新罕布什尔的新办公室
09 印度Microsoft Store被黑，用户密码泄露
10 闪存时代来临：传统机械硬盘难满足云计算需求

热门招聘职位 更多
- 【上海骏聿数码】诚聘软件开发人员、研发经理。
- 【FT中文网】诚聘JavaScript 开发员、PHP Senior F
- 【杭州海康威视数字】诚聘英才C/C++开发工程师、
- 【CSDN】高薪急聘PHP开发/UI设计/网站编辑/社区
- 【广东南航天合信息】诚聘需求分析、网页前端开发
- 【杭州驰度】高薪诚聘C++搜索引擎工程师、windov

精彩专题 更多
Metro UI完全解析 国内云计算第一刊

- 近1000硬件节点用于生产环境
- 平均每位工程师每月负责数以亿计的页面访问
- 每天上传大约50GB的文章，每天跟帖更新数据大约2.7TB（CSDN注：这两个数据的比例看上去不太合理，据Tumblr数据科学家Adam Laiacano在Twitter上解释，前一个数据应该指的是文章的文本内容和元数据，不包括存储在S3上的多媒体内容）

软件环境

- 开发使用OS X，生产环境使用Linux（CentOS/Scientific）
- Apache
- PHP, Scala, Ruby
- Redis, HBase, MySQL
- Varnish, HAProxy, nginx
- memcache, Gearman（支持多语言的任务分发应用框架）, Kafka, Kestrel（Twitter开源的分布式消息队列系统）, Finagle
- Thrift, HTTP
- Func——一个安全、支持脚本的远程控制框架和API
- Git, Capistrano（多服务器脚本部署工具）, Puppet, Jenkins

硬件环境

- 500台Web服务器
- 200台数据库服务器（47 pool，20 shard）
- 30台memcache服务器
- 22台Redis服务器
- 15台Varnish服务器
- 25台HAproxy节点
- 8台nginx服务器
- 14台工作队列服务器（Kestrel + Gearman）

架构

1. 相对其他社交网站而言，Tumblr有其独特的使用模式：

- 每天有超过5千万篇文章更新，平均每篇文章的跟帖又数以百计。用户一般只有数百个粉丝。这与其他社会化网站里少数用户有几百万粉丝非常不同，使得Tumblr的扩展性极具挑战性。
- 按用户使用时间衡量，Tumblr已经是排名第二的社会化网站。内容的吸引力很强，有很多图片和视频，文章往往不短，一般也不会太长，但允许写得很长。文章内容往往比较深入，用户会花费更长的时间来阅读。
- 用户与其他用户建立联系后，可能会在Dashboard上往回翻几百页逐篇阅读，这与其他网站基本上只是部分信息流不同。
- 用户的数量庞大，用户的平均到达范围更广，用户较频繁的发帖，这些都意味着有巨量的更新需要处理。

2. Tumblr目前运行在一个托管数据中心中，已在考虑地域上的分布性。

3. Tumblr作为一个平台，由两个组件构成：公共Tumblelogs和Dashboard

- 公共Tumblelogs与博客类似（此句请Tumblr用户校正），并非动态，易于缓存
- Dashboard是类似于Twitter的时间轴,用户由此可以看到自己关注的所有用户的实时更新。与博客的扩展性不同，缓存作用不大，因为每次请求都不同，尤其是活跃的关注者。而且需要实时而且一致，文章每天仅更新50GB，跟帖每天更新2.7TB，所有的多媒体数据都存储在S3上面。
- 大多数用户以Tumblr作为内容浏览工具，每天浏览超过5亿个页面，70%的浏览来自Dashboard。
- Dashboard的可用性已经不错，但Tumblelog一直不够好，因为基础设施是老的，而且很难迁移。由于人手不足，一时半会儿还顾不上。

老的架构

Tumblr最开始是托管在Rackspace上的，每个自定义域名的博客都有一个A记录。当2007年Rackspace无法满足其发展速度不得不迁移时，大量的用户都需要同时迁移。所以他们不得不将自定义域名保留在Rackspace，然后再使用HAProxy和Varnish路由到新的数据中心。类似这样的遗留问题很多。

开始的架构演进是典型的LAMP路线：

- 最初用PHP开发，几乎所有程序员都用PHP
- 最初是三台服务器：一台Web，一台数据库，一台PHP
- 为了扩展，开始使用memcache，然后引入前端cache，然后在cache前再加HAProxy，然后是MySQL sharding（非常奏效）
- 采用“在单台服务器上榨出一切”的方式。过去一年已经用C开发了两个后端服务：ID生成程序和Staircar（用Redis支持Dashboard通知）

Dashboard采用了“扩散-收集”方式。当用户访问Dashboard时将显示事件，来自所关注的用户的事件是通过拉然后显示的。这样支撑了6个月。由于数据是按时间排序的，因此sharding模式不太管用。

新的架构

由于招人和开发速度等原因，改为**以JVM为中心**。目标是将一切从PHP应用改为服务，使应用变成请求鉴别、呈现等诸多服务之上的薄层。

这其中，非常重要的是**选用了Scala和Finagle**。

- 在团队内部有很多人具备Ruby和PHP经验，所以Scala很有吸引力。
- Finagle是选择Scala的重要因素之一。这个来自Twitter的库可以解决大多数分布式问题，比如分布式跟踪、服务发现、服务注册等。
- 转到JVM上之后，Finagle提供了团队所需的所有基本功能（Thrift, ZooKeeper等），无需再开发许多网络代码，另外，团队成员认识该项目的一些开发者。
- Foursquare和Twitter都在用Finagle，Meetup也在用Scala。
- 应用接口与Thrift类似，性能极佳。
- 团队本来很喜欢Netty（Java异步网络应用框架，2月4日刚刚发布3.3.1最终版），但不想用Java，Scala是不错的选择。
- 选择Finagle是因为它很酷，还认识几个开发者。

之所以**没有选择Node.js**，是因为以JVM为基础更容易扩展。Node的发展为时尚短，缺乏标准、最佳实践以及大量久经测试的代码。而用Scala的话，可以使用所有Java代码。虽然其中并没有多少可扩展的东西，也无法解决5毫秒响应时间、49秒HA、4万每秒请求甚至有时每秒40万次请求的问题。但是，Java的生态链要大得多，有很多资源可以利用。

内部服务从C/libevent为基础正在转向Scala/Finagle为基础。

开始采用新的**NoSQL存储方案**如HBase和Redis。但大量数据仍然存储在大量分区的MySQL架构中，并没有用HBase代替MySQL。HBase主要支持短地址生产程序（数以十亿计）还有历史数据和分析，非常结实。此外，HBase也用于高写入需求场景，比如Dashboard刷新时一秒上百万的写入。之所以还没有替换HBase，是因为不能冒业务上风险，目前还是依靠人来负责更保险，先在一些小的、不那么关键的项目中应用，以获得经验。MySQL和时间序列数据sharding（分片）的问题在于，总有一个分片太热。另外，由于要在slave上插入并发，也会遇到读复制延迟问题。

此外，还开发了一个**公用服务框架**：

- 花了很多时间解决分布式系统管理这个运维问题。
- 为服务开发了一种Rails scaffolding，内部用模板来启动服务。
- 所有服务从运维的角度来看都是一样的，所有服务检查统计数据、监控、启动和停止的方式都一样。
- 工具方面，构建过程围绕SBT（一个Scala构建工具），使用插件和辅助程序管理常见操作，包括在Git里打标签，发布到代码库等等。大多数程序员都不用再操心构建系统的细节了。

200台**数据库服务器**中，很多是为了提高可用性而设，使用的是常规硬件，但MTBF（平均故障间隔时间）极低。故障时，备用充足。

为了支持PHP应用有6个**后端服务**，并有一个小组专门开发后端服务。新服务的发布需要两到三周，包括Dashboard通知、Dashboard二级索引、短地址生成、处理透明分片的memcache代理。其中在MySQL分片上耗时很多。虽然在纽约本地非常热，但并没有使用MongoDB，他们认为MySQL的可扩展性足够了。

Gearman用于会长期运行无需人工干预的工作。

可用性是以达到范围（reach）衡量的。用户能够访问自定义域或者Dashboard吗？也会用错误率。

历史上总是解决那些最高优先级的问题，而现在会对故障模式系统地分析和解决，目的是从用户和应用的角度来定成功指标。（后一句原文似乎不全）

最开始Finagle是用于Actor模型的，但是后来放弃了。对于运行后无需人工干预的工作，使用任务队列。而且Twitter的util工具库中有Future实现，服务都是用Future（Scala中的无参数函数，在与函数关联的并行操作没有完成时，会阻塞调用方）实现的。当需要线程池的时候，就将Future传入Future池。一切都提交到Future池进行异步执行。

Scala提倡无共享状态。由于已经在Twitter生产环境中经过测试，Finagle这方面应该是没有问题的。使用Scala和Finagle中的结构需要避免可变状态，不使用长期运行的状态机。状态从数据库中拉出、使用再写回数据库。这样做的好处是，开发人员不需要操心线程和锁。

22台**Redis**服务器，每台的都有8-32个实例，因此线上同时使用了100多个Redis实例。

- Redis主要用于Dashboard通知的后端存储。
- 所谓通知就是指某个用户like了某篇文章这样的事件。通知会在用户的Dashboard中显示，告诉他其他用户对其内容做了哪些操作。
- 高写入率使MySQL无法应对。
- 通知转瞬即逝，所以即使遗漏也不会有严重问题，因此Redis是这一场景的合适选择。
- 这也给了开发团队了解Redis的机会。
- 使用中完全没有发现Redis有任何问题，社区也非常棒。
- 开发了一个基于Scala Futures的Redis接口，该功能现在已经并入了Cell架构。
- 短地址生成程序使用Redis作为一级Cache，HBase作为永久存储。
- Dashboard的二级索引是以Redis为基础开发的。
- Redis还用作Gearman的持久存储层，使用Finagle开发的memcache代理。
- 正在缓慢地从memcache转向Redis。希望最终只用一个cache服务。性能上Redis与memcache相当。

（先到这里吧，敬请期待下篇，包括如何用Kafaka、Scribe、Thrift实现内部活动流，Dashboard的Cell架构，开发流程和经验教训等精彩内容。）

翻译：包研，张志平，刘江；审校：刘江

英文原文出自High Scalability

相关文章
- 云计算技术产业研讨会暨中国云计算技术产业联
- 专家质疑:云计算扼杀应用开发?

网友评论(共8条评论)..

hao05010323 2012-02-17 11:25:24
完全不懂，做cs太久了
回复(0) 支持(0) 反对(0) 举报(0) | 0条回复..

• binglang8632 2012-02-17 11:25:06

确实才知道。先觉得点点网创意不错，现在才知道，还是令人恶心的抄袭。我们国人就只能干这个吗？

回复(0) 支持(0) 反对(0) 举报(0) | 0条回复..

• phpppk 2012-02-17 11:24:59

高仿

回复(0) 支持(0) 反对(0) 举报(0) | 0条回复..

• ideation_shang 2012-02-17 10:41:16

点点网不就是高仿 tumblr 吗 ？哈哈

回复(0) 支持(0) 反对(0) 举报(0) | 0条回复..

• iamxi 2012-02-17 10:33:08

第一次听说这个网站，落伍啦。的确很不错的网站。

回复(0) 支持(0) 反对(0) 举报(0) | 0条回复..

• Aselan 2012-02-17 09:45:41

主页很简洁很漂亮

回复(0) 支持(0) 反对(0) 举报(0) | 0条回复..

• AntiPro 2012-02-17 09:25:04

我访问了这个网站

We're sorry
Our servers are over capacity and certain pages may be temporarily unavailable.
We're working quickly to resolve the issue.

回复(0) 支持(0) 反对(0) 举报(0) | 0条回复..

• codeallen 2012-02-17 09:17:46

学习学习，天天向上。

回复(0) 支持(0) 反对(0) 举报(0) | 0条回复..

第一页 上一页 1 下一页 最末页

发表评论/共8条评论..

欢迎你，falcon05

发表评论

请您注意

- 自觉遵守：爱国、守法、自律、真实、文明的原则
- 尊重网上道德，遵守《全国人大常委会关于维护互联网安全的决定》及中华人民共和国其他各项有关法律法规
- 严禁发表危害国家安全，破坏民族团结、国家宗教政策和社会稳定，含侮辱、诽谤、教唆、淫秽等内容的作品
- 承担一切因您的行为而直接或间接导致的民事或刑事法律责任
- 您在CSDN新闻评论发表的作品，CSDN有权在网站内保留、转载、引用或者删除
- 参与本评论即表明您已经阅读并接受上述条款