

大型网站的架构研究及解决方案

周 强 谢 靖 赵华茗
(中国科学院文献情报中心 北京 100190)

摘 要 随着互联网业务的发展,网站规模越来越大,各种技术被提出以用于提升网站的性能、可用性、伸缩性、扩展性、安全性。在分析影响性能、可用性、伸缩性、扩展性和安全性等架构因素的基础上,提出了一套网站架构解决方案,并为图书馆集成发现系统的管理运维探索总结成功经验。

关键词 性能,可用性,伸缩性,扩展性,安全

中图法分类号 TP393.4 **文献标识码** A

Architecture and Solution for Large Web Sites

ZHOU Qiang XIE Jing ZHAO Hua-ming

(National Science Library, Chinese Academy of Sciences, Beijing 100190, China)

Abstract With the development of Internet business, the scale of website is getting bigger and bigger. Various technology was proposed to upgrade the performance, usability, scalability, expandability and security of website. Based on the analysis of effect performance, usability, scalability, expandability and security, a website schema solution programme was proposed, providing success experience for library integrated found system management operation exploration.

Keywords Performance, Usability, Scalability, Expandability, Security

1 引言

随着社会信息化建设不断发展,日常工作和生活越发依赖各种信息化系统,作为信息化载体的网站显得尤其重要,但用户访问量、业务数据的持续增加,对网站系统提出了更高的要求。因此如何在有限的资源前提下,尽可能地提高系统性能、可用性、伸缩性、扩展性,并保证系统的安全性,成为一个紧迫的问题^[1-4]。

数字图书馆联盟(DLF)图书馆集成发现工作小组于2011年指出,现代图书馆用户的需求发生了很大变化,他们希望图书馆集成发现系统能整合更多相关机构的资源,集成更灵活的检索方式和获取方式,集成更多的接口获取无缝迁移所发现或搜索到的信息数据。在此情境下,基于元数据预索引的网络级的资源发现系统应运而生,网络级的资源发现系统就是大型网站,常见的有EBSCO的Discovery Service(EDS)(2010)、Ex Libris的Primo(2010)、Serials Solutions的Summon(2009)、OCLC的World Cat Local(2007)。

本文分析了大型网站的关键架构和实现要素,在此基础上介绍了中国科学院文献情报中心的资源发现系统的架构设计。

2 网站核心架构要素分析及应用实例

大型网站有以下特点。

(1)数据海量:需要存储、管理海量数据,数据量数以亿

计,需要使用大量服务器。

(2)高可用:系统7*24小时不间断服务。

(3)高并发,大流量:需要面对高并发用户,大流量访问。

(4)安全环境恶劣:由于互联网的开放性,使得网站更容易受到攻击,网站几乎每天都会面对黑客攻击。

2.1 网站核心架构要素分析

2.1.1 性能

性能是网站架构设计的一个重要方面^[5-6]。

使用CDN,CDN的本质仍然是一个缓存,CDN部署在网络运营商的机房,这些运营商又是终端用户的网络服务提供商,因此用户请求路由的第一跳就到达了CDN服务器。当CDN中存在浏览器请求的资源时,从CDN直接返回给浏览器,实现最短路径响应,加快了用户访问速度,减少了数据中心的负载压力。一般CDN能够缓存静态资源,如图片、CSS、Script脚本、静态网页等,这些文件被访问频率很高,缓存在CDN可极大加快网页的打开速度。

使用反向代理服务器,反向代理服务器位于网站机房一侧,代理网站Web服务器接收HTTP请求,反向代理服务器具有保护网站安全的作用,来自互联网的访问请求必须经过代理服务器。除了安全功能,反向代理服务器可以通过配置缓存功能加速Web请求。当用户第一次访问静态内容时,静态内容就被缓存在反向代理服务器上,从而当其他用户访问该静态内容时,就可以直接从反向代理服务器返回。

应用服务器就是处理网站业务的服务器,网站的业务代

本文受基于开放获取学术期刊的资源深度整合与揭示研究(16BTQ025)资助。

周 强(1971—),男,硕士,馆员,主要研究方向为网络信息系统、分布式技术,E-mail:zhouqiang@mail.las.ac.cn;谢 靖(1984—),男,硕士,馆员,主要研究方向为网络信息系统、语义搜索;赵华茗(1970—),男,博士,副研究馆员,主要研究方向为网络技术。

码都部署在这里,优化手段主要是缓存、集群、异步等。

缓存指把数据存储于相对高速的存储介质中,以供系统处理。一方面,缓存访问速度快,可以减少数据访问的时间;另一方面,如果缓存的数据是经过计算处理得到的,那么被缓存的数据无需重复计算即可直接使用,因此缓存还起到减少计算时间的作用。缓存的本质是一个内存 Hash 表,在网站应用中,数据缓存以 1 个〈Key, Value〉的形式存储在内存 Hash 表中。Hash 表数据读写的时间复杂度为 $O(1)$ 。

网站数据访问通常遵循二八定律,即 80% 的访问落在 20% 的数据上,因此利用 Hash 表和内存的高速访问特性,将这 20% 的数据缓存起来,可以很好地改善系统性能,提高数据读取速度,降低存储访问压力。

分布式缓存指缓存部署在多个服务器组成的集群中,以集群的方式提供缓存服务,其架构方式有两种:1) 以 JBoss Cache 为代表的需要更新同步的分布式缓存;2) 以 Memcached 为代表的互不通信的分布式缓存。JBoss Cache 多见于企业应用系统中,而很少在网站使用。Memcached 曾一度是网站分布式缓存的代名词,被大量网站使用,具有简单的设计、优异的性能、互不通信的服务器集群、海量数据可伸缩的架构,网站架构师们纷纷采用^[7-8]。

使用消息队列将调用异步化,可改善网站的扩展性。事实上,使用消息队列还可改善网站系统的性能。

在不使用消息队列的情况下,用户的请求直接写入数据库。在高并发的情况下,这会对数据库造成巨大的压力,同时也使得响应延迟加剧。在使用消息队列后,用户请求的数据发送给消息队列后立即返回,再由消息队列的消费者进程从消息队列中获取数据,异步写入数据库。由于消息队列服务器的处理速度远快于数据库,因此用户的响应延迟可得到有效改善。

消息队列具有很好的削峰作用,即通过异步处理,将短时间高并发产生的事务消息存储在消息队列中,从而削平高峰期的并发事务。

在使用消息队列进行业务异步处理后,需要适当修改业务流程以进行配合。

将多台应用服务器组成一个集群共同对外服务,提高整体处理能力,改善性能。

在代码层面,可以通过使用多线程、改善内存管理等手段优化性能。

在数据库服务器端,索引、缓存、SQL 优化等性能优化手段都已经比较成熟。而方兴未艾的 NoSQL 数据库通过优化数据模型、存储结构、伸缩特性等手段在性能方面的优势也日趋明显。

2.1.2 可用性

网站使用的服务器硬件通常是普通的商用服务器,这些服务器的设计目标本身并不保证高可用,也就是说,很有可能出现服务器硬件故障,即俗称的服务器宕机。网站高可用架构设计的前提是必然会出现服务器宕机,高可用设计的目标就是当服务器宕机时,服务或者应用依然可用。

高可用的主要手段是冗余,应用部署在多台服务器上同时提供访问,数据存储在台服务器上互相备份,任何一台服务器宕机都不会影响应用的整体可用,也不会导致数据丢失。

对于应用服务器,多台应用服务器通过负载均衡设备组

成一个集群共同对外提供服务,任何一台服务器宕机,只需把请求切换到其它服务器就可实现应用的高可用,但是一个前提条件是在应用服务器上不能保存请求的会话信息,否则服务器宕机,会话丢失,即使把用户请求转发到其他服务器上也无法完成业务处理。

由于存储服务器存储着数据,需要对数据进行实时备份,当服务器宕机时,需要将数据访问转移到可用的服务器上,并进行数据恢复以保证数据依然可用。

除了运行环境,网站的高可用还需要软件开发过程的质量保证。网站需要保证 7×24 小时高可用运行,同时又需要发布新功能以满足用户的需求。代码在发布到线上服务器之前需要进行严格的测试,可以采用自动测试工具或脚本完成测试。即使经过严格的测试,软件部署到线上服务器之后还是经常会出现各种问题,甚至根本无法启动服务器。因此在网站发布时,应先发布到预发布机器上,开发工程师和测试工程师在预发布服务器上进行预发布验证,执行业务流程,确认系统没有问题后才正式发布。可以采用灰度发布模式,把集群服务器分成若干部分,每天只发布一部分服务器,观察运行是否稳定以及是否存在故障,第二天继续发布一部分服务器,持续几天才把整个集群全部发布完毕,期间如果发现问题,只需要回滚已发布的一部分服务器即可。通过这些手段,减少将故障引入线上环境的可能^[9]。

衡量一个系统架构设计是否满足高可用的目标,就是假设系统中任何一台或者多台服务器宕机时以及出现各种不可预期的问题时,系统整体是否依然可用。

2.1.3 伸缩性

网站通过集群的方式将多台服务器组成一个整体共同提供服务。所谓伸缩性是指通过不断向集群中加入服务器的手段来缓解不断上升的用户并发访问压力和不断增长的数据存储需求。

衡量架构伸缩性的主要标准即为是否可以用多台服务器构建集群;向集群中添加新的服务器是否容易;加入新的服务器后是否可以提供和原来服务器无差别的服务;集群中可容纳的总的服务器数量是否有限制。

对于应用服务器集群,只要服务器上不保存数据,所有服务器都是对等的,通过使用合适的负载均衡设备就可以向集群中不断加入服务器。

对于缓存服务器集群,加入新的服务器可能会导致缓存路由失效,进而导致集群中大部分缓存数据都无法被访问。虽然缓存的数据可以通过数据库重新加载,但是如果应用已经严重依赖缓存,可能会导致整个网站崩溃。需要改进缓存路由算法以保证缓存数据的可访问性。

关系数据库虽然支持数据复制、主从热备等机制,但是很难做到大规模集群的可伸缩性,因此关系数据库的集群伸缩性方案必须在数据库之外实现,通过路由分区等手段将部署有多个数据库的服务器组成一个集群。

至于大部分 NoSQL 数据库产品,由于其先天就是为海量数据而生,因此其对伸缩性的支持通常都非常好,可以做到在较少运维参与的情况下实现集群规模的线性伸缩。

一个具有良好伸缩性架构设计的网站的设计总是走在业务发展的前面,在业务需要处理更多访问和服务之前,就已经做好充足准备,当业务需要时,只需要购买或者租用服务器进

行简单部署实施就可以了。

2.1.4 扩展性

网站的扩展性架构直接关注网站的功能需求。

衡量网站架构扩展性好坏的主要标准就是在网站增加新的业务产品后是否可以实现对现有产品透明无影响,不需要任何改动或者很少改动即有业务功能就可以上线新产品;不同产品之间是否很少耦合,若一个产品的改动对其他产品无影响,则其他产品和功能不需要受牵连而进行改动。

网站可扩展架构的主要手段是事件驱动架构和分布式服务。

事件驱动架构通常利用消息队列实现,将用户请求和其他业务事件构造成消息发布到消息队列,消息的处理者作为消费者从消息队列中获取消息进行处理。通过这种方式将消息产生和消息处理分离,可以透明地增加新的消息生产者任务或者新的消费者任务。

分布式服务是将业务和可复用服务分离,通过分布式服务框架来调用。新增产品可以通过调用可复用的服务实现自身的业务逻辑,而对现有产品没有任何影响。可复用服务升级变更时,也可以通过提供多版本服务对应用实现透明升级,不需要强制应用同步变更。

2.1.5 安全性

网站的安全架构旨在保护网站不受恶意访问和攻击,保护网站的重要数据不被窃取。

网站应用的攻击与防御:1)跨站点脚本攻击一般都是在请求中嵌入恶意脚本,通过对某些 html 危险字符转义来防止大部分攻击。2)SQL 注入攻击,在 HTTP 请求中注入恶意 SQL 命令;可以通过请求参数消毒、参数绑定来防止攻击。3)跨站请求伪造,攻击者通过跨站请求,以合法用户的身份进行非法操作;可以通过表单 Token、验证码和 Referer 检查来防止攻击。

为了保护网站的敏感数据,应用需要对这些数据进行加密处理,信息加密技术可分为 3 类:单项散列加密、对称加密和非对称加密。

衡量网站安全架构的标准就是针对现存和潜在的各种攻击与窃密手段,是否有可靠的应对策略。

2.2 应用实例

Google、百度等商业化信息服务系统给图书馆带来了相当大的冲击,针对该局面,为了使现有的图书馆信息服务系统跟上 Web2.0 的步伐从而更好地为用户服务,一些开发商纷纷推出了具备 Web2.0 特征的系统,例如 ProQuest 公司开发的 Summon、EBSCO 公司的 EBSCO Discovery Service 等,我们统称这类系统为新一代资源发现系统。新一代资源发现系统的主要特征有:

- 1)提供一站式检索,拥有简洁友好的 Google 式界面;
- 2)整合了图书馆的印本资源和电子资源,集成范围扩展到图书馆外的资源,如网络资源等;
- 3)提供更丰富的信息组织和揭示方式,帮助用户发现资源,如分面检索/浏览、相关性排序、多种格式输出等。

总体说来,新一代资源发现系统主要致力于全面揭示图书馆各种类型的资源,集成多种与资源发现和获取相关的服务,充分调动网络可用资源和读者的智慧,从而成为读者与图书馆资源良性互动的平台,最终实现读者对各类型信息资源

的最大化利用。它围绕资源发现而提供一系列服务,包含文献检索功能,还包含全文下载、文献传递、馆际互借、参考咨询、引文等文献获取和利用服务。

3 大型集成发现系统网站的架构设计

中国科学院文献情报中心开发的资源发现系统集成成了本图书馆的图书、期刊、期刊论文、会议论文、学位论文,还集成了开放获取数据(arXiv、PubMed Central 等),下一步计划集成 SpringerLink 期刊、维普中文科技期刊(中国科技经济新闻)数据库等商业数据库,数据量将达到数十亿,因此这是大型集成发现系统。

大型集成发现系统的所有计算机的操作系统是 Linux。

服务器机房的入口部署反向代理服务器,如果请求的资源在反向代理服务器的缓存中,则直接返回响应,否则把请求向前传到负载均衡调度服务器。

负载均衡调度服务器按照设定好的算法,把请求传到应用服务器。

应用服务器有本地缓存,如果请求的资源在本地缓存中,则直接返回响应。如果请求的资源在分布式缓存服务器中,则返回响应。应用服务器运行应用程序,从搜索引擎服务器、NoSQL 服务器、文件服务器、分布式数据库服务器中读取数据,生成响应并返回客户端。

反向代理服务器采用 Squid。Squid 是由美国政府大力资助的一项研究计划,其目的是解决网络带宽不足的问题,支持 HTTP、HTTPS、FTP 等多种协议,是目前 Linux 系统上使用最多、功能最完整的代理软件。

负载均衡调度服务器采用 LVS,即 Linux 虚拟服务器,一个由章文嵩博士于 1998 年发起的开源项目,现在 LVS 已经是 Linux 内核标准的一部分。通过 LVS 可以实现一个高性能、高可用的 Linux 服务器群集,它具有良好的可靠性、可拓展性和可操作性,从而以低廉的成本实现最优的性能^[10]。

应用服务器采用 Tomcat。Tomcat 是 Apache 软件基金会的 Jakarta 项目中的一个核心项目。Tomcat 技术先进,性能稳定,而且免费,是目前比较流行的 Web 应用服务器。

在架构设计时, Tomcat 部署为集群。

搜索引擎服务器采用 Solr。Solr 是一个高性能、基于 Lucene 的全文搜索服务器。对其进行了扩展,提供了比 Lucene 更为丰富的查询语言,同时实现了可配置、可扩展并对查询性能进行了优化,提供了一个完善的管理界面,是一个非常优秀的全文搜索引擎。在架构设计时, Solr 部署为集群。所有的资源数据都创建为 Solr 索引(倒排索引),检索速度比较快。

分布式文件服务器采用 Hadoop HDFS。HDFS 是一个高度容错性的系统,适合部署在廉价的机器上。HDFS 能提供高吞吐量的数据访问,非常适合大规模数据集上的应用。

NoSQL 服务器采用 HBase。HBase 是一个高可靠性、高性能、面向列、可伸缩的分布式存储系统,利用 HBase 技术可在廉价计算机上搭建起大规模结构化存储集群。

HBase 存储所有资源的数据。

分布式缓存服务器采用 Memcached,前面已对其进行了介绍。

分布式数据库采用 MySQL Cluster。MySQL Cluster 是

一种允许在无共享的系统中部署“内存中”数据库的 Cluster 的技术。通过无共享体系结构,系统能够使用廉价的硬件,而且对软硬件无特殊要求。此外,每个组件有自己的内存和磁盘,不存在单点故障。MySQL Cluster 由计算机集群构成,存储的数据包括:1)馆藏图书期刊目录的订购数据、借阅数据;2)电子期刊、数据库的订购数据;3)文献传递、馆际互借的图书馆数据;4)参考咨询的咨询人员数据。

结束语 本文分析了现有网站相关技术,在分析影响性能、可用性、伸缩性、扩展性和安全性等架构因素的基础上,提出了一套网站架构解决方案,并将其应用于实际系统中。网站的架构会随着性能、可用性、伸缩性、扩展性的提升变得越来越复杂,并且没有唯一性,只有不断地摸索寻找适合自身的拓扑结构。

参 考 文 献

- [1] 李钢. 大型公共服务网站架构设计初探[J]. 环球市场信息导报, 2015(39):72.

(上接第 566 页)

供服务,说明无单点故障(SPoF)产生。连接断开节点,可以快速重置负载均衡并提供服务。

3) 断开任一 OpenStack 云平台控制节点,测试确认 OpenStack 控制节点相关服务组件、数据库和消息队列集群均能提供正常服务,然后依次测试网络节点、计算节点、存储节点。

4) 断开 Ceph 集群的少量节点,OpenStack 云平台中的虚拟机实例仍能够正常运行,Ceph 出现警告,但能够正常调用 Ceph 存储集群。修复错误,Ceph 警告消失,丢失节点快速加入 Ceph 集群并提供正常服务。

通过以上 4 个实验得出,此高可用性方案可以对单点故障及服务实例故障进行检测和处理,可自动切换各个服务节点,实现了 OpenStack 云平台的高可用服务。可以将服务实例资源在高可用节点之间进行快速切换,实现不间断地提供云平台服务,保证虚拟机实例稳定服务。

结束语 本文首先简要介绍了云计算和系统高可用性的相关概念以及 OpenStack 云计算管理平台的相关服务组件实现高可用性的方式。其次,结合现有的计算资源环境,提出一种基于 Pacemaker + Corosync + HAProxy + Ceph 的解决方案,实现 OpenStack 云计算管理平台的高可用性并实现实验云平台的建设。该方案将 Active-Active 的双活模式、Active-Passive 的主备模式及集群技术 3 种高可用设计模式融合在一起,通过软硬件冗余和服务实例故障转移方式等实现 OpenStack 云计算管理平台的高可用性。最后,利用现有的计算资源,结合该高可用性解决方案,构建 OpenStack 云计算管理平台。通过实践证明,该高可用性实验云平台方案具有有效性和可行性。

参 考 文 献

- [1] 程宏兵,赵紫星,叶长河. 基于体系架构的云计算安全研究进展[J]. 计算机科学,2016,43(7):19-27.

- [2] 胡华海,王新宇. 浅析大型网站的基础架构[J]. 科技风,2009(4):167.
- [3] 林昊. 大型网站架构演变和知识体系[J]. 程序员,2008(11):66-69.
- [4] 韩树河. 大型网站应用技术架构演变的研究[J]. 吉林化工学院学报,2015(1):53-56.
- [5] 杨舟. 浅析大型网站的性能优化[J]. 软件工程师,2010(12):38-40.
- [6] 房辉,常盛. 大型网站高性能架构研究[J]. 信息系统工程,2015(12):76-77.
- [7] 俞华锋. Memcached 在大型网站中的应用[J]. 科技信息(科学教研),2008(1):70.
- [8] 周建儒. Memcached 在大型网站建设中的应用[J]. 电脑知识与技术,2016(1):57-59.
- [9] 赵丽荣. 大型门户网站运行维护服务模式探讨[J]. 中国高新技术企业,2011(21):71-73.
- [10] 章文嵩. 可伸缩网络服务的研究与实现[D]. 长沙:中国人民解放军国防科学技术大学,2000.

- [2] 宋俊锋. 基于 MILP 的云计算数据中心扩张策略优化模型[J]. 湘潭大学自然科学学报,2015,37(4):105-110.
- [3] CORRADI A, FANELLI M, FOSCHINI L. VM consolidation: A real case based on OpenStack Cloud[J]. Future Generation Computer Systems, 2014, 32(1):118-127.
- [4] 张帆,李磊,杨成胡,等. 基于 Eucalyptus 构建私有云计算平台[J]. 电信科学,2011,27(11):57-61.
- [5] KESSACI Y, MELAB N, TALBI E G. A multi-start local search heuristic for an energy efficient VMs assignment on top of the OpenNebula cloud manager[J]. Future Generation Computer Systems, 2014, 36(3):237-256.
- [6] MILOJČIĆ D, LLORENTE I M, MONTERO R S. OpenNebula: A Cloud Management Tool [J]. Internet Computing IEEE, 2011, 15(2):11-14.
- [7] 于飞. 基于 openNebula 云平台实验及性能评估[D]. 北京:北京邮电大学,2013.
- [8] PARADOWSKI A, LIU L, YUAN B. Benchmarking the Performance of OpenStack and CloudSack[C]//IEEE, International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing. IEEE, 2014:405-412.
- [9] 陈星,张颖,张晓东,等. 基于运行时模型的多样化云资源管理方法[J]. 软件学报,2014,25(7):1476-1491.
- [10] 马友忠,慈祥,孟小峰. 海量高维向量的并行 Top-k 连接查询[J]. 计算机学报,2015,38(1):86-98.
- [11] 马友忠,孟小峰. 云数据管理索引技术研究[J]. 软件学报,2015, 26(1):145-166.
- [12] ANGLANO C, CANONICO M, GUAZZONE M. FC2Q: exploiting fuzzy control in server consolidation for cloud applications with SLA constraints[J]. Concurrency and Computation: Practice and Experience, 2015, 27(17):910-915.
- [13] TROPE R L, RESSLER E K. Mettle Fatigue: VW's Single-Point-of-Failure Ethics[J]. IEEE Security & Privacy Magazine, 2016, 14(1):12-30.
- [14] 杨祥. 无线传感器网络无标度容错拓扑的连锁故障诊断算法[J]. 计算机应用研究, 2016, 33(2):549-551.