

<<实用负载均衡技术>>

图书基本信息

书名：<<实用负载均衡技术>>

13位ISBN编号：9787115314635

10位ISBN编号：7115314632

出版时间：2013-5

出版时间：人民邮电出版社

作者：[英] Peter Membrey,[澳] David Hows

译者：武海峰,陈晓亮

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<实用负载均衡技术>>

### 内容概要

《实用负载均衡技术:网站性能优化攻略》内容简介：处理负载均衡问题的相关概念和工具，说明了如何避免性能退化和服务器上的服务突然崩溃的风险，阐述了单个服务器以及可以执行cookie插入或者改善SSL吞吐量的负载均衡器，最后还探讨了云计算中的负载均衡。

## <<实用负载均衡技术>>

### 作者简介

Peter Membrey是获得认证的IT专业人员，拥有15年以上使用Linux和开源方案的实战经验。他17岁就通过了RHCE（Red Hat认证工程师）认证，成为最年轻的认证得主，还曾有幸为Red Hat工作，撰写过介绍开源方案的著作。

Eelco Plugge 本科毕业后就一直对信息安全领域很感兴趣。

他曾在McAfee担任数据加密专家。

现在，他在利物浦大学攻读信息安全硕士学位，并利用零散时间写书。

他持有多个专业证书，并热衷于Linux、网络安全和加密数据等诸多技术方面。

David Hows 来自澳大利亚，以优异成绩毕业于信息与通信技术专业。

他热衷系统性能优化和系统安全，并一直坚持在这个枯燥的领域工作。

## &lt;&lt;实用负载均衡技术&gt;&gt;

## 书籍目录

第1章 引言 1 1.1 性能问题 1 1.2 解决方案 2 1.3 什么是负载均衡 3 1.3.1 负载均衡的前世 3 1.3.2 负载均衡的今生 3 1.3.3 纵向扩展 3 1.3.4 横向扩展 4 1.4 负载均衡的实现 4 1.4.1 网络的构成 4 1.4.2 缓存：网站的曲速引擎 5 1.4.3 使用DNS进行负载均衡 5 1.4.4 内容分发网络 5 1.4.5 6P原则 6 1.4.6 基础知识 6 1.4.7 HTTP负载均衡 6 1.4.8 对数据库进行负载均衡 7 1.4.9 对网络连接进行负载均衡 7 1.4.10 SSL负载均衡 7 1.4.11 建立高可用性集群 7 1.4.12 云平台上的负载均衡 7 1.4.13 IPv6：实现和概念 8 1.4.14 下一步做什么 8 1.5 总结 8

第2章 网站工作原理 9 2.1 开始我们的旅程 10 2.1.1 来自非IT背景 10 2.1.2 开始浏览的过程 10 2.1.3 通过DNS查找网站 11 2.1.4 最终连接到服务器 11 2.1.5 服务器自身 12 2.1.6 连接到数据库 12 2.1.7 缓存技术速览 12 2.1.8 回传到客户端 13 2.2 进一步了解 13 2.3 网络 14 2.3.1 TCP 14 2.3.2 DNS 16 2.3.3 速度、带宽和延迟 17 2.3.4 网络连接小结 19 2.4 HTML和Web 19 2.4.1 HTML 20 2.4.2 为什么基于文本很重要 20 2.4.3 为什么链接很重要 21 2.4.4 HTML小结 22 2.4.5 浏览器 22 2.5 Web内容 23 2.5.1 静态内容 23 2.5.2 动态内容 24 2.5.3 创建动态内容 24 2.5.4 Web内容小结 25 2.6 数据库：最薄弱的环节 26 2.7 总结 26

第3章 内容缓存：保持低负载 28 3.1 什么是缓存 29 3.2 走马观花 29 3.2.1 基于浏览器的缓存 29 3.2.2 Web加速器 30 3.2.3 Web代理 31 3.2.4 透明Web代理 32 3.2.5 边缘缓存 33 3.2.6 平台缓存 34 3.2.7 应用缓存 35 3.2.8 数据库缓存 36 3.2.9 仅仅是个开始 3.3 缓存理论：缓存为什么这么难 36 3.3.1 HTTP1.0对缓存的支持 37 3.3.2 HTTP1.1加强的缓存支持 37 3.3.3 解决方案 38 3.3.4 缓存不像看起来那么简单 39 3.4 Web代理 39 3.4.1 Squid代理服务 39 3.4.2 开始了 40 3.4.3 故障排除 41 3.4.4 透明代理 42 3.4.5 发生了什么 42 3.4.6 获得帮助 44 3.4.7 Squid，代理中的瑞士军刀 44 3.5 边缘缓存：Varnish 45 3.5.1 默认保守缓存 46 3.5.2 安装Varnish 46 3.5.3 配置并运行 47 3.5.4 定制Varnish 49 3.6 总结 49

第4章 基于DNS的负载均衡 51 4.1 DNS内幕 51 4.1.1 IP地址 51 4.1.2 问题 52 4.1.3 解决方案 52 4.1.4 回退一步 53 4.2 DNS详解 53 4.2.1 亲自查询 54 4.2.2 DNS查询进阶 55 4.3 DNS缓存 56 4.3.1 查询DNS缓存 56 4.3.2 Linux系统上的DNS缓存 58 4.3.3 实质内容 58 4.4 BIND9 58 4.4.1 DNSDB的头 60 4.4.2 DNS数据库记录 61 4.4.3 加载数据库 62 4.4.4 检查配置文件 63 4.4.5 常见问题 63 4.4.6 测试DNS 63 4.5 基于DNS的负载均衡 64 4.5.1 基于DNS的负载均衡的优势 65 4.5.2 基于DNS的负载均衡的问题 65 4.6 总结 66

第5章 内容分发网络 67 5.1 选择CDN服务提供商 68 5.2 开始使用Rackspace 68 5.3 向CDN账户添加内容 69 5.4 Rackspace云文件API 72 5.4.1 将API集成到PHP中 72 5.4.2 用API密钥进行认证 72 5.4.3 建立连接和断开连接 73 5.4.4 对容器进行操作 74 5.4.5 对文件进行操作 79 5.4.6 其他有用的函数 84 5.5 总结 86

第6章 性能和可靠性计划 87 6.1 yoUMakeDinnerInTiME 87 6.1.1 理解 88 6.1.2 决策 90 6.1.3 设计与实现 91 6.1.4 安装 92 6.1.5 测试、维护、评估 93 6.1.6 计划的重要性 95 6.2 备份 96 6.2.1 为什么备份如此重要 96 6.2.2 前方可能有麻烦 97 6.2.3 必须实现自动化 98 6.2.4 战术备份 98 6.2.5 战略备份 98 6.2.6 增量备份与全备份 99 6.2.7 一定，一定要测试恢复！ 99 6.3 总结 100

第7章 负载均衡基础 101 7.1 什么是负载均衡 101 7.2 有哪些可用的计算资源 102 7.2.1 处理器 102 7.2.2 内存 103 7.2.3 使用top命令查看CPU和RAM的性能 103 7.2.4 网络 104 7.2.5 存储（磁盘） 105 7.3 负载均衡实战 105 7.4 指导原则 106 7.4.1 深入理解系统 106 7.4.2 规划 106 7.4.3 监测和测试 107 7.5 总结 107

第8章 对网站进行负载均衡 108 8.1 测量Web服务器的性能 108 8.2 加速ApacheHTTP 110 8.2.1 禁用空载模块 111 8.2.2 禁用DNS查询 111 8.2.3 采用压缩 112 8.2.4 FollowSymLinks和SymLinksIfOwnerMatch选项 113 8.3 加速nginx 114 8.3.1 worker\_processes和worker\_cpu\_affinity 114 8.3.2 Gzip压缩 115 8.4 对Web服务器进行负载均衡 116 8.4.1 配置 117 8.4.2 准备IPVS服务器 118 8.4.3 准备工作服务器 120 8.4.4 测试负载均衡器 121 8.5 划分动态和静态内容 122 8.6 总结 123

第9章 对数据库进行负载均衡 124 9.1 搭建MySQLCluster 124 9.1.1 安装管理程序 126 9.1.2 配置管理程序 126 9.1.3 准备集群数据节点 129 9.1.4 安装MySQLServer和NDB守护进程 129 9.1.5 配置NDB守护进程 130 9.1.6 启动集群节点上的服务 131 9.1.7 更新MySQL的root用户 132 9.1.8 测试上述安装和配置 133 9.2 实施负载均衡 135 9.2.1 建立负载均衡 135 9.2.2 设置负载均衡服务器 136 9.2.3 设置工作服务器 137 9.2.4 测试负载均衡服务器 138 9.3 总结 139

第10章 对网络进行负载均衡 140 10.1 分担负载 140 10.2 TCP/IP 141 10.2.1 TCP 141 10.2.2 IP 142 10.3 路由 142 10.4 负载均衡服务器 144 10.5 IPVS 146 10.5.1 IPVS的调度方式 146 10.5.2 在Ubuntu上安装IPVS 148 10.5.3 在CentOS上安装IPVS 149 10.6 IPVSADM 150 10.7 扩展IPVS 151 10.8 IPVS进阶 153 10.8.1 修改调度算法 153 10.8.2 分配权重 154 10.8.3 协议与多台虚拟服务器 155 10.8.4 增加IP地址 156 10.9 保存设置 158 10.10 总结 159

第11章

## &lt;&lt;实用负载均衡技术&gt;&gt;

对SSL进行负载均衡 160 11.1 什么是SSL和TLS 160 11.2 公钥密码学 161 11.3 信任和数字证书认证机构 162  
11.4 TLS加密 162 11.5 TLS负载均衡 162 11.6 配置Web服务器上的SSL 163 11.6.1 配置Apache服务器上的SSL  
165 11.6.2 配置nginx服务器上的SSL 166 11.7 SSL加速 166 11.7.1 在Apache上启用SSL加速 166 11.7.2 在nginx  
上启用SSL加速 166 11.8 SSL前端 167 11.9 测试SSL 168 11.10 进一步配置 170 11.10.1 在SSL前端中启用SSL  
加速 170 11.10.2 启用缓存 171 11.10.3 指定要支持的协议 172 11.10.4 指定加密方法 172 11.11 LVS和SSL终  
结前端 173 11.12 将负载均衡服务器 / SSL终端功能集成到同一台服务器上 175 11.13 总结 175 第12章 使  
用集群提高可用性 176 12.1 高可用性 177 12.2 单一故障点 178 12.3 集群化 179 12.4 IPVS故障恢复 180  
12.4.1 在Ubuntu上安装集群软件包 181 12.4.2 在CentOS上安装集群软件包 182 12.4.3 配置集群 182 12.4.4  
常见配置问题 185 12.4.5 检查系统 186 12.5 测试 187 12.6 Web服务器细节配置 189 12.6.1 Ubuntu 189 12.6.2  
CentOS 189 12.7 高级配置选项 189 12.7.1 ha.cf 189 12.7.2 ldirectord.cf 190 12.7.3 Web服务器 190 12.8 总结  
190 第13章 云端负载均衡 191 13.1 云计算 191 13.2 虚拟化 192 13.3 虚拟化资源 195 13.4 管理虚拟资源 196  
13.4.1 平衡 197 13.4.2 超量供给 198 13.4.3 计划 199 13.5 云的弹性 200 13.6 用云服务器工作 201 13.7 总结  
203 第14章 IPv6：影响和概念 204 14.1 IPv6 204 14.2 十六进制表示 204 14.3 缩略表示 205 14.4 IPv4地址的  
耗尽 205 14.5 部署IPv6 205 14.6 IPv6的优势 206 14.7 实现 206 14.8 互联网连接 208 14.9 DNS 208 14.10 操作  
系统 208 14.11 网络 208 14.11.1 单一网关的网络 209 14.11.2 双重网络 209 14.12 软件支持 209 14.12.1  
Apache 209 14.12.2 nginx 210 14.12.3 Varnish 210 14.12.4 Memcached 210 14.12.5 IPVS 211 14.12.6 ldirectord  
211 14.12.7 heartbeat 211 14.13 总结 212 第15章 何去何从 213 15.1 回顾 213 15.2 监控 214 15.3 安全 215 15.3.1  
访问控制 215 15.3.2 视图 216 15.3.3 常见的攻击防护 216 15.4 操作系统性能 217 15.4.1 自己编译 217 15.4.2  
裁剪 218 15.4.3 高性能操作系统 218 15.5 计划 218 15.6 总结 219 索引 220

## &lt;&lt;实用负载均衡技术&gt;&gt;

## 章节摘录

版权页：插图：一旦发现了一个特殊的问题，而且需要解决这一问题，那你可以在维护周期中解决该问题。

维护周期其实只做两件事：解决你认为紧要的问题，清除系统在日常使用中产生的垃圾。比如说，你可以清理临时文件，或者整理数据库——任何可以让应用运行得更流畅的事。

评估就是后退一步，看看初始的解决方案是否真的解决了问题。

在理想情况下，你最好一次搞定，然后在维护周期就可以以某种方式改进或者增强系统。不过，一般都需要好几次才能切中要害。

Time周期随着解决方案的使用而持续。

对于简单的或者成熟的系统，这个周期几乎是冻结的，每六个月到一年才可能发生变化。

对于较新的系统，可能每天都会有变化。

部署系统到生产环境几乎不可能不是你最后一次碰这个系统。

几乎总是会突然出现什么事，需要从简单的微调，到几乎完全重新编译的种种不同应对措施（你要是不信，就看看下面框注中的千年虫问题）。

所以，尽管你不需要干坐着发愁，但对于你构建并部署的任何东西都非常可能在什么时候需要改变这件事，你要做到心里有数。

再次停下来，想想这一步发生了什么，以及你的最终目标。

如果你的解决方案设计为了易于支持和维护，而且你在整个过程中也专注于这个目标（就像这是你原计划的一部分，而且你也坚持这个计划是你做的那样），那么到这个阶段的时候，你就会有一个易于维护和支持的解决方案。

种瓜得瓜，种豆得豆，但是要很努力地确保你确实在种你想种的（在正确的地点、正确的时间，但愿是在你自己的地里）。

如果你时不时地停下来检查一下罗盘，来确保自己是在向正确的方向前进，那么最终你会得到一个多得多的解决方案，这个解决方案将会极其易于使用。

千年虫问题（以2000年命名）可能是随着新千年的到来最广为人知的问题。

有人害怕核导弹发射井会混乱并点火发射，地球会遭受毁灭。

或者，领取养老金的人岁数会突然变成负数，于是从此以后再也领不到钱了。

所有这些歇斯底里都是一个简单的问题造成的——一个在最初根本不是问题的问题。

事实上，那是个降低成本、提高处理速度、更有效地利用昂贵的计算机的好办法。

这个问题的出现归咎于用两位数字而不是四位数字来存储年份。

比如说，系统只会存储83而不是1983。

当10M的内存都只能在价值100万美元的计算机上才有时，能节省一半的存储是一个很好的主意！

在那个时候，谁也没想到到了2000年会出问题，毕竟距2000年还有30年，到时候肯定没人再用这些系统了！

你猜怎么着？

我们还在使用那些系统，而且很多公司也直到很晚才发现问题。

如果你碰巧在1997年就知道COBOL了，在那片充满恐慌的阴霾中，光是升级系统就可以挣到让你一辈子衣食无忧的钱。

比如说，很多系统会用当前年份减去出生年份来计算一个人的年龄。

用之前的例子，就是2012—1983=29。

然而，如果你用最后两位数字，这个人一下子就变成了一71岁。

一个一71岁的人一年的车险应该是多少？

## <<实用负载均衡技术>>

### 媒体关注与评论

这是一本学习云架构的非常好的图书，精彩的内容包括：负载均衡服务器、缓存及SSL。  
——亚马逊读者评论

## <<实用负载均衡技术>>

### 编辑推荐

网站负载均衡架构全揭秘，完美应对云环境及大数据的挑战，网站性能优化必备指南，从整体上来看，《实用负载均衡技术：网站性能优化攻略》是一本比较好的负载均衡入门书籍，内容也较新（已出版的几本相关英文著作都较早）。



<<实用负载均衡技术>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>