改进的最小链接负载均衡调度算法①

陈燕升[1], 张赞波[1], 任江涛[2]

[1](广东轻工职业技术学院, 广州 510300)
[2](中山大学 软件学院, 广州 510275)

摘 要: 在有限资源情况下, FTP 服务资源共享平台仅凭多个独立的服务器无法承担访问量、数据容量、网络流量等快速增长所需要的运算需求,采用最小连接调度(LVS)策略设计并实现了改进的最小链接负载均衡调度算法, 在通过增加备用调度服务器方法设计的调度服务器冗余资源共享平台中评测传输速度、负载用户数和磁盘读取速度等方面的性能, 结果表明文中设计的资源共享平台性能有显著提高, 这对促进有限资源条件下的高效资源共享平台构建技术发展具有实际意义.

关键词: FTP 服务; 资源共享平台; LVS; 最小链接调度; 负载均衡

Improved Minimum Link Load Balancing Scheduling Algorithm

CHEN Yan-Sheng[1], ZHANG Zan-Bo[1], REN Jiang-Tao[2]

[1](Guangdong Industry Technical College, Guangzhou 510300, China)
[2](School of Software, Sun Yat-Sen University, Guangzhou 510275, China)

Abstract: In case of limited resources, FTP service resource sharing platform cannot afford the rapid growth of visits, data capacity and network traffic required for computing needs with single server. This paper adopts a minimum connection scheduling policy design, achieves improved minimum link load balancing scheduling algorithm and evaluates the performance of the transmission speed, loading users number and disk read speed and other aspects in the redundant resource sharing platform of scheduling server designed by increasing the backup scheduling servers. The results shows that the performance of the resource sharing platform designed in this paper has significantly improved, which has practical significance in promoting the technology of building efficient resource sharing platform under conditions of limited resources.

Key words: FTP Servers; resource sharing platform; the minimum link scheduling; LVS; load balancing technology

FTP 服务资源共享平台是一种软、硬件资源受限的资源载体, 随着访问量、数据容量、网络流量的快速增长, 其数据计算要求和处理能力也要相应的增强, 使得仅凭单个服务器无法承担如此巨大的运算[1]. 在这种情况下, 如果完全放弃现有的设备去把全部硬件进行重新升级, 将会浪费现有的硬件资源, 而且在下一次业务量提升的时候, 这次所升级的硬件又可能因为达不到提升的要求而再次需要投入更高的成本, 即使是性能再好的独立设备也会因为业务量需求的不断增多而被淘汰. 在现有网络的基础上, 通过适量增加一些独立的 FTP 服务器, 改变单一独立的 FTP 服务器承载为多独立的 FTP 服务器来分担, 无疑是一种很好的选择. 文中主要研究在有限资源基础上, 基于传统的备用调度服务器的冗余算法, 采用最小连接调度策略设计并实现了改进的最小链接(LVS)负载均衡调度算法, 以建立一种新的负载分配均衡机制实现更高效的资源共享平台,这对促进有限资源条件下的高效资源共享平台构建技术发展具有实际意义.

① 基金项目:国家自然科学基金(11471342);广东省教育科研"十二五"规划 2012 年度研究项目(2012JK089);广东省高等职业教育教学改革研究项目(201401034);校级自然科学基金(KJ2013110,KJ201417)
收稿时间:2014-11-26;收到修改稿时间:2015-01-07

1 调度服务器的冗余设计

在 FTP 服务资源共享平台中, 所有服务器的物理链路均可使用链路聚合方式的技术, 后端配置了一组真实服务器[2], 存储设备配置 RAID5, 这些配置均具有冗余功能, 完全可以保证平台不会因某一个单点故障而导致整个资源共享服务停止. 由于调度服务器只有一台, 虽然在物理链路上有冗余, 但是, 当操作系统出现故障时还是会让整个平台不能正常地提供服务.

为了保证资源共享平台能长期在线, 一则可以通过 DNS 冗余查询的方法分配到各相应的真实服务器上[3]. 当调度服务器失效时, 后端的真实服务器会变成分布式的独立服务器, 由于系统使用统一的存储方式, 各真实服务器提供的服务内容也一致. 就而, 仅仅需要查询已经设置好的 DNS 的冗余, 既能将用户的请求直接分配到相应服务器的真实 IP 地址上[4]. 但是由于调度服务器的单点故障是通过 DNS 方式来解决, 当调度服务器恢复正常时再重新配置 DNS, 使其指向另一个调度服务器. 此方法需要手动修改资源共享平台之外的服务, 加上 DNS 缓存的因素, 因此相对较繁琐, 效率也较低.

二则可以通过增加备用调度服务器的方法. FTP 服务资源共享平台会因为主调度服务器故障等原因而不工作, 通常可通过新增备用调度服务器的方式实现冗余, 并利用 Heartbeat[5]的守护进程监听主调度服务器的心跳数据包[6]. 当备用调度服务器未监听到主调度服务器发送的心跳时, 备用调度服务器将发送 GARP 广播, 以便网络上的其他机器能知道备用调度服务器所拥有的 Virtual IP 地址, 并利用备用调度服务器所创建 IPVS 表, 向后端的真实服务器转发用户的请求, 使得资源共享平台可以继续的运作, 并通过监控系统对管理员发出的通知, 以便同时对主调度服务器进行相应的维护和修复, 如此循环可以保证该资源共享平台的高可用性.

归纳表述, Heartbeat 通过设置主服务器和备用服务器, 并在备用服务器上执行 Heartbeat 的守护进程监听主服务器的心跳数据包. 当备用服务器未收到主服务器的心跳请求时, 则自动启动故障转移机制从而代替主服务器来提供调度任务等工作[7]. 当故障的主服务器恢复后, 会自动变成备用服务器, 然后再监听当前正在提供服务的调度服务器的心跳, 如此循环, 达到冗余的功能.

调度服务器的冗余设计需要在资源共享平台中加入备用调度服务器, 并设置 Heartbeat 守护进程和故障转移机制, 它可以通过不断的循环来保证平台的长期在线, 不需修对平台之外的服务, 这样提高了平台的整合度, 减轻了管理员的负担, 明显优于只通过 DNS 冗余查询的方式.

2 负载均衡的调度策略和改进的调度算法

最小连接调度算法首先由均衡调度器统计所有服务器的连接数, 得到一张服务器负载的升序排序表, 完成后监听网络请求; 如果有新的连接请求访问, 则把新的请求分配给在负载排序表中最靠前的服务器, 该服务器的连接数增加 1, 参与下一轮的服务器连接数的统计, 并更新排序表. 最小连接调度算法是一个动态调度算法, 它通过当前活跃的连接数评估服务器集群负载状况[8].

在实际的系统设计中, 一般会引入权值来表示服务器负载状况, 权值为零时表示该服务器不可用且不能在下一周期被调度[9]. 在一组服务器集群中, 设服务器为 $S=\{S_0,S_1,...,S_{n-1}\}$, $W_C(S_i)$表示第 i 个服务器权值, $C(S_i)$表示第 i 个服务器的连接数, 常见的最小连接权调度算法的步骤如下:

Step1: 统计所有服务器的连接数 $C(S_i)$, 完成从 $C(S_0)$到 $C(S_i)$的迭代排列, 形成升序排序表, 完成后监听网络请求;

Step2: 当网络侦听到新的连接请求访问 $C(S_i)$, 则把新的请求分配给在负载排序表中最靠前的服务器 $W_C(S_i)$, 该服务器的连接数 $C(S_i)$增加 1;

Step3: $W_C(S_i)$参与下一轮的服务器连接数 $C(S_i)$的统计, 转为 Step1.

当每个服务器的处理性能都相同时, 该算法能够把访问请求平滑分布调度到各个服务器上[10]. 但是当每个服务器的处理性能不一致的时候, 该算法的执行效果非常差, 尤其是在每个请求任务的负荷量变化很大的时候. 一个很可能发生的情况是, 当TCP连接处理新的访问请求后就进入等待状态, 等待时间一般保持 2 分钟, 在这个时间段里, 新的请求连接是占用着服务器资源的, 所以性能高的服务器已经处理所收到的连接, 连接处于等待状态, 而性能低的服务器已经忙于处理所收到的连接, 还不断地收到新的连接请求[11].

因此, 文中将采用最小连接调度策略, 在以上研究的基础上设计一种改进的最小链接负载均衡调度算法, 表述如下:

定义 1. 设 C_i 为第 i 个访问请求量, R_i 为服务第 i 个请求所分配的资源量, N_j 为第 j 个服务器在网络中的吞吐量, G 为整个网络的带宽, P_j 为第 j 台服务器的处理能力极值, 那么第 j 台服务器的可负载率可以用式(1)表达:

$$M(j)=\left(1-\sum_{i=1}^{C}(C_iR_i)\Big/P_j\right)N(j)/G \tag{1}$$

定义2. 设 P_{min} 为集群中处理性能最低的服务器的处理能力极值, 该值表征了集群中服务器处理能力的最小值, $P_{min}=\min(P_1,P_2,\ldots,P_n)$, 则某访问请求的归一化负载率 L 可用式(2)表达:

$$L = R_i/P_{min} \tag{2}$$

可负载率 L 表明了当前服务器的负载情况, L 值越大, 表明服务器的负载越小, 剩余处理能力越大, 它能接受新访问请求的处理能力越强, 也即还可接收更多的负载, 也表示该服务器可被分配新请求的概率越大. 反之, 则该服务器的剩余处理能力不足, 在下一调度周期需要避免再次被分配处理任务.

依据对改进的最小连接调度算法的分析, 得出其算法的步骤如下:

Step1: 均衡器定时评估各个服务器的状态, 主要包括负载情况、处理性能极值和网络状态;

Step2: 按定义 1 的表述计算所有服务器的可负载率并排序, 更新可负载率表;

Step3: 网络侦听到新的请求, 均衡器评估新请求需要占用多少处理资源, 按照定义 2 及式(2)计算出归一化负载率;

Step4: 评估新请求的归一化负载率, 对比可负载率表的情况, 如果超出最大的可负载率, 则向用户返回错误信息; 否则分配给可负载率表中排第一位的服务器服务;

Step5: 更新接受服务的服务器可负载率, 转为 step1.

改进算法的流程图如图 1 示.

算法的时间复杂度分析如下:

根据算法的流程, 算法的主要工作是在计算服务器节点的状态和新请求的归一化负载率, 很明显算法的时间复杂度为 $O(n^2)$, n 是服务器节点的个数. 均衡器一直轮询各服务器节点的状态, 只需占用很小的带宽和服务资源, 几乎可以忽略不计, 因此该算法不会增加集群的负荷, 算法能不断地执行下去.

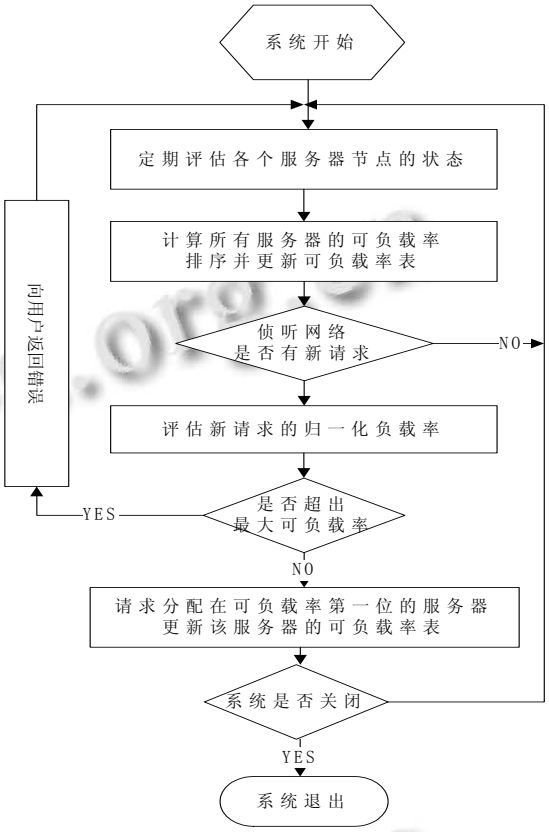

图 1　改进的最小连接调度算法流程图

3　资源共享平台的性能测试与评价

在基于负载均衡的 FTP 服务资源共享平台搭建成功后, 需要对资源共享平台进行相关的性能测试, 以评价设计效果和后续改进. 文中通过模拟在大数据量下载资源的应用条件下对资源共享平台的负载、响应能力以及最大用户数进行压力测试以获得测试数据 . 并依此来分析资源共享平台在以上负载压力情况下的传输速度、最大吞吐量已经能够支持的最大用户数, 进而确定运行瓶颈, 评价文中设计方案的处理能力.

3.1 负载均衡的 FTP 服务资源共享平台

3.1.1 准备和规划

① 以一台 Director 机器和 2 台 Real Server 为例进行实现, Director 端为 Red hat Linux10.0 系统, Real Server 端为 Windows 2000 sever 操作系统.

② 网络拓扑结构配置:

Load Director (eth0):　192.150.0.1,

Virtual IP (VIP): 192.150.0.2
Real Server 1 IP: 192.150.0.4,
Virtual IP (VIP): 192.150.0.3,
Real Server 2 IP: 192.150.0.5,
Virtual IP (VIP): 192.150.0.3
客户端访问地址为 VIP: 192.150.0.3

③ LVS 软件包: ipvs-2.0.20.tar 和 ipvsadm-2.21.tar, 其中后者完全是用来与 IPVS 交互的命令行工具集, 在负载均衡器上编译、安装以上程序包.

3.1.2 安装

① 将 Linux 的内核打补丁(即 ipvs2.0.20.tar 软件包), 并重新编译内核补丁包, 然后启用新内核, 操作过程如下:

```
cd /usr/src/ ipvs-2.0.20
make patchkernel     /*patch ipvs to Linux kernel*/
make installsource   /*install ipvs to Linux kernel*/
```

② 安装 ipvsadm2.21 工具. 在重启 Linux 系统之后, 选择进入有检测到新内核的 linux 系统, 安装 ipvsadm2.21, 用来管理 ipvs2.0.20, 其操作过程如下:

```
cd /usr/src/ipvsadm1.21
make                 /*编译模块*/
make install         /*安装模块*/
```

③ 执行 ip vsadm 命令, 并得到 IPVS 的新版本提示, 表示 ipvsadm 已经安装成功, 其操作如下:IP Virtual Server version 1.0.10……

3.1.3 配置

① Director 端(负载均衡器)的配置:

```
/sbin/ipvsadm -A -t 192.168.0.3:80 -s wlc
/sbin/ipvsadm -a -t 192.168.0.3:80 -r 192.168.0.4 -g -w 1
/sbin/ipvsadm -a -t 192.168.0.3:80 -r 192.168.0.5 -g -w 1
```

② Real Server 端配置:

在 VS/DR 的方式中, 支持 Windows 2000 作为 Real Server, 并且必须安装了 MS Loopback Adapter.

Windows 2000 上安装 MS Loopback Adapter Driver 的步骤如下: 进入控制面板->选择添加新硬件->添加网络适配器->选择 Loopback Adapter. 点击添加 VIP(Virtual IP)地址在 MS Loopback Adapter 上(操作方法同修改 IP 地址相类似), 但是要注意 LVS 中对要求真实服务器 IP 子网掩码必须为自定的值 255.255.255.255, 但这在 Windows 2000 中通常会被认为是无效的地址. 所以其处理方法为修改注册表的方式设置子网掩码(HKEY_LOCAL_MACHINE\ System\ CurrentControlSet\Services\Tcpip\Parameters\Interfaces).

以上方式通过软件方法解决服务器的负载过重问题, 提高了整个系统的伸缩性. 调度器借助系统资源监控软件(如 Mon)自动屏蔽掉产生故障的服务器, 然后将用户的请求分发给正常的真实服务器, 并等待故障服务器恢复正常. 当故障服务器恢复工作后, 调度器再将该服务器重新加入服务器组中, 这就构成高性能的虚拟服务器集群系统[12]. 在响应频繁的应用系统中, 通过负载均衡技术配置调控无疑是非常好的选择.

3.2 资源共享平台传输速度的性能评测

对服务器系统的测试主要按三个步骤进行, 即传统模式下的单一 FTP 服务器负载测试, 传统模式下的多 FTP 服务器集群负载测试和基于 FTP 负载均衡的服务测试, 同时对模拟大用户量的并发访问与阶段测试的侧重点不同.

利用跨平台的传输速度测试软件 iperf 对该服务器系统的带宽进行了 2 批次测试. 测试时均使用默认的测试参数传输, 以保证两批测试数据的准确性. 表 1 分别给出了平台搭建前服务器使用单个网络适配器情况的传输速度和平台搭建后服务器使用了链路聚合条件下两个网络适配器传输度的测试数据.

表 1 服务器传输速度测试数据

序号	第 1 批测试结果 传输速度(Mbit/s)		第 2 批测试结果 传输速度(Mbit/s)	
	单个网络适配器	聚合后的速度	单个网络适配器	聚合后的速度
1	841	1788	820	1879
2	808	1736	835	1832
3	617	1734	860	1794
4	825	1746	821	1787
5	838	1782	800	1830
6	870	1748	827	1850
7	745	1885	812	1825
8	828	1892	859	1855
9	820	1800	833	1801
10	843	1738	824	1755

由表 1 可知, 两批次测试所得到结果如下: 单个网络适配器的平均传输速度为 816.3Mbit/s, 两个网络适配器使用链路聚合后的平均传输速度为 1802.85Mbit/s, 后者约为前者的 2.2 倍, 单服务器相对于集群服务器的方差为 D(X)=3000 Mbit/s2. 测试结果表明聚合适配器的传输速度远高于单个适配器的传输

速度, 这将有助于提高网络服务器资源的调配和利用.

3.3 资源共享平台负载用户数性能评测

在资源共享平台负载用户数性能评测方面, 文中实际测试的原系统使用两台独立的Serv-U 6.1.0.1服务器提供 FTP 服务, 一台主要提供电子书籍、教程资源等, 另一台提供电影、连续剧等资源, 单台服务器所能承载的最高用户数为 200. 与基于最小连接调度策略设计改进的最小链接负载均衡调度算法搭建的包括一台主调度服务器, 一台备用调度服务器(VSFTP 2.2.2)的方案. 尽管两套方案所能承载的最高用户数没有变化(均为 400), 但文中所设计的方案具有以下优点:

①用户接口更简单, 原方案需要用户记住 2 个 ip, 而现方案仅记住 1 个.

②权限管理也更为统一, 管理员不需要分别维护两个, 而新方案两台真实服务器完全对称, 仅需同步配置文件即可;

③解决了原方案负载不均匀的问题.

新方案不仅可以保证两台真实服务器具有相当的负载, 更重要的是针对原方案缺乏水平的伸缩性(对 FTP 所提供的任何内容最多能负载的用户均为 200), 而新方案只需要适当增加真实服务器的数量, 就可实现在一定程度的水平可伸缩性. 如果追加一台真实服务器, 其所能负载的用户上限将几乎可以接近 600.

3.4 资源共享平台磁盘读取速度的性能评试

评测分别使用了多操作系统平台下的 HDtune 和 HDparm磁盘测试工具进行了读取速度的测试. 表2给出了对原单个磁盘数据读取速度与所组建成的 RAID5 的磁盘阵列读取速度的测试结果的对比.

表 2 磁盘读取速度测试结果

序号	第 1 批测试结果 平均读取速度(MB/s)		第 2 批测试结果 平均读取速度(MB/s)	
	单个磁盘	RAID5	单个磁盘	RAID5
1	54.4	254.7	48.5	207.9
2	50.3	236.3	54.4	238.2
3	48.2	288.2	67.7	197.5
4	59.9	216.4	63.4	226.7
5	52	294.3	54.8	285.3
6	63.5	289.1	48.2	205.6
7	48.4	280.9	67.4	184.6
8	64.4	184.1	57.4	226.3

从表 2 可知, 单个磁盘数据的平均读取速度测试值为53.34MB/s, RAID5磁盘阵列的平均读取速度测试值为 298.54MB/s, 后者约为前者的 5.28 倍, 单个磁盘相对于RAID5的磁盘阵列读取速度的方差D(x)=652.5 MB/s. 这个测试结果表明RAID5的读取速度远高于单个磁盘的读取速度.

4 结语

文中采用最小连接调度策略设计并实现了改进的最小链接负载均衡调度算法, 并在通过增加备用调度服务器的方法设计调度服务器的冗余的资源共享平台中评测传输速度、负载用户数和磁盘读取速度等方面的性能, 结果表明文中设计方案的资源共享平台性能有显著提高. 但改进的最小链接负载均衡调度算法在评估新请求的归一化负载率, 对比可负载率表的情况, 如果超出最大的可负载率, 设计为向用户返回错误信息, 并终止负载, 可否有更好的方法处理, 有待进一步研究.

参考文献

1 Leino J, Virtamo J. Insensitive load balancing in data networks. Computer Networks, 2006, 50(8): 1059–1068.

2 Cao JW, Spoonerb DP, Jarvisb SA, Nuddb GR. Grid load balancing using intelligent agents. Future Generation Computer Systems, 2005, 21(1): 135–149.

3 王琼,杨冬,高德云.兼容 DNS 的一体化网络资源解析系统. 计算机技术与发展,2013,(1):1–4.

4 王小娅,何玲.DNS 功能在 Cisco 设备中的应用.长春大学学报,2012,22(2):163–165.

5 史文路,胡平.双机热备份系统的研究与改进.微处理机, 2008,29(3):180–182.

6 李临风.集群系统的动态负载均衡算法的研究和实现[学位论文].南京:东南大学,2010.

7 杨锐,张玉洲,杨霖,陈家元.高校大型仪器设备管理系统的开发.天津工程师范学院学报,2006,16(1):45–47.

8 乔治.基于 LINUX 负载均衡系统的研究与实现[学位论文]. 北京工业大学,2005.

9 王燕.web 服务器集群负载均衡技术研究[硕士学位论文]. 西安:西安电子科技大学,2007.

10 鞠光明.校园网负载平衡调度算法研究[硕士学位论文]. 南京:南京理工大学,2006.

11 陈宣.KYLIN 中虚拟服务器系统研究与实现[硕士学位论文].长沙:国防科学技术大学,2005.

12 庞辽军,王力,李慧贤.基于集群技术的 Linux 虚拟服务器. 计算机工程与应用,2003,39(14):161–163.