最新随笔

1. socket泄露的问题
2. gdb 调试多线程
3. MMAP和DIRECT IO区别
4. 三年回首：C基础
5. 定时器管理：nginx的红黑树和libevent的堆
6. strsep和strtok_r替代strtok
7. 缓存穿透和缓存失效
8. mmap为什么比read/write快(兼论buffercache和pagecache)
9. B+Tree和MySQL索引分析
10. c++拷贝构造和编译优化

我的标签

linux(9)
c/c++(8)
data structure(7)
tcp/ip(7)
operating system(4)
database(4)
Optimization(3)
interview(3)
algorithm(1)
network programming(1)
更多

积分与排名

积分 - 170574
排名 - 2061

阅读排行榜

1. WEB服务器、应用程序服务器、HTTP服务器区别(64358)
2. shell中exec解析(57616)
3. python中的编码问题：以ascii和unicode为主线(45572)
4. Linux进程调度原理(42859)
5. Linux内存管理原理(42478)
6. 正确解读free -m(38443)
7. mysql事务和锁InnoDB(35684)
8. JavaScript和JQuery好书推荐(27560)
9. c语言libcurl库的异步用法(25029)
10. VIM插件攻略(22318)

评论排行榜

1. mysql事务和锁InnoDB(9)
2. 80X86寄存器详解(8)
3. HTTP报文(4)
4. 公司大了怎么办(4)
5. Linux内存管理原理(4)
6. Linux进程调度原理(3)
7. shell中exec解析(3)
8. WEB服务器、应用程序服务器、HTTP服务器区别(3)
9. 可重入性与线程安全(2)
10. 糊涂窗口综合症和Nagle算法(2)

负载均衡与HTTP加速

1. 负载均衡技术简介

现代企业信息化应用越来越多的采用B/S应用架构来承载企业的关键业务，因此，确保这些任务的可靠运行就变得日益重要。随着越来越多的企业实施数据集中，应用的扩展性、安全性和可靠性也越来越受到企业的重视。

负载均衡技术通过设置虚拟服务器IP（VIP），将后端多台真实服务器的应用资源虚拟成一台高性能的应用服务器，通过负载均衡算法，将大量来自客户端的应用请求分配到后端的服务器进行处理。负载均衡设备持续的对服务器上的应用状态进行检查，并自动对无效的应用服务器进行隔离，实现了一个简单、扩展性强、可靠性高的应用解决方案。解决了单台服务器处理性能不足，扩展性不够，可靠性较低的问题。

近年来，随着Web2.0和B/S技术的迅猛发展，HTTP应用逐渐成为当今的主流应用，而负载均衡技术也有了很大的发展。从传统的基于四层端口号进行简单的应用请求转发，到目前基于七层内容进行请求的转发和处理。尤其是在HTTP协议的优化和加速方面，一些技术逐渐发展成熟，如：TCP连接复用、内容缓存、TCP缓冲、HTTP压缩、SSL加速等。这些技术的应用有助于进一步改善用户访问响应时间、节约广域网链路带宽和服务器资源。

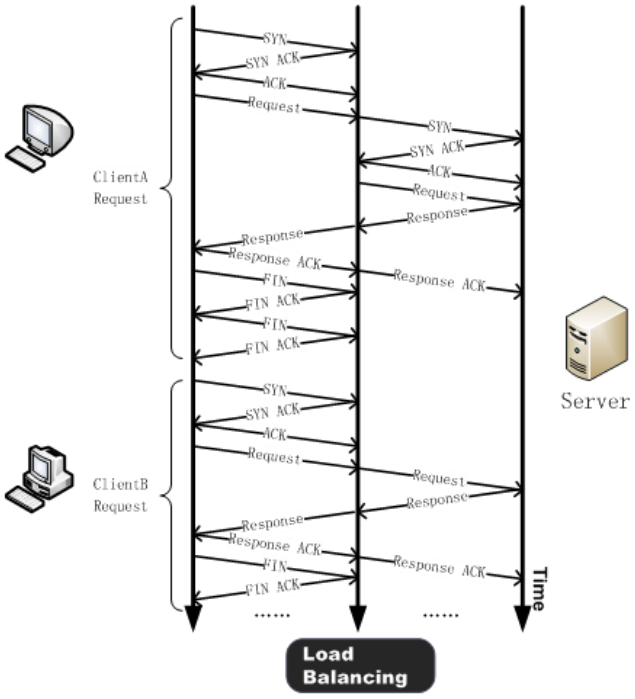
2. HTTP优化和加速特性带来的益处

2.1. TCP连接复用（TCP Connection Reuse）

TCP连接复用技术通过将前端多个客户的HTTP请求复用到后端与服务器建立的一个TCP连接上。这种技术能够大大减小服务器的性能负载，减少与服务器之间新建TCP连接所带来的延时，并最大限度的降低客户端对后端服务器的并发连接数请求，减少服务器的资源占用。

一般情况下，客户端在发送HTTP请求之前需要先与服务器进行TCP三次握手，建立TCP连接，然后发送HTTP请求。服务器收到HTTP请求后进行处理，并将处理的结果发送回客户端，然后客户端和服务器互相发送FIN并在收到FIN的ACK确认后关闭连接。在这种方式下，一个简单的HTTP请求需要十几个TCP数据包才能处理完成。

采用TCP连接复用技术后，客户端（如：ClientA）与负载均衡设备之间进行三次握手并发送HTTP请求。负载均衡设备收到请求后，会检测服务器是否存在空闲的**长连接**，如果不存在，服务器将建立一个新连接。当HTTP请求响应完成后，客户端则与负载均衡设备协商关闭连接，而负载均衡则保持与服务器之间的这个连接。当有其它客户端（如：ClientB）需要发送HTTP请求时，负载均衡设备会直接向与服务器之间保持的这个空闲连接发送HTTP请求，避免了由于新建TCP连接造成的延时和服务器资源耗费。



图例 1 TCP连接复用（TCP Connection Reuse）

在HTTP 1.0中，客户端的每一个HTTP请求都必须通过独立的TCP连接进行处理，而在HTTP 1.1中，对这种方式进行了改进。客户端可以在一个TCP连接中发送多个HTTP请求，这种技术叫做HTTP复用（HTTP Multiplexing）。它与TCP连接复用最根本的区别在于，TCP连接复用是将多个客户端的HTTP请求复用到一个服务器端TCP连接上，而HTTP复用则是一个客户端的多个HTTP请求通过一个TCP连接进行处理。前者是负载均衡设备的独特功能；而后者是HTTP 1.1协议所支持的新功能，目前被大多数浏览器所支持。

建立的TCP连接总数的比值。但是， TCP连接复用率和应用的特点、服务器设置、计算周期以及请求的发送模式等也有很大的关系，不同的应用环境下计算出来的TCP连接复用率会有很大的差异。其实，连接复用效率的关键在于负载均衡设备是否能够及时释放已经空闲的服务器端连接。有些厂商采用发送HTTP响应后等待一定时间，如果这段时间内无数据传输即释放该连接。而等待时间往往是秒级的，对于数据往返时间的毫秒级，其复用效果明显不会很好。**最为有效的连接复用技术是在负载均衡设备给客户端发送HTTP响应之后，收到客户端确认ACK数据包即释放该连接**。这种方式避免了任何额外的等待时间，理论上没有更高效的复用方法。

2.2. 内容缓存（RAM Caching）

内容缓存技术将应用服务器中的一些经常被用户访问的热点内容缓存在负载均衡设备的内存中。当客户端访问这些内容时，负载均衡设备截获客户端请求，从缓存中读取客户端需要的内容并将这些内容直接返回给客户端。由于是直接从内存中读取，这种技术能够提高网络用户的访问速度，并大大减轻后端服务器的负载情况。

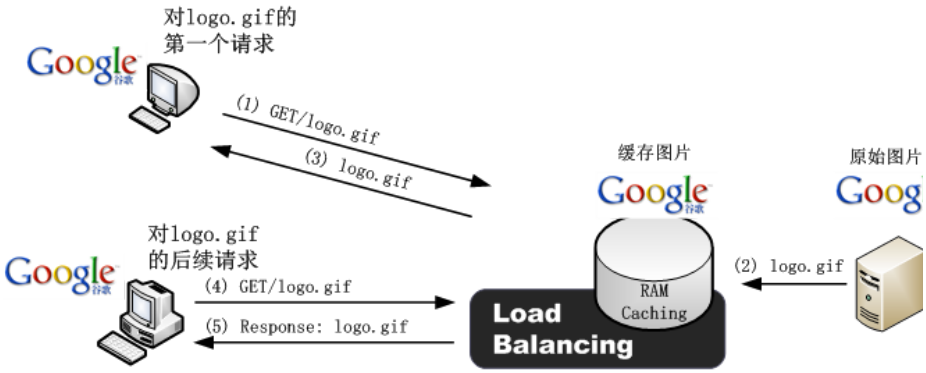
内容缓存的工作原理非常简单，我们将通过下图用户访问logo.gif的实例来解释内容缓存的工作过程：

1、当有客户端发起对logo.gif的第一个请求时，负载均衡首先会检查本地缓存中是否存在该对象。如果不存在这个对象，负载均衡会将这个HTTP请求转发给后端的服务器；

2、服务器收到对logo.gif的HTTP请求后，将图片内容回应给负载均衡设备；

3、负载均衡设备将logo.gif对象缓存在内容缓存中，并将其发送给客户端；

4、后续的其它客户端发起对logo.gif的访问请求时，如果负载均衡检测到内容缓存中已经存在该对象，并确认该对象并未失效的话，负载均衡直接将该对象返回给客户端，而无需服务器再次发送该对象。



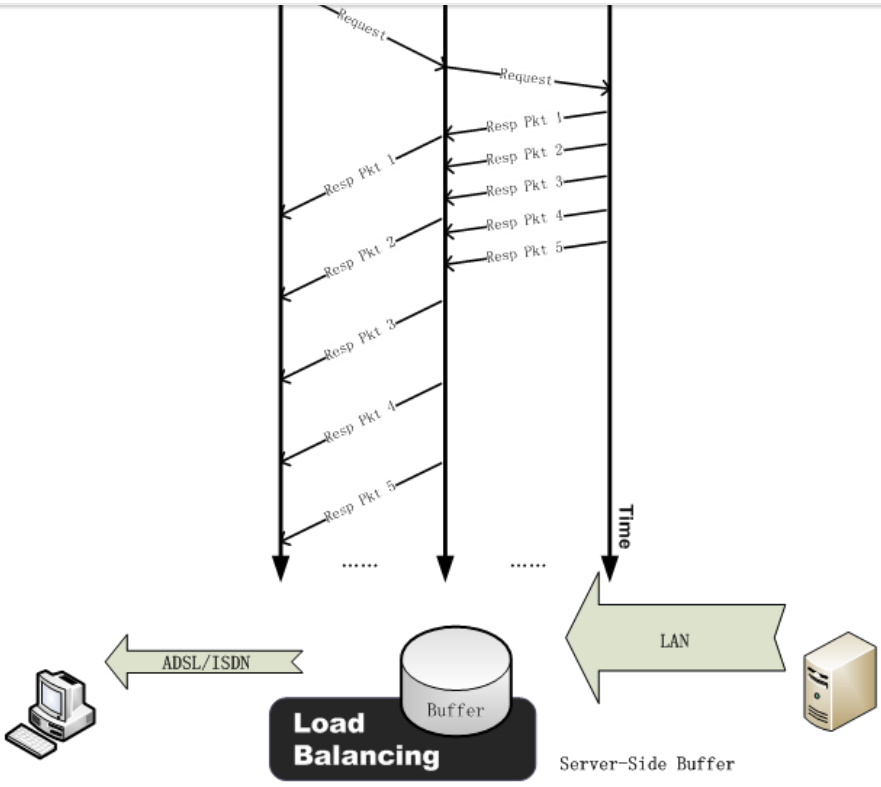
图例 2 内容缓存（RAM Caching）

内容缓存技术采用了HTTP协议中的一些标准缓存处理技术，将本应保存在客户端本地浏览器缓存中的内容共享给其他用户。因此，对于客户端来说，内容缓存技术是完全透明的。最常见的对象包括：gif/jpg图片，静态的css/js/html等文本文件等。

2.3. TCP缓冲（TCP Buffer）

TCP缓冲是为了解决后端服务器网速与客户的前端网络速度不匹配而造成的服务器资源浪费的问题。由于服务器与负载均衡设备之间的网络带宽速率高，时延小，通过将服务器端的请求缓冲在负载均衡设备的缓冲区中，防止由于客户端缓慢的网络链路和较高的时延造成服务器端连接阻塞问题。

通过采用TCP缓冲技术，可以提高服务器端响应时间和处理效率，减少由于通信链路问题给服务器造成的连接负担。另外，由负载均衡设备来处理网络阻塞造成的数据包重传，使每个客户端的流量得到最佳的控制。



图例 3 TCP缓冲（TCP Buffer）

如上图所示，客户端与负载均衡之间采用的链路具有较高的时延和较低的带宽，而负载均衡与服务器之间采用时延较低和高带宽的局域网连接。

1、负载均衡收到客户端发来的HTTP请求并将其转发给后端的服务器进行处理；

2、服务器对请求进行处理后，将响应的内容依次返回负载均衡设备，负载均衡设备收到响应的数据包后，会将数据包依次缓存在缓冲区中，服务器的响应速度将依据负载均衡和服务器之间的链路质量；

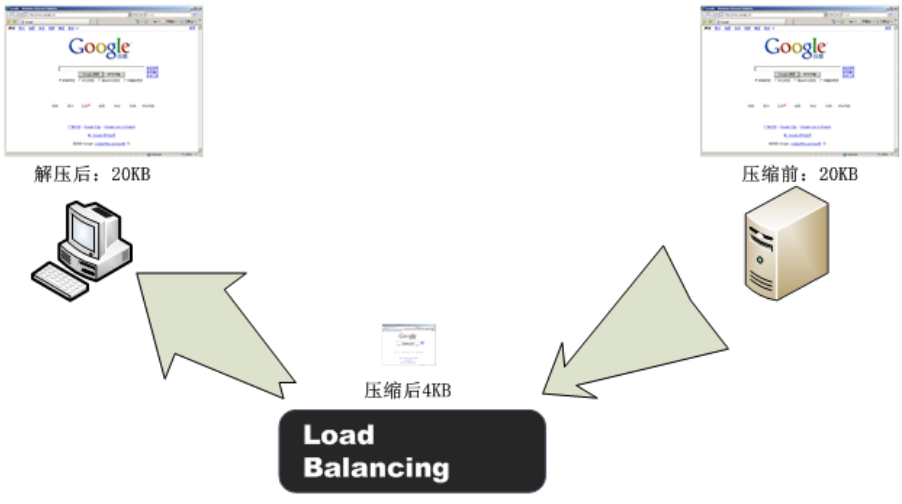
3、当负载均衡上缓存了第一个响应的数据包后，负载均衡将响应的数据包按次序返回给客户端，此时，响应的速度将依赖于负载均衡与客户端之间的链路质量；

4、当响应内容数据包依次传送给客户端并收到客户端的ACK确认请求后，负载均衡将缓冲区资源释放出来为其它TCP连接使用。

TCP缓冲技术是L7应用负载均衡的核心，它将服务器与客户端之间的TCP连接分成两个独立的TCP连接，并分别进行处理，以适应两边不同的网络环境。此外，通过TCP缓冲技术，将客户端的HTTP请求完整的接收下来并进行分析，还可以提供一些高级负载均衡的应用功能，如：URL-Hashing，URL交换（URL-Switching），基于Cookie或会话的连接保持（Cookie/Session Persistence）等等。

2.4. HTTP压缩（HTTP Compression）

HTTP协议在v 1.1中新增了压缩功能，如果客户端浏览器和服务器都支持压缩功能的话，通过客户端和服务器进行协商，对客户端的响应请求进行压缩处理。大幅节省内容传输时所需要的带宽，并加快客户端的响应速度。但是，压缩算法本身需要耗费大量的CPU资源，因此，负载均衡设备通过对HTTP压缩功能进行支持，减轻Web服务器的资源耗费，提高其处理效率。另外，由于负载均衡一般都采用硬件的方式进行压缩，因此，压缩的效率更高。此外，对于一些不支持HTTP压缩功能的老版本的Web服务器，通过启用负载均衡上的压缩功能，可以实现对系统的优化和加速。



如上图所示，在负载均衡上实现HTTP压缩功能的流程如下：

1、客户端与负载均衡建立TCP连接后，发送HTTP请求（如Get请求），客户端会将自身浏览器所支持的功能和配置情况发送给负载均衡，如：是否支持压缩、支持的压缩算法、是否支持Keep-alive（连接保持）、连接保持的时间等；

2、负载均衡在收到HTTP请求后，会将其中的有关压缩的标记删除，然后将请求转发给服务器进行处理；

3、服务器将响应的内容转发给负载均衡；

4、负载均衡收到响应的内容后，依照与客户端之间协商的压缩算法对响应的内容进行压缩，然后将压缩后的内容发送回客户端；

5、客户端收到响应的内容后，由浏览器对网页内容进行解压缩并进行浏览。

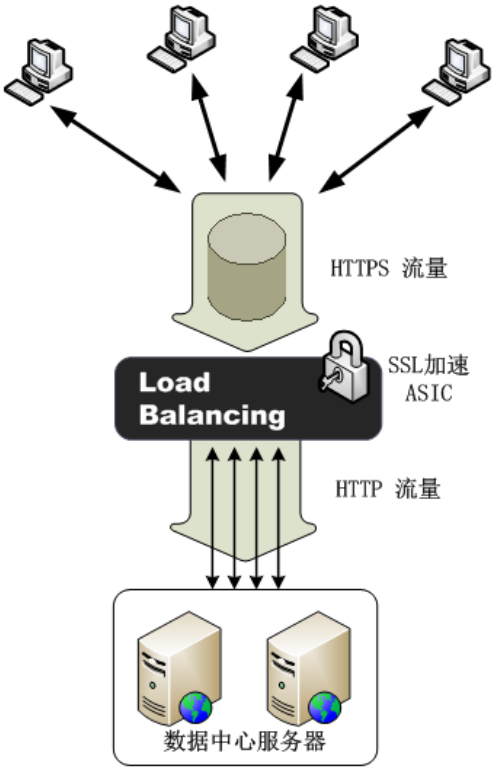
由于HTTP压缩采用的是HTTP v1.1协议中支持的标准压缩算法，因此，目前主流的浏览器（如：Internet Explorer，Firefox，Opera，Netscape等）均默认支持HTTP v1.1中的压缩功能。对于用户来说，无需修改浏览器配置也不需要安装任何插件。采用负载均衡来代替服务器做压缩，能够大幅节省服务器的资源，可以使服务器专注于应用的处理，从而提高业务处理量。另外，即使服务器不支持HTTP压缩，通过负载均衡也能实现压缩功能。

压缩能力的大小取决于被压缩对象的性质。一般来说，HTTP压缩算法对于文本格式的内容有较好的压缩效率；而对于gif等图片格式的内容，由于本身已经进行过压缩处理，压缩效率并不高。所以，需要负载均衡支持选择性压缩，即可以根据对象的类型进行选择性压缩。

2.5. SSL加速器（SSL Acceleration）

一般情况下，HTTP采用明文的方式在网络上传输，有可能被非法窃听，尤其是用于认证的口令信息等。为了避免出现这样的安全问题，一般采用SSL协议（即：HTTPS）对HTTP协议进行加密，以保证整个传输过程的安全性。在SSL通信中，首先采用非对称密钥技术交换认证信息，并交换服务器和浏览器之间用于加密数据的会话密钥，然后利用该密钥对通信过程中的信息进行加密和解密。

SSL是需要耗费大量CPU资源的一种安全技术。目前，大多数负载均衡设备均采用SSL加速芯片进行SSL信息的处理。这种方式比传统的采用服务器的SSL加密方式提供更高的SSL处理性能，从而节省大量的服务器资源，使服务器能够专注于业务请求的处理。另外，采用集中的SSL处理，还能够简化对证书的管理，减少日常管理的工作量。



图例 5 SSL加速

SSL的处理流程如下：

1、客户端发起HTTPS连接请求，协商传输的加密算法，确认双方身份，并交换会话密钥。

2、负载均衡收到客户端加密的HTTPS请求后，对请求的信息进行解密，然后通过HTTP的方式发送给后端的服务器。

3、服务器将请求的处理结果返回给负载均衡设备。

4、负载均衡设备利用会话密钥对请求的结果进行加密，然后将结果返回给客户端。

5、客户端采用会话密钥对返回结果进行解密，并显示在浏览器上。

端发送的请求以及负载均衡返回的响应均采用会话密钥进行加密，而负载均衡设备与后端服务器之间则采用HTTP的方式进行请求的发送和处理。

3. 在实际环境中应用负载均衡产品需要注意的问题

负载均衡设备提供的这些HTTP优化和加速功能，能够大大的降低客户端的响应时间，降低带宽利用率，并且能降低服务器端负载情况，节省服务器资源。但是，目前主流的负载均衡产品在同时开启多个HTTP优化和加速功能时，设备的处理性能会大大降低。尽管各个厂家均宣称自己的设备有较高的性能处理指标，并通过第三方的测评机构公布一些产品性能测试数据，但是，这些数据一般都是在一些特别设置的实验环境下测试出来的，只能作为不同厂商产品之间性能对比的依据，而不能作为用户实际应用环境的选型依据。

此外，大多数厂商都采用基本硬件平台上加装功能卡或购买许可证的方式提供这些HTTP的优化和加速功能。如果用户想在应用加速的整体解决方案中采用这些功能，需要花费更多的采购成本。对于用户来说，这些高级特性和功能如同水中之月，可望而不可及。因此，在选购负载均衡产品的时候，一定要了解自己的应用究竟需要采用哪些功能特性，而实现这些特性又需要哪些额外的费用。

A10 Networks的AX系列高级流量管理器采用目前主流的高性能多核CPU进行设计开发，通过结合四层流量处理ASIC和自主研发的ACOS系统，提供无比的性能优势，尤其是在开启多个优化和加速特性后，仍能保持较高的处理性能。2008年1月，在Tolly Group的测试中，单台AX设备实现了每秒百万的交易处理量，是截至目前唯一实现每秒百万级交易处理量的负载均衡设备。AX以其卓越的性能为客户提供最佳性价比，并提供最大的每瓦特性能，满足当今绿色计算对节能的要求。此外，AX产品采用all-in-one的销售模式，在单台硬件设备中提供所有的高级功能特性，如：内容缓存，HTTP压缩，SSL加速，IPv6，GSLB等。这些功能已经内置在AX的ACOS系统之中，不需要用户花费任何额外的费用。AX这种新的销售模式可以为用户带来巨大的利益，节省用户投资成本。

标签: Optimization, tcp/ip

aitao
关注 - 1
粉丝 - 100
+加关注

2 0

« 上一篇：实践：服务器编写/系统架构/cache
» 下一篇：为什么是三次握手

posted @ 2012-09-21 22:10 aitao 阅读(5348) 评论(0) 编辑 收藏

刷新评论 刷新页面 返回顶部

相关博文：
· http应用优化和加速说明-负载均衡
· HTTP服务负载均衡总结
· cache与负载均衡
· 测试WWW方案（反向代理，负载均衡，HTTP加速缓存）
· 使用nginx作为HTTP负载均衡

· MIT和微软的新成果，能否帮自动驾驶摆脱成长的烦恼?
» 更多新闻...