文章编号:1001 - 2486(2002)06 - 0077 - 05

代理型防火墙 Cache 策略的研究与实现*

陶　静,赵　龙
(国防科技大学计算机学院,湖南 长沙　410073)

摘　要:在代理型防火墙中设置 Cache,可以有效地减少响应延迟和降低网络带宽。对 Cache 实现中的信息过滤规则、过期页面处理和替换算法几个关键技术进行了深入研究,并提出了一种基于防火墙日志系统开销极小的 Cache 管理策略。

关键词:代理型防火墙;Cache;信息过滤;替换算法
中图分类号:TP393.08　　文献标识码:A

Research and Implementation of Cache Scheme in a Proxy Firewall

TAO Jing, ZHAO Long
(College of Computer, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: Proxy caching has been recognized as one of the effective schemes to minimize the user access latency and reduce the network traffic. This paper gives a deep research into the key techniques of a caching system, such as information filtering, stale page access and replacement algorithm, and finally presents a cache scheme based on the firewall log with little cost and a reasonable performance.

Key words: proxy firewall; Cache; information filtering; replacement algorithm

Internet 的飞速发展标志着世界经济进入了信息经济时代,同时也给自身带来了难题:地址短缺、带宽拥挤、网络安全等,采用网络代理服务系统在一定程度上能够解决这些问题。一个高效的代理服务器通常设置高速缓存(Cache),对经常访问的信息创建缓冲区,当其它用户再访问相同的信息,则直接由缓冲区中取出信息,传给用户,从而有效地减少带宽拥挤,提高访问速度。

我们在防火墙'银河卫士'的设计开发中实现了一个具有单点 Cache 代理服务器,防火墙内部用户属于同一个公司或部门,对 Web 信息有着相同的兴趣点,很有可能共享一组数据,Cache 可以更好地发挥减少响应延迟和降低网络带宽的作用。然而,防火墙在应用层进行用户身份认证、存取控制和安全报警,同时要完成许多安全防护工作,提供给 Cache 可用的硬盘大小和系统开销会受到限制。因此我们在设计中重点考虑有限 Cache 空间的利用率,同时选择高效的替换算法,减少 Cache 实现的系统开销,避免单纯追求 Cache 性能,影响代理服务器的整体性能。

围绕 Cache 存取和替换策略人们进行了大量研究,有些沿袭了传统的 LRU、FIFO 算法,缺乏考虑 Web 信息自身的特点,另一些如 LRU-k、LRU-MIN 等算法,针对 Web 特点提出,但是实现复杂,消耗过多 CPU 和内存,整体性能也差强人意。我们在分析 Web 业务访问特性的基础上,利用防火墙现有资源,提出了一种基于日志的最小代价的 Cache 替换策略。应用结果表明,这种策略实现简单、系统开销小,并且提供了很好的 Cache 命中率。

1　过滤规则

为了更好地利用有限的硬盘空间,同时考虑防火墙对信息安全的要求,我们首先依据信息自身特点,对经过防火墙的信息设计了过滤规则。以 HTTP 协议[1]代理为例,分析客户请求信息和服务器端

* 收稿日期:2002 - 06 - 13
基金项目:国家部委基金资助项目
作者简介:陶静(1971—),女,讲师,硕士。

的应答信息,只有满足以下规则,才被存入 Cache:

1) 浏览器向 Web 服务器请求信息时,主要有 GET、PUT、HEAD 和 POST 方法。PUT 和 POST 请求要向远程服务器提交数据,POST 操作还要启动 CGI 应用程序,因此它们利用缓存区的意义不大。GET 用来请求一个具体的实体,而 HEAD 只是请求实体的元信息,因为元信息较小,并且容易获得,只占据极少量的带宽,且更新变化频繁,因此也不适于缓存,所以只对 GET 方法建立的连接进行缓存。

2) GET 方法有时并不是请求一个实体,而仅仅是用来向 CGI 程序提交数据,并获得查询结果,而查询结果一般时效性较强,因此,通过 GET 发出的 CGI 的查询请求也不予缓存,而只对 HTML 文件、GIF 文件、JPEG 文件、TEXT 文件进行缓存。

3) 数据未被保护,即在 HTTP 的响应头中没有 WWW-Authorization 域,否则数据具有较高的私有性,或者是用户使用自己的私有帐号付费获得的数据,不适宜存放于公共的防火墙硬盘中。

4) HTTP 响应头中不包括 Pragma:no-Cache,Set-Cookie,Vary。

5) HTTP 请求头不包括 Authorization,Cache-control:Private 和 Pragma:no-Cache。

6)"Expires:"段中的日期大于"Date:"段中的日期,或"Expires:"段中的日期等于 0 都表示禁止使用 Cache。

7) HTTP 响应状态码 200,只有这时其所附带的数据才是可信的成功请求结果,否则不予缓存。

8) 由于并不是所有的 java applet 都是免费分发的,其中涉及知识版权,另外,单个用户下载的小程序,对于其它用户的作用并不是很大,因此不予缓存。

9) 考察服务器返回文件的大小,如果文件小于一定的阈值或大于一定的阈值,则同样不予保留。Web 中传输文档大小体分布服从对数正态分布,尾分布服从 Pareto 分布,用户比较倾向于访问较小的文档,而且并不是越小的文档访问频率越高,同时小文件的获取并不占用太多的时间和带宽,大文件消耗了过多的磁盘空间,而再次被请求的可能性小,保留价值不大。

10) 考察请求文件 URL 的深度,舍弃深度过大的文件。

2 一致性机制

Cache 进行数据缓存的一个负面效应就是会提供给客户端浏览器已经过期的信息,目前的解决策略主要有两种 强一致性机制和弱一致性机制。

强一致性机制是要确保用户可以得到最新页面信息。例如,在代理接到客户请求后,假设 Cache 内容很可能已经过期了,向服务器端转发请求,请求头添加"If-Modified-Since:Date"域,Date 值取 Cache 中相应页面内容的获取时间,如果内容未被更新,服务器端返回响应码为 304 的信息,否则返回更新后页面。这种机制必须与服务器进行一次交互,同时要对 Cache 缓存信息进行检索,实现开销大。

我们在"银河卫士"中对 Cache 的管理采用弱一致性机制,给 Cache 页面文件定制一个生存周期 TTL(time-to-live)。研究表明 Web 文件的生存周期符合双峰分布[2],也就是说如果文件很久未被改变,我们可以假设它最近仍不会被改变。用文件获取时间减去响应头中"Last-Modified"域所含的时间计算文件的"年龄",按某一比例(例如 50%)乘以文件"年龄"为该文件的 TTL 值。例如今天获取的文件在 10 天前被最后一次修改,那么它的 TTL 值为 5 天,在生存周期内的文件是有效文件,过期(5 天后)则应该重新向服务器请求,这种策略实现相对简单。当然弱一致性机制不能完全避免页面过期,在发生错误时,用户可以采用浏览器的"刷新"功能,拒绝任何 Cache 缓存,直接请求服务器的信息。

3 替换策略

替换算法的研究是改善代理型防火墙 Cache 性能的关键技术,近年来人们在这方面做了大量工作,归纳目前流行的算法,主要可以分为三类:

1) 传统的替换算法。LRU:最先移出最近最少被使用的文档,LFU:最先移出最少使用的文档,FIFO:先进先出算法等;

2) 基于某一个关键值的替换算法,例如 Size 算法[3] 页面文档的大小决定替换最大的文档,LLF

(Lowest Latency First)[4]为降低平均延迟时间,总是移出下载延时最小的文档;

3) 基于代价的替换算法,例如 SLRU(Size-adjusted LRU)[5]、LCN-R(Least Normalized Cost Replacement)[6],综合考虑若干关键因素,提出一个代价函数,计算每个文档的代价值,移出代价最大的文档。

算法的性能优劣与应用环境下的 Web 信息分布和存取规律关系密切,因此很难断言哪一种实现具有绝对的优势。像目前流行的 Squid 等大型 Cache 软件中实现了相当复杂的替换算法,性能很好,但也消耗了可观的系统资源。我们在分析了 WWW 业务访问特性[7]的基础上,针对代理型防火墙的应用背景,提出了一种基于日志的最小代价的 Cache 替换算法。算法的突出特点是利用防火墙原有资源——运行日志,来预测哪些页面最有可能被再访问,与 LFU 和 LLF 算法相比,Cache 命中率相当,但实现简单,同时系统维护 Cache 的开销大大降低了。

防火墙日志记录防火墙系统的运行信息,对于安全预警、流量统计等有着至关重要的意义,是防火墙设计中必须具备的基本功能。日志中通常包括下载文档的 URL、大小、最近修改时间和下载开销等信息。我们在代理日志中加入对 Cache 管理的内容,对未被过滤掉的符合缓存条件的访问信息做标记。

研究表明,WWW 业务访问具有局部性的特点,我们可以认为今后一段时间的业务访问和前一段时间相似,曾经被访问的文档最有可能被再次访问,也就是 Cache 应该缓存的页面文档,而曾被访问文档的信息已经记录在日志中。每隔时间 T(例如取 T 为 24 小时),读取前 T 时间(24 小时)运行日志的记录,选择日志中记录的做了 Cache 标记的访问信息,然后按本文第三部分介绍的算法计算它们的 TTL 值,已经过期的页面文件也不必缓存,取消它的 Cache 标记。

由于 Cache 可用硬盘大小的限制,不可能对所有符合条件的页面文件进行缓存,还需要计算各个页面的权重,对页面文件缓存的代价进行排序。这里我们引用文献[7]中的研究结果,以文件大小和访问频率作为计算代价的决定因素,统计日志中某一文件的被访问次数 Count,提取文件大小 Size,文件的权重 Cost 等于 Size/Count,将文件按 Cost 值从小到大进行排序,将位于前面的 Size 之和等于 Cache 大小的页面文件记录下来,建立一个表,作为 Cache 内容的索引。

最后的工作是对硬盘 Cache 中的文件内容作调整。搜索 Cache 目前缓存的文件,有些在索引表中,则保留;不在,则删除。对于那些目前未包含在 Cache 中的页面文件,可以在搜索的同时向 Web 服务器请求,得到的页面内容存入 Cache。

由于利用防火墙的原有资源,记录日志的开销可以忽略不计。计算页面代价的过程耗时较多,但每 T 时间内只计算一次,Cache 调整也只进行一次,将调整时间设置在防火墙服务空闲的时段,与业务服务的冲突很小。目前流行的其它算法在实现中都需要动态调整 Cache 内容,维护一个动态的索引表,记录当前时刻的 Cache 映像,并且每当 Cache 饱和时,页面的换入、换出引起大量磁盘 I/O,越是业务繁忙的时刻,对 Cache 的维护操作也就越频繁,相比之下我们的算法维护的是一个静态的索引表,运行中只对它进行查询工作,性能表现相对与系统开销具有很大的优势。

4　Cache 模块的实现

在'银河卫士'防火墙的设计中,我们在 Windows 2000 操作系统上使用 Delphi 作为开发平台,代理在指定的端口监听客户请求,每接到一个客户请求,就派生出一个独立的线程完成服务,处理该请求的存取控制以及转发操作。下面给出 Cache 策略实现中几个关键的流程。

4.1　从 Cache 中读数据

在 HTTP 协议代理中,每个服务线程接收到客户请求后,首先按协议格式分解客户请求头的 URL、请求方法、Cache-control 和 Pragma 等信息,对照过滤规则的 1、2、5、6 条,对满足 Cache 条件的 URL 查找索引表,在索引中,则从 Cache 中读取对应的数据文件,返回给客户。

4.2　从服务器端接受数据

HTTP 服务线程将不满足 Cache 条件或不包含在索引中的请求直接转发到 Web 服务器,接收到服

务器的响应数据后,分解响应头,按过滤规则判断每个文件是否满足缓存条件,计算 TTL 值,为方便比较,我们直接将 TTL 值记取为页面文件过期的日期值,缓存代价 Cost 取 0,与页面文件其它信息一起写入日志。

4.3 向运行日志中写数据

我们设计独立的线程完成写日志的动作,每次成功完成客户请求,都将 URL、文档大小、最后修改时间、请求耗时和是否 Cache 等信息写入日志,如果是从 Cache 中读取的数据请求耗时用 0 标记,为方便日志的管理,我们将日志信息存入 Access 数据库。日志表的结构设计为:

日志时间	URL	用户名	客户 IP	服务器 IP	文件大小	过期时间	修改时间	下载时间	Cache 标记	Cache 代价
Date	Text	Text	Text	Text	int	Date	Date	int	text	int

4.4 Cache 页面替换

代理型防火墙通常提供 24×7 的不间断服务,而系统业务总有相对空闲的时段,我们提供防火墙管理员设定 Cache 整理的时机,如果防火墙系统是为有严格上下班制度的公司服务,将整理时间安排在临上班前的一段时间,效果会更为理想。借助 Access 数据库,可以很方便地完成日志信息统计、计算权重和按权重排序等一系列工作,修改 Cache 索引表,将不包含在索引中的页面文件从 Cache 中删除,然后向 Web 服务器请求那些包含在索引中,但目前不在 Cache 中的页面文件。Windows 2000 不支持文件名中包含特殊字符,我们利用 MD5 函数从页面 URL 属性转换出 Cache 中的页面文件名,并将对应关系记录在索引表中。考虑到一些新闻性的或每日更新的数据较多,一般将页面 TTL 与修改时间差的比率设计为 50%。索引表包含页面 URL、对应文件名和过期时间三项内容,具体实现流程见图 1。

5 性能分析

在对照实验中我们实现了另两种替换算法 LRU 和 Size,对 Cache 命中率进行测试,测试中按以下假设模拟用户请求:(1)用户请求服从信息包队列模型(packet train)[8],表示为三元组(l,m,n),其中随机数 l 表示用户访问对象,随机数 m 表示对象的大小,随机数 n 表示对象的传输速率。(2)对于某一特定的用户群体,其访问对象是有限的,服从正态分布,其中 $3\times\delta$ 为用户群体的访问范围。(3)Web 中传输文档大小体分布服从对数正态分布,尾分布服从 Pareto 分布。(4)Cache 串行响应用户请求,Cache 大小配置为 128MB。

测试结果表明 LRU、Size 和最小代价这三种替换策略在 Cache 命中率方面表现相当,分别为 26.4%、30.3%和 27.7%,而另两种算法在维护 Cache 时所占用的系统开销却相对大得多。

6 结束语

本文针对代理型防火墙系统的应用特点,利用防火墙的日志资源,结合信息过滤、一致性处理等技术实现了一个开销极小的单点 Cache 系统,在非大型专用的 Cache 环境中系统资源受限的情况下,具有很大的实用价值。算法还可以略加改进,应用于其它场合。适当加大一点开销,可以改变 Cache 整理时对尚未包含但应该包含在 Cache 中的页面处理,在用户第一次请求该页面时传送请求,同时将页面存入 Cache,这样不需要额外的请求过程,但有可能浪费部分 Cache 空间 还可以考虑在统计日志时增加对服务器一级的统计,对热点服务器的页面加大 Cache 力度,这些我们将在后续的工作中进一步研究。

参考文献:

[1] Fielding R, Gettys J, Mogul J, Frystyk H, Berners Lee T. RFC 2068 Hypertext Transfer Protocol—HTTP/1.1[S]. January 1997.

[2] Cate V, Alex. A Global File System[C]. Proceedings of the 1992 USENIX File System Workshop, 1992:1 – 12.

[3] Williams S, Abrams M, Standridge C R, et al. Removal Policies in Network Caches for World Wide Web Documents[C]. Proceedings of Sigcomm '96.

[4] Wooster R P, Abrams M, Proxy Caching that Estimates Page Load Delays[C]. Proceedings of the 6th International WWW Conference,

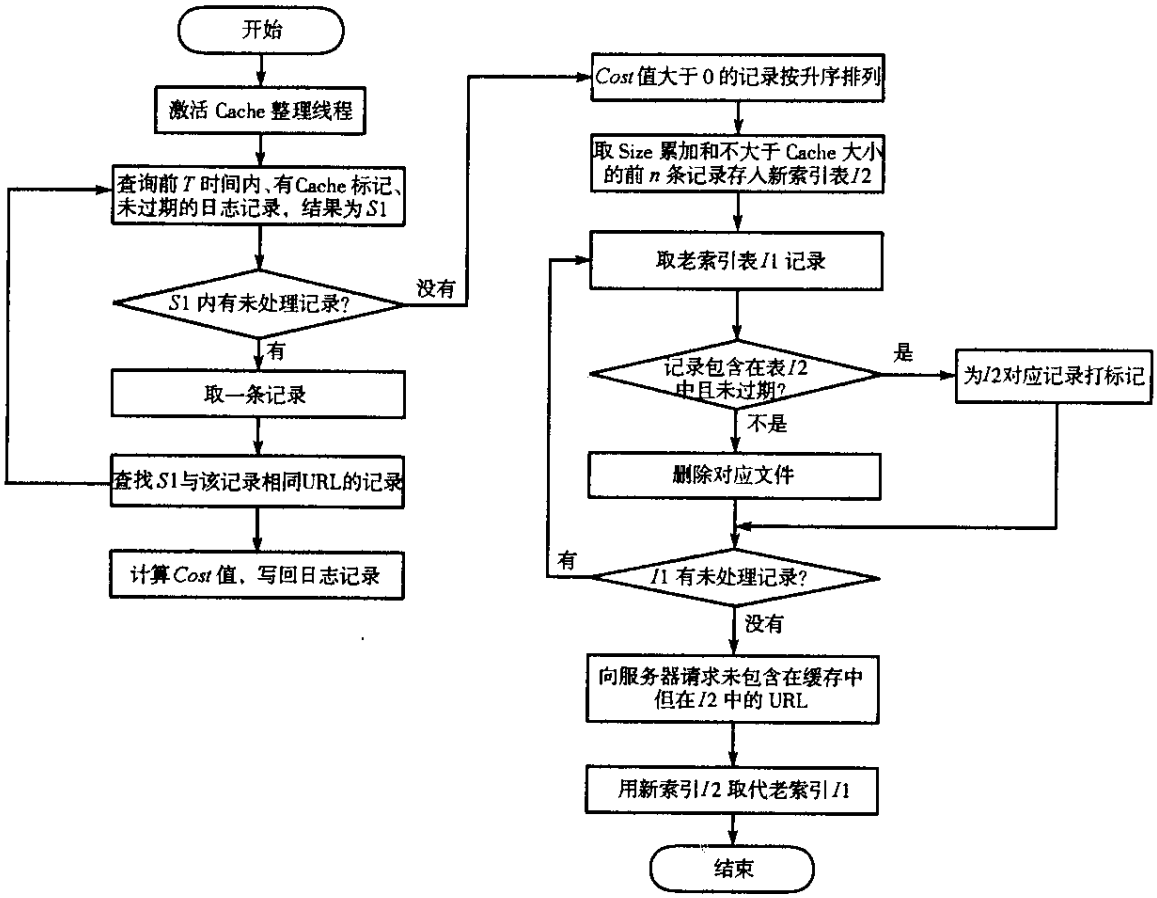

图 1　Cache 页面替换实现流程图

Fig.1　Implementation of cache page replacement

April 1997 (http://www.cs.vt.edu/ chitra/docs/www6r/).

[5] Aggarwal C, Wolf J L, Yu P S. Caching on the World Wide Web[J]. IEEE Transactions on Knowledge and Data Engineering, 1999,11(1).

[6] Scheuermann P, Shim J, Vingralek R. A Case for Delay-conscious Caching of Web Documents[C]. Proceedings of the 6th International WWW Conference, Santa Clara, April,1997.

[7] 郝沁汾等. WWW 业务访问特性分布研究[J]. 计算机研究与发展,2001(10).

[8] 陈良洲. 公共 Cache 在 WWW 中的应用[J]. 计算机工程与应用,1997(7).