

基于 HTTP 协议的大规模软件发布技术

张晓峰

(清华大学 计算机科学与技术系, 北京 100084)

(zhang_xiaofeng@hotmail.com)

摘 要: 软件发布技术是企业管理软件的核心技术, 而大规模软件发布技术的核心在于合理的体系结构和有效的带宽控制技术。通过 Web Service 体系、断点续传和带宽控制技术的综合应用, 实现了易于部署, 可以实现动态带宽控制的大规模软件发布系统。

关键词: 带宽控制; 断点; 软件分发; 评估窗口

中图分类号: TP393.04 **文献标识码:** A

Large scale software distribution technology based on HTTP protocol

ZHANG Xiao-feng

(Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

Abstract: Software distribution is the key technology for many enterprise management systems. A well designed architecture and effective bandwidth throttling technology is the core of large scale software distribution systems. Large scale software distribution technology using Web service architecture, bandwidth throttling and checkpoint were discussed.

Key words: bandwidth throttling; checkpoint; software distribution; rating window

0 引言

大规模软件发布技术是企业 IT 管理人员经常使用的技术。例如在操作系统提供商发布了某些关键补丁(Patch)后, IT 管理人员经过充分测试, 需要在较短时间内将 Patch 发布到企业内的各个计算机中。这时将出现大量的计算机同时访问 Patch 内容服务器, 会大量占用网络带宽资源, 造成网络拥堵, 影响业务系统的运行。同时, 为了保证能够同时为大量的客户端提供下载服务, 需要高性能的内容服务器, 增加了系统的实施成本。一般的配置管理工具软件^[1~3], 如 Altiris 的 Client Management Suite, Microsoft 的 SMS, CA 的 Unicenter^[1~3] 都提供了软件分发的功能。在 Windows 环境中, 这些软件都依赖于两种下载方式: 文件共享和 HTTP 下载。

文件共享方式下载

文件共享方式是基于 NETBIOS 协议的下载方式, 现在实际使用的是 NBT(NetBIOS over TCP/IP)^[4]。虽然现在使用文件共享方式来分发软件依然非常普遍, 但这种下载方式存在几个明显缺陷:

1) 从部署的角度讲, 需要复杂的权限配置。Windows 环境中, 为了共享某一目录, 必须指定对于该目录的访问权限。为了方便部署, 很多系统都采取了“Null Session”的方式。虽然可以利用各种工具来解决 Null Session 的问题^[6], 但 Null Session 给软件分发的配置带来了很大的困难, 而且 Null Session 是 Windows 系统的重大安全漏洞^[7], 黑客可以利用 Null Session 来获取系统的用户、密码及安全策略等信息, 一般在要求比较严格的企业环境内是限制使用的。

2) 下载方式的控制。在文件共享下载时, 缺乏必要的流量控制和断点续传的能力。在企业环境中分发软件时, 为了防止网络的阻塞, 通常需要一定的流量控制能力。文件共享

方式没有提供相关的功能, 在文件传输发生中断需要重新传输文件时, 文件共享方式也缺乏文件比较的机制。

HTTP 下载

HTTP 下载是利用 Web 服务提供的 HTTP 协议下载文件的一种方式。与传统的文件共享方式相比, HTTP 协议具有明显的优势。HTTP 协议设计的目的就是供多用户下载使用的, 用户管理在不需要特殊认证时, 无需作任何配置, 使用系统的默认配置即可满足软件分发的要求。HTTP 协议提供了断点续传的能力。HTTP 协议是 Internet 上广泛使用的协议, 在企业网络系统中, 不需要作任何特殊的配置就可以使用, 系统的部署非常简单^[8]。

1 基于 HTTP 的大规模软件发布的系统构架

1.1 传统软件发布技术系统构架

传统发布技术中, 一般采取软件分发点的方式, IT 管理人员需要手工建立软件分发点, 并指定各客户端下载软件使用的软件分发点, 如图 1 所示。

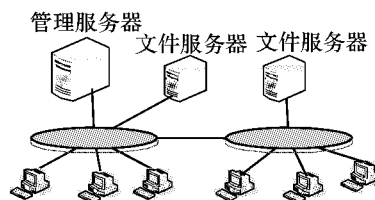


图 1 传统的软件分发体系结构

图 1 中, 要求 IT 管理人员先手工指定各客户端能够访问的文件服务器, 然后启动管理服务器上的管理分发软件, 向各文件分发点作必要的配置, 再下发发布命令, 整个操作过程非常繁琐。

1.2 基于 HTTP 的大规模软件发布系统架构

为了提高大规模软件发布的效率,可以将图1的传统架构作适当的修改,利用各子网内的代理作为新的文件传输点,从而提高系统的整体发布效率。

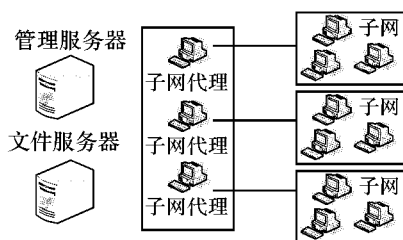


图2 基于子网代理的分发方式

图2中,管理服务器在下发软件发布命令时,会自动为该子网选取一个代理服务器。下载时,该子网代理服务器从系统的文件服务器下载相关的文件,而子网内的各客户端直接从处于本子网内的代理服务器下载软件。在这样的体系结构中,多数的软件的下载占用的带宽限制在一个网段内。

2 基于 HTTP 大规模软件发布系统的软件设计

基于 HTTP 的大规模软件发布系统的设计包括服务器端软件、客户端软件和发布下载命令的 Web Service 软件三部分。

2.1 服务器端软件的设计



图3 服务器端软件模块

与 HTTP 软件下载有关的服务器端软件包括如下三个模块:

- 1) 客户端配置信息管理保存各客户端的网络配置信息和 WWW 服务的配置信息。当安装客户端时,为客户端自动建立用于软件下载的虚拟目录。
- 2) 服务器端任务调度程序保存和管理服务器端的各种下载任务。
- 3) XML Web Service^[9] 构件 用来通过 HTTP 协议发送给用户下载命令。

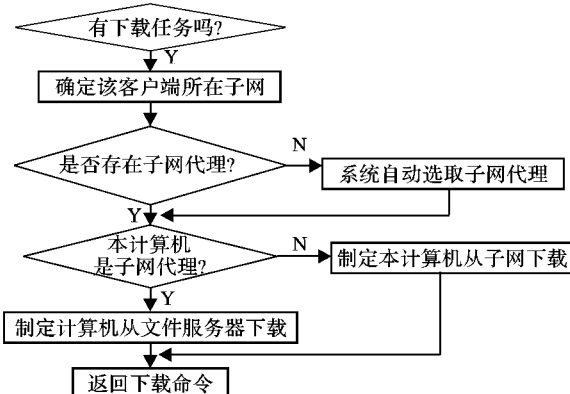


图4 子网代理选定流程图

整个体系中,客户端会定时查询服务器端是否存在下载任务。在接收到用户请求后,任务调度程序会查询当前的任

务。如果存在下载任务,任务调度程序计算该计算机所在的网段,并确定该计算机是否能够作为所在网段的子网代理。当有多个可作为子网代理的请求时,根据 IT 管理人员所制定的策略来确定子网代理。确定子网代理后,如果本计算机是子网代理,设定该计算机直接从文件服务器下载文件;如果是非子网代理,指定该计算机通过所在的子网代理下载文件。

2.2 客户端软件的设计

客户端软件主要有两个模块:系统调度模块和 HTTP 下载模块。系统调度模块管理从服务器端的 Web Service 获取下载命令,并根据下载命令使用适当的参数启动 HTTP 下载,HTTP 下载模块用来从文件服务器或子网代理下载软件。

2.2.1 系统调度模块

基于子网代理的分发技术是 Web Service 构架的,所有的命令通过基于 HTTP 协议的 SOAP 协议传输。目前几乎所有企业网络系统对 HTTP 协议都开放,而且这种构架要求服务器端的服务能主动访问客户端,系统部署很方便。为了达到和客户机/服务器结构类似的效果,要求系统调度模块在一定的时间间隔内访问 Web Service,取得本客户端当前的任务。系统调度模块的另一个功能是在适当的时刻启动 HTTP 下载。在基于子网代理的下载方式中,子网代理所在的客户端会在接收到下载任务后马上启动 HTTP 下载,其他的客户端需要在一定时间后从子网代理下载软件。服务器端在执行分发任务时,会根据分发软件包的大小和系统的带宽情况,计算子网代理下载所需的大概时间,系统调度程序在这一时间间隔后启动 HTTP 下载。

2.2.2 HTTP 下载模块

HTTP 下载模块完成软件的下载任务。子网代理从文件服务器下载文件,子网内的其他客户端从子网代理下载文件。服务器端在发布命令时,给出相关的 URL,HTTP 下载模块利用 HTTP Get 命令来获取该文件。为了保证传输的效率,HTTP 下载模块支持断点续传和带宽控制以保证传输的效率。当客户端由于网络故障或其他原因造成文件传输中断而重新下载文件时,可以利用断点续传下载还未下载的文件部分,避免重新下载全部文件,从而提高传输效率。HTTP 协议可以使用 Conditional Get 来实现断点续传。在 HTTP 协议中,与断点续传有关的是 byte range 和 ETag^[8]。其中,byte range 是在 HTTP 协议头中指定所需获取的 URL 的字节范围的部分,具体的格式如下:

Range: bytes = 起始字节-终止字节,起始字节-终止字节

协议中,用“-”表示从文件的尾部计算。例如 bytes = 100-200, -500 表示传输文件的第 100 到 200 个字节和最后的 500 个字节。

ETag(Entity Tag) 是为 HTTP 服务器中的文件创建的唯一标记。当文件的内容发生变化时,HTTP 服务器会为该文件创建一个新的 ETag。在 HTTP/1.1 的协议头中通过 If-Range: ETag 和 range 配合达到断点续传的目的。如果在 If-Range 中给定的 ETag 与该服务器文件的 ETag 一致(标志着这个文件没有被修改过),HTTP 服务器会返回 206 回应,并传回在 Range 中规定的字节范围的内容;如果 ETag 不匹配,表示在服务器端的文件发生修改,HTTP 服务器会返回 200 回应,并返回整个文件的内容。

在企业环境中分发 Patch 时,为了保证不影响业务系统的正常运行,需要对软件下载所使用的带宽进行控制。这里的带宽控制和在网络协议栈中所提到的带宽控制有一定的差

别。网络协议栈中的带宽是针对特定协议在特定的网络协议层,为了防止网络堵塞所实施的控制,关于这一方面的研究相当多,如 Leaky bucket^[10,11]。在软件分发时,需要的带宽控制是网络应用层的带宽。本方法采取了一种简单的“评估窗口”^[12]的技术来实现带宽控制。

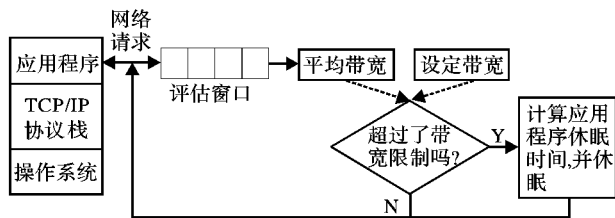


图5 使用“评估窗口”进行带宽控制

图5表示了网络应用层的带宽控制流程。在数据传输过程中,对每个窗口要计算评估窗口内的带宽,判断它是否超过带宽阈值,如超过带宽阈值,应用程序休眠一定的时间从而避免出现网络拥堵。

设 B_w 是在评估窗口内的字节总数, T_w 是评估窗口的总的时间, R_i 是实际阈值带宽。在开始限制带宽时,计算评估窗口内的带宽。如果:

$$\frac{B_w}{T_w} > R_i$$

表示实际使用的带宽符合要求,否则,计算该应用程序需要休眠的时间为:

$$T_w - \frac{B_w}{R_i}$$

休眠时间的选择需要考虑网络的实际情况,过短的休眠时间对于带宽控制的影响不大,太大的休眠时间会使得整个带宽控制不均衡。在实际使用带宽控制时,需要设定上下限阈值,当休眠时间小于阈值时,应用程序不休眠。当休眠的时间超过阈值时,在下一个窗口评估时,综合上一次休眠的时间和本次的评估结果计算需要的休眠时间。

$$(T_{w-1} - \frac{B_{w-1}}{R_{i-1}}) \times \delta + (T_w - \frac{B_w}{R_i}) \quad (0 \leq \delta \leq 1)$$

δ 要根据不同网络的特点做一定数据传输试验来选取。图6,图7是在企业实际工作环境中的带宽工作的实际数据(δ 值为0.2)。

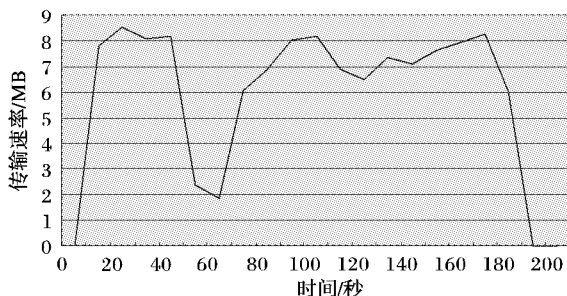


图6 无带宽限制时的网络流量图

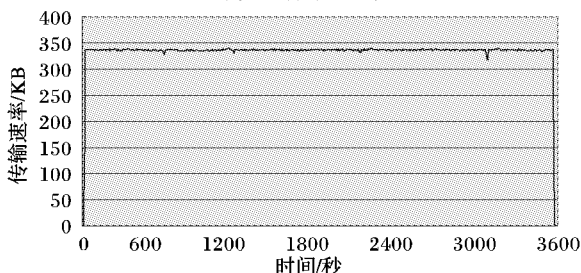


图7 限制带宽为400K时的网络流量图

图6和图7是在一个100M内部网内测试的网络流量图,其中的网络流量是点对点(发送端和接受端)的网络流量。当设定带宽为400K时,实际的网络带宽为340K,这里的误差主要来源于Windows操作系统中的定时器。由图看出,“评估窗口”带宽控制的结果还是比较理想的,如果希望能够得到更加精确的带宽控制,可以参考文献[11]中的软件计数器的方式。

3 大规模软件分发技术的应用

基于HTTP的大规模软件分发技术的实现方式可以避免传统的基于客户机/服务器方式软件分发技术部署的复杂性。IT管理人员可以使用比较廉价的服务器,通过客户端的带宽控制和断点续传,在不中断业务系统的情况下,达到在企业范围内大规模分发软件的目的。基于HTTP软件分发技术,可以发展新的廉价的,易于使用和部署的企业配置管理软件和文件复制工具,也可以在传统的管理软件中,加入基于HTTP的软件分发技术来改善软件分发的效果。

基于HTTP的软件分发技术为了方便系统的部署,使用了公共的HTTP协议端口,但带来的问题是服务器端无法主动对客户端发送命令,造成系统有一定的延时,客户端调度程序在运行过程中的查询工作方式也增加了网络系统的带宽使用。如果在使用时需要主动发送命令给客户端,可以利用传统的客户机/服务器的方式在客户端增加看护进程,但由于看护进程需使用系统专有的网络端口,在一定程度增加了系统在企业网络内部署的困难。

参考文献:

- [1] Altiris, Inc. Client Management Suite[EB/OL]. <http://www.altiris.com/products/clientmgmt/>, 2004.
- [2] Computer Associates International, Inc. Unicenter Enterprise Management[EB/OL]. <http://www3.ca.com/Solutions/Solution.asp?id=315>, 2004.
- [3] Microsoft, Inc. Software Update Services Deployment White Paper[Z], 2004.
- [4] W3C. RFC 1001[EB/OL]. <http://www.w3.org>, 2004.
- [5] Microsoft, Inc. Null session[EB/OL]. <http://support.microsoft.com/kb/q132679/>, 2004.
- [6] Microsoft, Inc. 如何在基于 Windows 2000 的计算机上启用空会话共享[EB/OL]. <http://support.microsoft.com/kb/q289655/#Task1>, 2004.
- [7] SANS Institute. Top Vulnerabilities to Windows Systems[EB/OL]. <http://www.sans.org/top20/#w3>, 2004.
- [8] W3C. RFC 2616, Hypertext Transfer Protocol[S/OL]. <http://www.w3.org/Protocols/rfc2616/rfc2616.html>, 2004.
- [9] W3C. Web Services Architecture[EB/OL]. <http://www.w3.org/tr/ws-arch>, 2004.
- [10] COMER D, STEVENS D. Internetworking with TCP/IP[M]. Prentice Hall, 1996.
- [11] ARON M, DURSHEL P. Soft Timers: efficient microsecond software timer support for network processing[Z]. 17th ACM Symposium on Operating Systems Principles[C], 1999.
- [12] RYN KD, HOLLINGSWORTH JK, KELEHER PJ. Efficient Network and I/O Throttling for Fine-Grain Cycle Stealing[A]. Proceedings of Supercomputing[C], 2001.