第 24 卷第 1 期　　南 京 气 象 学 院 学 报　　Vol. 24 No. 1
2001 年 3 月　　Journal of Nanjing Institute of Meteorology　　Mar. 2001

文章编号: 1000-2022(2001)01-0148-06

一种高效、容错的性能管理工具

汤莉莉[1)]　童　玮[2)]　刘海建[2)]

(1) 南京气象学院环境科学系, 南京　210044; 2) 解放军理工大学通信工程学院, 南京　210016)

摘要: 性能管理是网络管理中的重要组成部分, 本文从性能监视、性能统计和故障统计三个方面分别介绍了实现一个性能管理工具的一般步骤, 特别是对于如何减小计算误差提出了自己的方法, 最后介绍了实现一个完整的性能管理工具需要进一步做的工作。

关　键　词: 性能管理; 简单网络管理协议; 计算机网络

中图分类号: TP393　　文献标识码: A

随着信息时代的到来, 作为信息传播与共享工具的计算机网络得到了飞速发展, 网络中采用的先进技术越来越多, 组成越来越丰富, 规模也越来越大。与此同时, 网络的维护和管理工作也日趋复杂。为了对网络进行集中有效的控制和管理, 开放的基于标准的网络管理框架也就应运而生, 现在通常使用的有两个重要网络管理框架: 基于 TCP/IP 的 SNMP(simple network management protocol)和基于 OSI 的 CMIP(common management information protocol)。无论是哪一种管理框架其所要实现的功能都是相同的, 国际标准化组织(ISO)定义了网络管理的五大功能域: 故障管理(FM)、配置管理(CM)、安全管理(SM)、性能管理(PM)和计费管理(AM)。其中性能管理考虑的是网络的利用情况, 也就是网络运行的好坏。性能管理的最大作用是帮助网络管理者减少网络的拥塞现象, 从而为用户提供一个水平稳定的服务。利用性能管理工具, 管理者可以实时监控设备和连接的使用情况, 利用性能管理工具中的统计结果可以了解某个端口在一段时间内的运行情况, 并且根据结果进一步作出决策[1,2]。

本文介绍了一个高效、容错的性能管理工具, 主要包括性能监测、性能统计和故障统计三个部分。

1　系统结构

计算机网络管理和安全系统的核心是一个基于 SNMP 的网络管理平台, 如图 1 所示, 性能管理是网络管理平台上的一个基础管理工具, 性能监测工具收集管理端口的数据写入数据库, 同时可以实时监视某个端口的运行情况。事件分析和告警处理模块通过两种方式来获得设备或端口的状态和具体告警内容, 一种是接收设备发来的 Trap, 另一种是通过定时轮询。获得的结果一方面送给拓扑与配置管理模块来实时显示, 另一方面写入数据库。管理者根据数据库中的数据来统计端口的故障和性能, 根据结果来检查网络趋势, 使用关于趋势的数据可以预测

收稿日期: 2000-08-30; 修订日期: 2000-11-05

作者简介: 汤莉莉, 女, 1973 年 5 月生, 学士, 助教

网络运行的峰值, 从而避免网络饱和及不可通行带来的低性能。

2　性能监视

性能监视模块首先查询路由器、以太网交换机等网络设备各个接口的 MIB 变量 ifSpeed, ifSpeed 的值标识了接口实际的带宽。需要统计的大多数 MIB 变量都是用 32 个二进制位表示的 Counter 类型, Counter 类型的变量溢出后又从 0 开始计数(即值最大可达到 2^{32}) 。所以为了不漏掉计数, 必须最多间隔($2^{32}\times 8$) / ifSpeed 秒查询 Counter 变量一次。例如, 对于一个 10 M 的接口查询间隔设置为 5 min。当然接口不可能一直满负荷地传输数据, 实际设置的查询间隔可以更长一些, 保证既不会丢失数据, 又尽可能地减少带宽的占用。

需要监测的数据有: 1) 接口接收的字节数: ifInOctets; 2) 接口接收的分组中被接受的非广播分组数: ifInUcastPkts; 3) 接口接收的分组中被接受的广播和多播分组数: ifInNUcastPkts; 4) 接口接收的分组中被丢弃的分组数: ifInDiscards+ ifInErrors+ ifInUnknownProtos; 5) 接口发送的字节数: ifOutOctets; 6) 接口发送的非广播分组数: ifOutUcastPkts; 7) 接口发送的广播和多播分组数: ifOutNUcastPkts; 8) 接口发送的被丢弃的分组数: ifOutDiscards+ ifOutErrors; 9) 设备接收的 IP 分组数: ipInReceives; 10) 设备请求发送的 IP 分组数: ipOutRequests; 11) 设备接收的 TCP 报文段: tcpInSegs; 12) 设备请求发送的 TCP 报文段: tcpOutSegs。

由于 SNMP 使用不可靠的无连接的 UDP 作为传输层协议, 因此性能监测工具发出的 SNMP 查询报文有可能得不到响应, 性能监测工具就认为是设备的相应接口出现了某种故障(数据库中记为 "- 1") , 这就导致了监测数据的不可靠性。

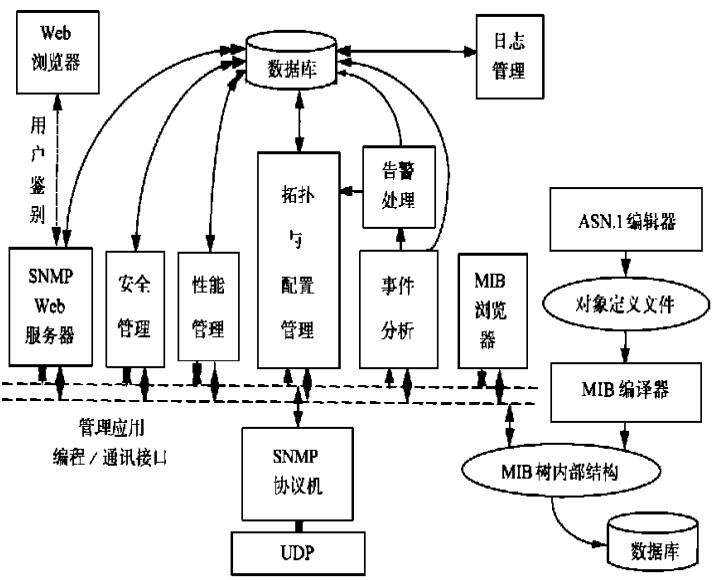

图 1　计算机网络管理和安全系统模块组成

Fig. 1　Composition of computer network management and savity system

3　性能统计

性能统计模块统计一段时间内一个或多个设备以及设备接口的流量、出错分组数、利用率和差错率。统计单位可以按年、月、日或小时进行设置, 统计结果可以采用表格、柱状图和曲线

图的方式显示, 也可以进行报表打印。

性能统计的 4 种数据类型定义如下。

流量= 接口接收的字节数+ 接口发送的字节数=
ifInOctets+ ifOutOctets;

可用率= 流量/接口速率=
(接口接收的字节数+ 接口发送的字节数) / 接口速率=
(ifInOctets+ ifOutOctets) · 8/ifSpeed;

出错分组数= 接口接收的分组中被丢弃的分组数+ 接口发送的被丢弃的分组数=
ifInDiscards+ ifInErrors+ ifInUnknownProtos+ ifOutDiscards+
ifOutErrors;

差错率= 出错分组数/ 总的接收和发送的分组数=
(接口接收的分组中被丢弃的分组数+ 接口发送的被丢弃的分组数)/(接口接收的分组中被接受的非广播分组数+ 接口接收的分组中被接受的广播和多播分组数+ 接口发送的非广播分组数+ 接口发送的广播和多播分组数)=
(ifInDiscards+ ifInErrors+ ifInUnknownProtos+ ifOutDiscards+
ifOutErrors)/(ifInUcastPkts+ ifInNucastPkts+ ifOutUcastPkts+
ifOutNUcastPkts)。

利用以上定义的 4 种数据类型可以从不同角度反映网络的运行情况, 为管理者和决策者提供了有力的保证。

在实现过程中, 假定性能监视进程在后台不间断运行, 即保证数据的连续性。由于数据库表中的记录可能存在 "- 1", 也就是查询数据的报文没有响应, 为了保证数据的可靠性和准确性, 需要分以下几种情况来讨论。

(1) 后面的有效数据比前面的有效数据大, 并且两者之间没有 "- 1", 这是正常情况。

(2) 后面的有效数据比前面的有效数据小 4 G, 并且两者中间没有 "- 1"时认为溢出, 否则跳过这次比较计算。注意这里只对前后的两条记录进行比较。

(3) 两条有效记录之间出现了 "- 1", 这时又需要分 3 个方面统计:

1) 后面的数据比前面的数据小, 这时认为是故障;

2) 后面的数据比前面的数据大, 两者中间只有一个 "- 1", 这时就认为是 SNMP 查询报文或响应报文丢失而不认为是故障;

3) 后面的数据比前面的数据大, 两者中间有多个 "- 1", 这时认为出现了故障。

当认为出现故障时, 在计算中就认为后面一条数据 MIB 变量清 0 后从 0 开始计数。

这里的规则是在分析大量数据的基础上得出的, 并且已经在解放军理工大学通信工程学院校园网中进行了验证。使用我们开发的网络管理平台对校园网中的设备进行监视, 因为管理平台基于SNMP, 所以只有支持 SNMP 协议的设备才能通过性能监视模块取到数据。观察网中一台有 27 个端口的交换机, 在接收到的近 2 万条数据中, 对于出现一个 "- 1"的设备查看数据库中的 Trap 表, 结果发现在这个时间里设备没有一次出现故障, 实际都是报文丢弃。而连续出现两个以上 "- 1"的情况都是设备出现了故障。当然这只是对于特定网络中的设备进行的测试, 从理论上讲, 连续出现两个以上报文没有响应也是可能的, 只是相对于仅有一个报文没有响应概率要小得多。当出现一个 "- 1", 即只有一次查询报文超时, 也有可能是设备故障原因, 但是相对于报文超时概率要小得多。

对于上面的第三条需要特别注意, 出现这种情况是比较常见的。譬如一台有 10 M 网卡的主机, 性能监视查询间隔设置为 5 min, 在这段时间内主机有可能重新启动, 也就是计数器清 0 后重新计数。如果启动后计数器中的 MIB 变量值比启动前的值大, 按照第一条规则认为没有出现故障。但是当启动后计数器中的 MIB 变量值比启动前的值小时, 这时认为是溢出就会有很大的误差。例如: 重启动前的值为 V_1= 1 000, 而重启动后的值为 V_2= 100, 再下一次的值为 V_3= 1 000, 32 位寄存器的最大值 $MAX = 2^{32}$= 4 294 967 296, 认为中间溢出一次, 误差为

$$\delta_1 = MAX - V_1 = 4\ 294\ 967\ 296 - 1\ 000 = 4\ 294\ 966\ 296。$$

这个误差对于一个端口几乎是致命的, 因为一个 10 M 端口有时一个月的流量也达不到这个值, 不过这是最坏情况。根据规则(3), V_2 比 V_1 小 900, 这个值远比 4 G 小, 所以不认为是溢出, 跳过这次比较计算, 直接计算 V_3 与 V_2 的差值, 这样计算误差得

$$\delta_2 = V_2 + V_1 \text{ 到重启动时的增加值。}$$

显然在大部分情况下 δ_2 要比 δ_1 小得多, 规则(3)要求 V_1 比 V_2 大 4 G 以上时会大大减小误差 δ_1。这里还有一些情况需要考虑: 一是查询数据是正确的, 并且 MIB 变量值实际已经溢出一次, 但是后面的数据比前面的数据小不到 4 G, 这样按照规则(3)就要跳过这次计算, 引起计算误差, 不过这种情况几乎不会发生, 原因是一个端口不可能满负荷运行, 端口利用率很低, 另外还可以缩短查询时间间隔避免这种情况发生; 另一种就是故障后的数据值比故障前的数据值大, 这里的计算误差为

$$\delta_3 = V_2 - (V_2 - V_1) = V_1。$$

这个误差不能避免, 但是一般比较小, 可以不用考虑。

4　故障统计

故障统计模块统计一段时间内一个或多个设备以及设备接口的可用率、故障时间和故障明细表。统计单位同样可以按年、月、日或小时进行设置, 统计结果可以采用表格、柱状图和曲线图的方式显示, 并且可以进行报表打印。

故障统计的 3 种数据类型定义如下。

可用率= (总的统计时间—统计时间内的故障时间) / 总的统计时间;

故障时间表= 统计时间内的故障时间;

故障明细表: 这个表记录的是一段时间内的具体故障情况。

SNMP 定义的 Trap 域包括 7 种值。coldStart、warmStart、linkDown、linkUp、authenticationFailure、egpNeighborLoss、enterpriseSpecific。实际自定义 Trap 在数量上往往大大超过一般的 Trap。对于 Trap PDU 接收者不会作出响应, 因而一旦 Trap PDU 丢失, 那么设备就不能及时向网络管理站提供某些重要的事件。为了得到设备或接口的状况就需要定时轮询各个端口, 并把轮询结果写入数据库。轮询时间间隔可以由网络管理员选择, 时间间隔太长就不能及时得到网络的信息, 时间间隔太短会增加网络负荷。

事件分析和告警处理模块将告警写入数据库。由于采用接收 Trap 和轮询的方式, 所以会出现重复告警信息。告警处理模块在把告警写入数据库时将重复的滤掉, 最后剩下的是只有告警起始时间 StartTime、告警恢复时间 EndTime 和告警类型的一张表, 这样在统计的过程中可以大大提高效率。按照统计要求需要分以下几种情况。1) StartTime 与 EndTime 都在统计时间内; 2) StartTime 不在统计时间内, EndTime 在统计时间内; 3) StartTime 在统计时间内, EndTime 为空; 4) StartTime 不在统计时间内, EndTime 为空; 5) StartTime 在统计时间内,

EndTime 不为空也不在统计时间内; 6) StartTime 不在统计时间内, EndTime 不为空也不在统计时间内。

根据以上 6 种情况可以完成故障统计功能。

5 工具的高效性

根据上面定义的数据类型和算法, 可以对性能和故障进行统计, 产生管理员需要的图表。但是性能监视模块采集的数据在没有处理的情况下数量非常大, 例如在校园网中一个月的数据有 40 多万条, 在更大的网络或更长的时间里产生的数据将会更多。所以为了提高统计效率, 需要注意两点: 1) 尽量减少应用程序与数据库的交互; 2) 尽量将数值计算的工作交给数据库系统来完成。

网络管理员在管理工具中设置的查询条件可能很复杂, 在最初的设计中, 应用程序不断地查询数据库才能产生最终结果, 全部过程需要花费很长的时间。为了克服这个问题, 我们从 3 个方面进行了改进。1) 性能监视模块对采集的数据先进行后台处理, 形成多级表; 2) 使用一次组合查询, 先将数据在数据库中进行处理, 充分利用数据库进行数值计算的能力; 3) 优化统计算法, 减少统计时间。

经过改进, 统计效率大大提高。改进前, 统计 20 个端口一个月的流量大约需要 15 min, 改进后只需要 30 s 左右。

6 结束语

本文介绍了一个高效、容错的性能管理工具, 在这个工具中包含了故障管理的一部分内容, 实际应用中故障管理和性能管理是不可分割的, 一旦出现故障必然导致网络性能下降或者不可用。网络管理员必须同时使用这两种工具才能对网络进行有效的管理, 判断网络故障, 评价网络性能, 为网络用户提供更好的服务。已经开发的网络管理平台和一部分基础管理工具在试用过程中反应良好, 特别是基础管理工具比较适合我国目前的网络现状, 灵活、实用、可操作性强。

作为一个完整的性能管理工具, 还需要完成以下两方面工作。

(1) 设置阈值。性能管理中另一个重要方面是设置阈值, 在应用过程中可以对影响网络性能的各项指标设置阈值。例如可以对网络流量、利用率、出错率等项设置阈值。一旦阈值被设定, 当网络性能达到一个特定的值时, 性能管理工具就会向网络管理员报告。但是把阈值设置为多少是很困难的, 通常情况下是通过试验来找到一个合理的值。阈值使网络管理员在影响网络性能之前就找出问题并尽快解决它。

(2) 使用网络摸拟。使用模拟可以确定如何将网络调整到最佳性能, 但是建立网络模拟本身是一件很困难的任务, 实际上到现在为止还没有一家软件公司能模拟复杂的网络配置。网络模拟工具应能从运行中的网络上或模拟系统上取得输入数据并就用户期望的网络性能作出预测, 能够帮助分析未来性能和确定何种配置将产生最佳性能。假如有足够的数据输入, 工具还应该能够模拟网络流量并显示可能的响应时间、可用率, 从而帮助网络管理者购买合适的网络设备和分配正确的带宽[3, 4]。

参考文献:

[1] 谢希仁. 计算机网络[M]. 第二版. 大连: 大连理工大学出版社, 1996

[2] Sean Harnedy. 简单网络管理协议教程. 第二版[M]. 胡谷雨, 等译. 北京: 电子工业出版社, 1999

[3] 岑贤道, 安常青. 网络管理协议及应用开发[M]. 北京: 清华大学出版社, 1998

[4] Norman R J. Object-oriented systems analysis and dsign[M]. 北京: 清华大学出版社, 1998

A Kind of High-Efficiency and Error-Tolerance Performance Management Tool

TANG Li-li[1], TONG Wei[2], LIU Hai-jian[2]

(1. Department of Enivornmental Science, NIM, Nanjing 210044;
2. Institute of Communications Engineering, PLAUST, Nanjing 210016)

Abstract: Performance management (PM) is one of the important parts in network management (NM). This paper introduces the general process to implement a performance management tool in three aspects such as performance watch, performance management and fault statistics. Espacially, a method to reduce computing error is presented. Finally, some comments to this tool and future plan are given.

Keywords: performance management, simple network management protocol, network management