Linux集群系统性能的实时监测及其可视化研究

彭土有
(成都理工大学 信息工程学院,四川 成都 610059)

摘 要:在研究如何获取单个Linux集群服务器节点性能参数的基础上,利用CGI(Common Gateway Interface)技术,通过HTTP协议实现了异步采集Linux集群服务器节点性能参数的方法。在监控节点上利用QT编程实现Linux集群系统性能的实时监测及其可视化。通过定时器、CGI技术、HTTP协议、专用算法的有机结合,实现无序返回结果的有序化,达到实时、异步采集Linux集群服务器节点性能参数目的,是一种具有创新性的Linux集群系统性能实时监测方法。与传统的网络监测方法相比,具有占用系统资源少、可靠、实用的特点,可广泛应用于各种Linux集群系统,也为下一步实施Web服务器动态负载均衡打下基础。

关键词:CGI;Linux集群系统;HTTP协议;节点性能
中图分类号:TP31 **文献标识码**:A **文章编号**:1673-629X(2010)11-0033-04

Study on Real Time Monitoring of Linux Cluster System Performance and Its Visualization

PENG Tu-you
(Institute of Information Engineering, Chengdu University of Technology, Chengdu 610059, China)

Abstract: Based on study of single Linux cluster system node performance and its extraction methods, performance parameters of nodes are collected asynchronously by HTTP protocol and CGI(common gateway interface) technology. At last, the whole performance of Linux cluster system can be monitored on one node by coding on QT platform, which is a combination result of timer, CGI, HTTP protocol and special algorithm. It is a kind of innovational performance real-time monitoring method of Linux cluster system. It is proved that this solution is not only a reliable and practical monitor for different kinds of Linux cluster system, but also it is the basics of implementation of Web dynamic load balance based on Linux cluster system.

Key words: CGI; Linux cluster system; HTTP protocol; node performance

0 引 言

为实施Linux Web服务器[1]集群的负载均衡[2],实时、准确获取Linux集群[3]服务器节点(如下简称:节点)的CPU使用率、内存占有率、磁盘IO负载及网络负载等参数至关重要。目前,绝大部分网络设备(包括服务器)的性能监控[4]是通过简单网络管理协议(SNMP,Simple Network Management Protocol)来实现的,该方法占用网络资源多,对Linux集群系统来说实时性不足。文献[5]利用Gtk+(GIMP Toolkit)开发工具,通过客户、服务器端的Socket编程实现Linux集群系统并行应用程序监测技术;文献[6]提出了基于Client/Server模式,在并行编程环境[7,8](MPI,Message Passing Interface)中加入额外的例行程序(MPE,Multi Processing Environment)来实现的Linux集群可视化监测技术,这些方法仅适用于Linux高性能集群系统[8]。为此,笔者探讨利用CGI(Common Gateway Interface)技术,通过HTTP协议异步采集Linux集群服务器节点性能参数这种创新的思路和方式实现了完全基于开源环境的Linux集群系统性能实时监控及其可视化。其中,CGI方式提取节点性能参数,CGI程序占用节点资源少、快速、实用;同时利用定时器控制发送HTTP异步请求,并实现HTTP无序返回结果的有序化,极大地提高了监测系统实时性。目前为止,尚未发现同类实现方法的相关论述。

收稿日期:2010-03-19;**修回日期**:2010-06-11
基金项目:广东省自然科学基金(8152900006000002)
作者简介:彭土有(1964-),男,成都理工大学博士研究生,广东省江门职业技术学院电子与信息技术系副教授,研究方向为地球探测与信息技术、Linux网络及集群技术。

1 算法流程

Linux集群系统负载实时监测及其可视化系统由以下七大模块组成:

(1)CGI模块;

(2)HTTP 请求模块;

(3)定时器模块;

(4)HTTP 异步采集数据模块;

(5)HTTP 请求失败处理模块;

(6)节点性能参数集成的专用算法模块;

(7)集群性能可视化显示模块。

其中,CGI 模块主要功能是实时提取节点性能参数;HTTP 请求模块主要功能是向所有节点发送 HTTP 请求;定时器模块主要功能是生成可控制定时器,并与 HTTP 请求模块结合使用;HTTP 异步采集数据模块主要功能是根据 CGI 模块的返回,异步采集各节点性能参数数据;HTTP 请求失败处理模块主要功能是处理 CGI 模块的出错响应;节点性能参数集成的专用算法模块主要功能是,按照节点顺序集成所有节点各种实时的性能参数,为正确显示集群性能提供基础数据。集群性能可视化显示模块主要功能是以直观的柱状图方式展示集群实时性能,并提供用户对各种参数的控制或选择功能。

其算法的流程图如图 1 所示。

2 节点实时性能参数的提取方法

2.1 节点性能参数

节点的实时性能参数主要包括 CPU 使用率、内存占有率、磁盘 IO 负载[9]及网络负载。在 Linux 系统中,内存虚拟文件系统实时记录了系统的运行参数,其中:文件/proc/stat 记录了 CPU 实时性能参数,包括用户模式(user)、低优先级的用户模式(nice)、内核模式(system)以及空闲(idle)的 CPU 时间。CPU 利用率计算公式为:(user + nice + system)/(user + nice + system + idle) * 100%;/proc/meminfo 文件记录了当前内存的使用量(usedmem)以及内存总量(totalmem),可以通过执行命令 $ free - m 来读取。内存占有率计算公式为:(usedmem/totalmem) * 100%;/proc/net/dev 文件记录了从节点输出的数据包和流入节点的数据包数。/proc/diskstats 文件记录了当前磁盘 IO 负载的相关数据,可以通过执行命令 $ iostat 来读取。最后,通过执行命令 $ /sbin/ifconfig 来获取节点 IP 地址作为节点标识,用于区分不同节点的性能参数。

2.2 CGI 程序实现节点性能参数的提取

Web 服务器[10]的 CGI 程序用于实时提取节点性能参数。编写的 C 语言代码要遵循 CGI 标准,主要实现代码分述如下。

1)计算内存占有率。

```
if(system("free - m | awk '{print $2, $3}' > /tmp/mem"));
fp = fopen("/tmp/mem", "rb");
if(fp < 0) return -1;
fread(StatBuf, sizeof(StatBuf), 1, fp);
```

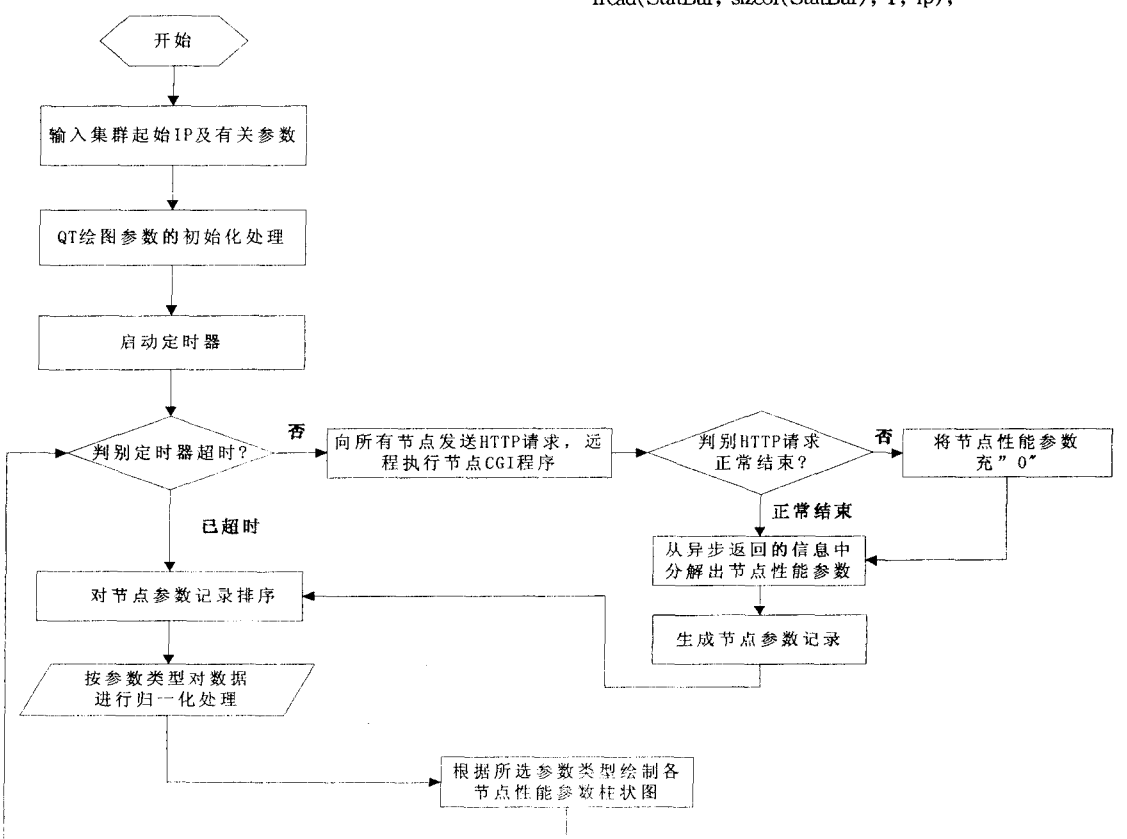

图 1 Linux 集群系统负载实时监测及其可视化算法流程图

```
line = strstr(StatBuf, "\n");
TotalMem = atoi(line);
line = strstr(line, " ");
UsedMem = atoi(line);
float memused=(float)UsedMem/TotalMem * 100.0;
if(strlen(StatBuf) > len) return -1;
fclose(fp);
```

2)计算 CPU 使用率。

```
long user, nice, system_tmp, idle;
static long total;
float scale;
char buf[1024];
if ((fp = fopen ("/proc/stat", "r")) ! = NULL){
  while (fgets (buf, sizeof(buf), fp)){
    if( * buf = = 'c' && * (buf+1) = = 'p') break;
  }
}
else return 0.0;
sscanf(buf, "cpu %ld %ld %ld %ld", &user, &nice, &system,
&idle);
total = user + nice + system + idle;
if (total < 1) total = 1;
scale = 1.0 / total;
float cpuused=(float)((user + nice + system) * scale * 100);
fclose(fp);
```

3)计算 IO 负载率。

```
if(system("iostat | awk '{print $5, $6}' >/tmp/iofile"));
fp = fopen("/tmp/iofile", "rb");
if(fp < 0) return -1;
fread(StatBuf, sizeof(StatBuf), 1, fp);
line = strstr(StatBuf, "idle\n");
line = strstr(line, "\n");
float ioload = 100.0 - atof(line);
```

4)计算网络负载率。

```
if(system("cat /proc/net/dev | awk '{print $2}' >/tmp/net-
Tfile"));
fp = fopen("/tmp/netTfile", "rb");
if(fp < 0) return -1;
fread(StatBuf, sizeof(StatBuf), 1, fp);
line = strstr(StatBuf, "bytes");
long netTransmit1,netTransmit2,netTransmit3;
sscanf ( line, " bytes% ld% ld% ld ", &netTransmit1,
&netTransmit2, &netTransmit3);
long netTransmit = netTransmit1 + netTransmit2 + netTransmit3;
fclose(fp);
if(system("cat /proc/net/dev | awk '{print $10}' >/tmp/netR-
file"));
fp = fopen("/tmp/netRfile", "rb");
if(fp < 0) return -1;
fread(StatBuf, sizeof(StatBuf), 1, fp);
line = strstr(StatBuf, "packets");
long netReceive1,netReceive2,netReceive3;
sscanf ( line, " packets% ld% ld% ld ", &netReceive1,
&netReceive2, &netReceive3);
long netReceive = netReceive1 + netReceive2 + netReceive3;
long netTotal = netReceive + netTransmit;
fclose(fp);
```

5)求取节点 IP 地址。

```
if(system("/sbin/ifconfig | awk '{print $2}' >/tmp/ipfile"));
fp = fopen("/tmp/ipfile", "rb");
if(fp < 0) return -1;
fread(StatBuf, sizeof(StatBuf), 1, fp);
line = strstr(StatBuf, "addr:");
sscanf(line, "addr:%s,%s", &ip, &iptmp);
fclose(fp);
```

6)按顺序返回内存占有率、CPU 使用率、磁盘 IO 负载率、网络发送数据负载、网络接收数据负载及 IP 地址数据。

```
printf("%f\n%f\n%f\n%ld\n%ld\n%s\n",memused,
cpuused,ioload,netTransmit,netReceive,ip);
return 1;
```

3　集群系统性能的实时监测及其可视化

3.1　QT 的网络开发功能

QT[11]是由挪威的 Trolltech 公司开发基于 C++ 语法的跨平台应用程序框架,是开源桌面系统 KDE 的基石,QT 还支持 Linux 嵌入式开发。QT 提供了一个全新的网络模块(QtNetwork)和功能强大的绘图功能。如下将通过 QT 网络模块中 QHttp 类实现异步采集节点服务器实时性能参数及其可视化功能。

3.2　HTTP 客户端的实现

运行监测系统的节点作为 HTTP 客户端,使用基于 TCP/IP 中 HTTP 协议的 QHttp 类,并启动定时器定时远程执行各节点的 CGI 程序,采用异步工作方式通过 QHttp 类的 get()函数采集节点 CGI 程序返回的实时性能参数。当节点 Web 服务器运行正常时,将获取到节点实时的性能参数;当节点 Web 服务器出现故障时,QHttp 类将返回出错代码,此时,需要通过 QHttp 的槽函数 httpRequestFinished()将节点所有性能参数充"0"。实现函数 getServerInfo()的代码如下:

```
getServerInfo()
{
QLocale qvariant;
QString hostPre="http://"+subnet;
errorSum=0;
file->seek(0);
```

```
for(int i = qvariant. toInt(hostBegin); i < = qvariant. toInt(hostEnd);i + +)
{
http[i] = new QHttp(this);
connect(http[i], SIGNAL(responseHeaderReceived(const QHttpResponseHeader &)),this, SLOT(readResponseHeader(const QHttpResponseHeader &)));
connect(http[i], SIGNAL(requestFinished(int, bool)), this, SLOT(httpRequestFinished(int, bool)));
QString urlString = hostPre + qvariant. toString(i) + "/cgi - bin/cgi";
QUrl url(urlString);
QFileInfo fileInfo(url. path());
QHttp::ConnectionMode mode = url. scheme(). toLower() == "https" ? QHttp::ConnectionModeHttps : QHttp::ConnectionModeHttp;
http[i] - >setHost(url. host(), mode, url. port() == -1 ? 0 : url. port());
QByteArray path = QUrl::toPercentEncoding(url. path(), "!$&'()* +,;=:@/");
if (path. isEmpty()) path = "/";
httpGetId = http[i] - >get(path, file);
}
}
```

槽函数 httpRequestFinished()代码如下：

```
httpRequestFinished(int requestId, bool error)
{
int temp = (requestId - miniResponseID)/2 + hostBegin. toInt();
if (error)
{
QTextStream out(file);
out < < "0 \ n0 \ n0 \ n0 \ n0 \ n0.0.0.0" < < endl;
}
}
```

3.3 各节点性能参数集成

由于采用 HTTP 协议的异步工作方式，每个节点返回信息是无序的，所以用于记录节点性能参数的 file 文件中的数据记录也是无序的。如何区分属于哪个节点的性能参数是关键，因此，通过返回节点 IP 地址作为节点的唯一标识，来区分节点并实现节点性能参数集成。如下是从 file 文件中提取并集成节点性能参数的关键代码：

```
file - >seek(0);
QLocale qvariant;
int hostNumber = hostEnd. toInt() - hostBegin. toInt() + 1;
int nodeinfoTemp[254][5], nodeinfo[254][7], nodeOrder[254];
QTextStream out(file);
QString datatmp;
for(int i = 0;i < hostNumber;i + +)
{
for(int j = 0;j < 5;j + +)
{
datatmp = out. readLine(20);
if(datatmp. toFloat() = = 0) nodeinfoTemp[i][j] = 0;
else nodeinfoTemp[i][j] = (int)(datatmp. toFloat() + 0.5);
}
datatmp = out. readLine(20);
if(datatmp. length() > 0) nodeOrder[i] = datatmp. split(". "). at(3). toInt();
}
```

3.4 集群系统性能参数的可视化

集群系统性能参数的可视化以图示方式直接展示各节点的实时性能及故障情况，是 Linux 集群系统负载实时监测系统的重要组成部分。可视化功能包括：允许操作者修改集群 IP 地址以监测不同的集群系统；允许用户选择不同的性能参数进行监测；允许设定不同时间间隔的定时器来监测。监测结果图示中，使用红色短柱指示故障节点，同时显示故障节点主机名。横坐标表示各节点 IP 的主机位，纵坐标分别展示节点的 CPU 使用率、内存占有率、磁盘 IO 负载、网络输入负载及网络输出负载，如上性能参数超过 95% 时用红色展示，指示节点的超负荷运行。集群节点性能参数可视化能够实时、直观地展现各节点的运行状态，篇幅所限，实现代码及各种集群性能的可视化监测结果在此不再赘述。

4 结束语

文中在详细分析单个 Linux 集群服务器节点性能参数及其提取方法的基础上，研究了利用 CGI 技术及 HTTP 协议异步请求，通过实时采集 Linux 集群服务器节点性能参数并利用 QT 编程实现 Linux 集群系统性能监测及其可视化的方法。文中实现的算法为进一步实现 Web 服务器动态负载均衡[12]打下了基础。实践证明：该方法具有占用系统资源少、可靠、实用的特点，可广泛应用于各种 Linux 集群系统[13]。

参考文献：

[1] 彭土有. 基于开源 LAMP 架构的 Web 打印技术研究[J]. 计算机应用，2008(28):92 - 99.

[2] 龚 梅，王 鹏，吴 跃. 一种集群系统的透明动态反馈负载均衡算法[J]. 计算机应用，2007(11):662 - 665.

[3] 彭土有，陈光海，李耀麟，等. 新编 Linux 网络组建与实训[M]. 北京：北京出版社，2008.

[4] 林洪祥，刘大茂. 基于服务分类和性能监测的负载均衡研究[J]. 计算机应用技术，2008,24:137 - 142.

(下转第 41 页)

4 结束语

Contourlet 变换相比于小波变换,具有更好的方向特性,能够更好地显示图像的几何特征,所以能为图像拼接提供更多的信息。但是由于在分解和重构过程中同时具有上、下采样过程,所以拼接过程不具有平移不变性。针对这一缺点,NSCT 方法应运而生,它利用 NSPFB 和 NSDFB 对图像进行多尺度、多方向分解,只具有上采样过程,具有平移不变性,得出的实验结果相比于非下采样小波方法,得到的结果图更加自然,细节信息更加清晰,过渡更加平滑。采用基于二维直方图的 B 样条曲面拟合的颜色校正方法在处理拼接图像明暗差异效果较好,相比于传统的颜色校正方法具有多个优点:不需要目标图;B 样条曲面具有良好的灵活性;二维直方图所描述的是不同量在整幅图像中所占的比例,跟空间位置无关,应用在全景图像的颜色校正时,此方法更加的简洁。

参考文献:

[1] 蔡丽欢,廖英豪,郭东辉.图像拼接方法及其关键技术研究[J].计算机技术与发展,2008,18(3):1-4.

[2] Lucas B D, Kanade T. An iterative image registration technique with an application in stereo vision[C]//In:Seventh International Joint Conference on Artificial Intelligence (IJCAI-81). Vancouver:[s.n.],1981:674-679.

[3] 钟　力,胡晓峰.重叠图像拼接算法[J].中国图像图形学报,1998,3(5):367-370.

[4] 肖　甫,吴慧中,肖　亮,等.基于静态小波分解和能量函数优化的图像拼接[J].光子学报,2007,36(4):763-767.

[5] Burt P J, Adelson E H. The Laplacian pyramid as a compact image code[J]. IEEE Trans. Common, 1983, 4(31): 532-540.

[6] 郁　梅,易文娟,蒋刚毅.基于 Contourlet 变换尺度间相关的图像去噪[J].光电工程,2006,33(6):73-77.

[7] 李剑峰,高志荣,熊承义.基于非下采样小波与方向上下文滤波的图像边缘检测[J].中南民族大学学报,2008,27(1):68-70.

[8] 张　强,郭宝龙.基于非采样 Contourlet 变换多传感器图像融合算法[J].自动化学报,2008,34(2):135-140.

[9] 梁　栋,殷　兵,于　梅,等.基于非抽样 Contourlet 变换的自适应阈值图像增强算法[J].电子学报,2008,36(3):527-530.

[10] Zhou Jianping, Cunha A L, Do M N. Nonsubsampled contourlet transform: construction and application in enhancement [C]//Proc. of IEEE International Conference on Image Processing. Genoa, Italy:[s.n.],2005:469-472.

[11] 张　强,郭宝龙.一种基于非采样 Contourlet 变换红外图像与可见光图像融合算法[J].红外与毫米波学报,2007,26(6):476-480.

[12] 戚世贵,戚素娟.一种基于图像特征点的图像匹配算法[J].理论与方法,2008,27(1):3-4.

(上接第 32 页)

[7] 白　刚,王重钢,隆克平,等.流控制传输协议 SCTP 及其性能分析与应用[J].北京邮电大学学报,2001,24(4):63-66.

[8] Stewart R R. 流控制传输协议 SCTP 参考指南(影印版)[M].北京:清华大学出版社,2003.

[9] 夏　云,孙力娟,叶晓国,等. SCTP 协议分析与仿真研究[J]. 计算机技术与发展,2009,19(11):27-30.

[10] 李　健,陶　洋. 基于 SCTP 多归属主机特性的多路径传输算法研究[J]. 重庆邮电学院学报,2005,17(4):1-4.

[11] 冉春玉,张广军,屈　力,等. 基于 SCTP 的客户/服务器的设计与实现[J].计算机技术与发展,2007,17(6):127-130.

[12] NIST NET Emulator[EB/OL]. 2005-07. http://snad.ncsl.nist.gov/nistnet/.

(上接第 36 页)

[5] 王文义,梁青云,王若雨,等.Linux 集群系统并行应用程序监测技术的研究[J]. 郑州大学学报,2005(2):98-101.

[6] 王若雨. Linux 集群可视化监测技术方法研究[J]. 中原工学院学报,2006(6):9-11.

[7] 王勇超,张　景,王新卫,等.基于 MPICH2 的高性能计算集群系统研究[J]. 计算机技术与发展,2008,18(9):101-104.

[8] Talia D, Trunflo P. Toward a synergy between P2P and grids [J]. IEEE Internet Computing, 2003,7(4):94-95.

[9] 潘志华,张　涛. Linux 预取算法分析与研究[J]. 计算机技术与发展,2009,19(12):93-96.

[10] 魏　晓,胡金初.基于 Linux 系统的分布式网络管理系统[J]. 计算机技术与发展,2007,17(6):60-63.

[11] 蔡志明、卢传富,李立夏. 精通 Qt4 编程[M]. 北京:电子工业出版社,2008.

[12] Devine, Boman K D, Erik G, et al. New Challenges in Dynamic Load Balancing[J]. Applied Numerical Mathematics, 2005,52(2):133-152.

[13] Buyya R, Cortes T, Jin Hai. Single System Image[J]. The International Journal of High Performance Computing Applications, 2001, 15(2):124-135.