第32卷 第20期 计 算 机 工 程 2006年10月
Vol.32 № 20 Computer Engineering October 2006
·软件技术与数据库· 文章编号：1000—3428(2006)20—0040—03 文献标识码：A 中图分类号：TP311

Linux环境下PowerPC-NC运行性能参数的获取

孙立江，朱 利，魏恒义
(西安交通大学软件学院，西安 710049)

摘 要：基于PowerPC的NC系统正逐步得到应用，但其运行性能监测工具尚未开发出来。PowerPC-NC系统硬件性能参数的获取必须解决两个主要问题：内核数据的采集，用户空间与内核空间的通信。通过对Linux环境下NC性能参数采集方法的研究，提出了一种基于LKM技术的虚拟设备驱动程序方法，并在此基础上实现了PowerPC-NC系统的数据采集模块。实验结果表明，该方法具有效率高、易用性好的特点。

关键词：NC系统；性能指标；LKM；内核空间

Collecting Techniques of PowerPC-NC System Runtime Performance Data under Linux Environment

SUN Lijiang, ZHU Li, WEI Hengyi
(School of Software Engineering, Xi'an Jiaotong University, Xi'an 710049)

【Abstract】PowerPC-based NC systems are gradually put into action, but the corresponding tools of runtime performance monitoring for these NCs are not yet developed. Two problems must be resolved for the collection of PowerPC-NC system runtime performance parameters: one is the collecting kernel data and the other is communication between user space and kernel space. Performance parameter collecting methods in Linux are researched and the virtual-device-driver method based on LKM is proposed. Based on this collecting method a data collection module of PowerPC-NC system is designed and implemented, which can collect the performance parameters of NC server. Experiment results indicate that this tool has a high effectiveness and is easy to be used.

【Key words】Network computer system; Performance parameter; LKM; Kernel space

系统性能监测是系统管理和性能优化过程中的一个基本手段，它包括数据采集和数据分析两个步骤。选择合适、高效的数据采集方法不仅能够提高获取数据的精度，而且节省被测试系统的资源，尽可能小地影响被测系统。

网络计算机(NC)是近几年新技术的产物，它的特点是具有很强的行业应用针对性，具有系统安全性高、网络结构简单、价格低廉的特点。因此，NC的应用正在普及。PowerPC处理器有较高的性价比，其应用具有平台独立的特点，广泛应用到NC系统里。目前，针对NC特别是针对基于PowerPC的NC系统的监控设备还很少。因此，研究开发面向NC的监控系统具有很重要的现实意义和广泛的市场前景。

目前Linux/Unix系统用来显示进程和系统资源使用的工具一般有Ps、Top、Free等，而这些系统信息一般只有在内核空间才有，对于这种进程与内核之间的交互，还没有一种被广泛接受的接口。在传统的应用中，Kvm过程需要用户程序了解从内核读入的数据结构的大小和格式，所以这样的程序可移植性较差；Performance Inspector程序[1]由于要给内核打补丁，需要重新编译内核，影响被测试系统的稳定性，而且费时；读Proc文件系统使用了较多的系统调用，数据采集效率较低。本文从实现的难易程度以及通信效率高低两方面考虑，建立NC系统自己的指标体系，在Linux内核中建立虚拟设备驱动程序，将Linux内核空间中与性能指标相关的参数数据返回给用户进程，达到数据采集的目的。

1 测量模型

本文所设计的数据采集组件是整套NC测试系统(network computer system test，NCST)中最核心的部分，该方法在基于PowerPC的Linux操作系统平台下采用LKM技术，选择在新内核系统中建立设备驱动的方式，使用Ioctl方法完成从内核获取性能参数。组件主要包括虚拟字符设备建立和数据采集处理模块两部分，其中虚拟字符设备是系统的控制核心，用户进程通过它来完成系统的操纵控制、数据清除和读取。参数获取模型如图1。

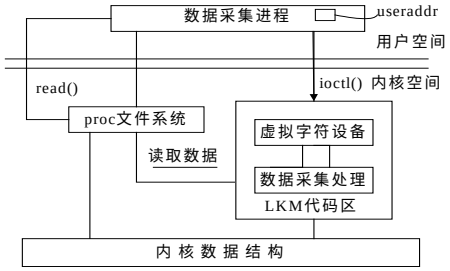

图1 性能测量模型

该模型的工作流程为：首先由用户采集进程通过Ioctl()方法向虚拟字符设备发送请求命令号cmd以及一个用户空间的地址指针useraddr；然后内核中的LKM代码区中的数据采集模块根据命令号来定位不同的数据采集函数；最后数据采集函数直接把数据写入到useraddr所指向的用户进程地址空

基金项目：国家“863”计划基金资助项目(2003AA1Z2610)
作者简介：孙立江(1980－)，男，硕士生，主研方向：计算机应用技术；朱 利，博士、副教授；魏恒义，副教授
收稿日期：2005-10-28 **E-mail：**sunlijiang@gmail.com

间。此外，测量数据也通过 Proc 文件系统挂接在 Proc 文件目录下，这样即可生成固定格式的测量数据，便于用第三方的专门分析工具进行读取。

2 基于虚拟字符设备的性能参数测量

现在，越来越多的应用程序需要编写内核级和用户级的程序来一起完成具体的任务，通常采用以下模式：首先编写内核服务程序利用内核空间提供的权限和服务来接收、处理和缓存数据；然后编写用户程序来和先前完成的内核服务程序交互。为了提高数据采集效率，减少代码量，虚拟字符设备中服务器与客户端内核性能参数(主要是服务器操作系统内核性能参数)采集部分在内核中实现。

2.1 性能指标相关的数据结构定义

NC 系统应用在服务器上性能参数的在 Linux OS 下通常可以直接或间接地从 Linux 内核数据结构中获得，这里给出关于 CPU、内存等硬件信息的数据结构：

(1)系统资源描述：包括 CPU 利用率、页交换、IO 使用情况、中断统计。其数据结构如下：

```
struct kernel_stat {
  unsigned int per_cpu_user[NR_CPUS],
                per_cpu_nice[NR_CPUS],
                per_cpu_system[NR_CPUS];
  unsigned int dk_drive[DK_MAX_MAJOR][DK_MAX_DISK];
  unsigned int dk_drive_rio[DK_MAX_MAJOR][DK_MAX_DISK];
  unsigned int dk_drive_wio[DK_MAX_MAJOR][DK_MAX_DISK];
  unsigned int dk_drive_rblk[DK_MAX_MAJOR][DK_MAX_DISK];
  unsigned int dk_drive_wblk[DK_MAX_MAJOR][DK_MAX_DISK];
  unsigned int pgpgin, pgpgout;
  unsigned int pswpin, pswpout;
  unsigned int irqs[NR_CPUS][NR_IRQS];
  unsigned int context_swtch;
};
  extern struct kernel_stat kstat;
```

(2)CPU 负载、内存使用情况、进程的统计等信息在内核里面使用 sysinfo 结构表示。

其数据结构如下：

```
 struct sysinfo
 {
     long uptime;                    /*系统运行时间(s)*/
     unsigned long loads[3];         /* CPU在1min，5min，15min内的平均负载 */
     unsigned long totalram;         /* 内存总量*/
     unsigned long freeram;          /* 可用内存*/
     unsigned long sharedram;        /* 共享内存量 */
     unsigned long bufferram;        /* 内存缓冲区使用量 */
     unsigned long totalswap;        /* 交换分区总量*/
     unsigned long freeswap;         /* 交换分区剩余容量*/
     unsigned short procs;           /* 当前运行的进程数*/
     unsigned long totalhigh;        /* 高端内存总量*/
     unsigned long freehigh;         /* 可用的高端内存量*/
     unsigned int mem_unit;          /* 内存单元(字节)*/
 ....
 };
```

所有数据结构，包括描述各进程的属性及资源使用情况的 task_struct 及 mm_struct，在内核中都是以变量的形式导出，形成内核的符号表，供内核的其它部分以及虚拟字符设备程序使用。

2.2 内核数据采集的具体实现

为了提高数据采集的效率以及数据精度，采用了一种虚拟字符设备驱动程序的方法在Linux内核中建立自己的数据采集模块，通过模块方法将新的虚拟设备安装到内核中就能方便使用[1]。

该方法的内核数据采集与 Proc 文件的数据采集方法类似，因此可以参考 Proc 的数据收据函数，即 Proc 的读函数。这些函数在 Linux 内核中的具体实现如下：

(1)针对内核数据结构 kernel_stat 的数据采集函数是 kstat_read_proc()，主要负责系统总体资源数据的提取，包括 CPU 利用率、交换、I/O 读写使用情况、内存换入、换出、中断统计、上下文切换统计等。

(2)针对内核数据结构 sysinfo 的数据采集函数是 meminfo_read_proc()及 loadavg_read_proc()，主要负责内存的使用情况统计以及 CPU 负载、进程总数的统计等。由于 Linux 已有一个 sysinfo()系统调用来读取它，因此在读 struct sysinfo 时，用户进程可以直接使用这个系统调用。

(3)针对内核数据结构 task_struct 及 mm_struct 的数据采集函数是 proc_pid_stat()和 proc_pid_statm()，主要负责各个进程的基本信息及一些资源使用统计。

(4)关于进程信息列表获取的伪代码如下：

```
 void p_list(void * useraddr)
 {
    read_lock(&tasklist_lock);    /*加读锁*/
     for_each_task(p) {    /*遍历进程表*/
            取得该进程号放入 useraddr;
              useraddr->num++;
 }
      read_unlock(&tasklist_lock);    /*解读锁*/
}
```

For_each_task()、read_lock()、read_unlock()的实现可参考内核源代码。

(5)单个进程详细信息获取的伪代码主要如下：

```
void readpid(void    *useraddr)
{
    struct task_struct *p;
    p=find_task_by_pid(useraddr->进程号);    /*根据进程号在系统的进程哈希表
中查找指定进程的 task_struct 数据结构*/
    进程信息向用户空间赋值；
}
```

find_task_by_pid()的实现可参考内核源代码。

3 实验结果

3.1 实验环境：

网络环境：100Mbps 局域网；

测试机：IBM P4 2.0GHz，内存 512MB；

操作系统：RedHat Linux9.0(kernel 2.4.20-8)；

NC 服务器： PowerPC 750(800MHz)CPU，512MB 内存，单 CPU，单硬盘；

操作系统：Yellow Dog Linux(kernel2.6.6)。

3.2 实验结果

实验围绕与 Proc 方法的效率比较展开，表 1 是 Proc 方法读取/proc/stat 时，采用 strace 对主要系统调用的跟踪结果。从表中可以看出，Proc 方法使用了过多的系统调用，总时间为 0.000 100s，总调用次数为 7 次，其中打开和读取文件占

用了大量的时间。

表 1 读/proc/stat 文件的实验结果

调用名	每次调用所用时间	调用次数
Open	0.000 012s	1
fstat64	0.000 003s	1
old_mmap	0.000 005s	1
Read	0.000 033s	2
Close	0.000 005s	1
Munmap	0.000 009s	1

系统调用的时间是消耗在系统空间的时间，对于读/proc 文件，在用户空间还要将读出的字符串数据进行分析处理。为了测得这一段时间的值，将这段处理程序单独放在一个模块里并制成一个系统调用，用 strace 测量其执行时间，然后减去一个空的系统调用的时间。得到其值为：0.000 021s。因此，Proc 方法所用总时间为：0.000 100+0.000 021= 0.000 121(s)。而用 strace 测试设备驱动方法读取 struct kernel_stat 数据结构，唯一的一次系统调用时间是：0.000 001 3s。

用同样的实验方法，分别统计采用不同方法获取 kernel_stat、sysinfo 及服务器进程列表和单个进程信息的数据所用时间，结果如图 2 所示。从图中可以明显看出，设备驱动方法的效率更高。

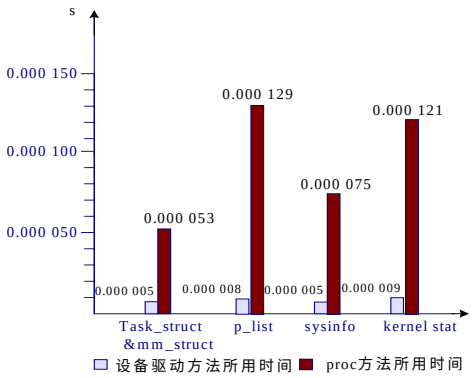

图 2 效率对比

利用该数据采集模块实现的 NC 系统性能监测工具所采集到的 CPU 负载，内存使用情况如图 3、图 4 所示。

图 3 中的曲线分别代表了 CPU 在这段时间内的 1min、5 min 和 15 min 的平均活动进程数，垂直虚线代表了事件发生的时间(NC 机同时启动的时间)，重点观察 1 min 负载曲线，发现在 NC 机启动的这 3 个 1min 内(12:21:02—12:24:02)，CPU 的 1 min 平均负载较高，负载曲线上升速度较快，系统性能迅速下降。

图 4 反映的是低端内存空闲的情况，对于该服务器而言，低端内存的大小就是实际物理内存的大小。从图中可以发现，随着 NC 机的启动和登录，空闲的低端内存迅速减小，直至消耗完毕，接着交换分区也会被占用。

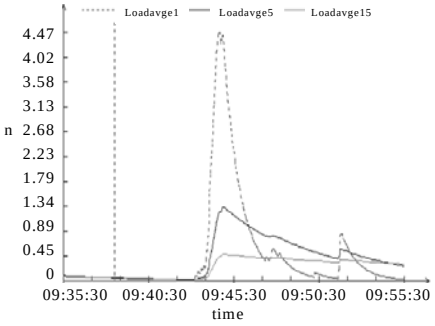

图 3 CPU 负载曲线

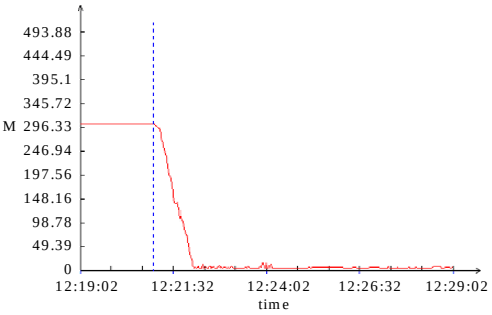

图 4 Low_free 曲线

4 总结

通过实验证明，在基于 Linux 系统的 PowerPC-NC 平台上，通过使用动态模块技术(LKM)在 Linux 操作系统中增加虚拟字符设备驱动程序的方法，可以大大提高获取内核性能参数的效率。它实现简单，避免了对文件和字符串的反复读取操作，比读 Proc 文件系统实现起来简洁、清晰、高效。

参考文献

1 Li Ge, Scott L, Wiele M V. Putting Linux Reliability to the Test[EB/OL]. http://www-900.ibm.com, 2003.

2 Rusling D A. Linux Programming White Papers[Z]. Coriolis, 2004.

3 Mata K, Pu J, Dewitt J. Five Easy-to-use Performance Tools for Linux on PowerPC[EB/OL]. http://www-900.ibm.com/developerWorks/cn/linux/es-PerformanceInspectoronpLinux/index_eng.shtml#IDAAEO0B, 2004.

4 Rubini A, Corbet J. Linux Device Driver(2nd Edition)[M]. O`Reilly, 2001.

(上接第 17 页)

参考文献

1 吕建平，秦　岩. 地图中河流自动识别方法的探讨[J]. 西安电子科技大学学报, 1996, 23(1): 54-58.

2 Dillabaugh C R. Semi-automated Extraction of Rivers from Digital Imagery[J]. Geoinfarmatica, 2002, 6(3): 263-284.

3 Kass M, Witkin A. Snakes: Active Contour Model[C]. Proceedings of the First International Conference on Computer Vision, London, 1987: 259-268.

4 李培华，张田文. 主动轮廓线模型(蛇模型)综述[J]. 软件学报, 2000, 11(6): 751-757.

5 郑华利，周献中. 彩色扫描地图的自动分色算法研究及实现[J]. 计算机辅助设计与图形学学报, 2003, 15(1): 29-33.

6 周献中，郑华利. 基于可变形模型及区域流向分析的等高线自适应矢量化算法[J]. 计算机学报, 2004, 27(8): 1056-1063.

7 张青年. 线状要素的动态分段与制图综合[J]. 中山大学学报, 2004, 43(2): 104-107.