基于 Intel VT－x 的 XEN 全虚拟化实现

顾晓峰，王 健
（东南大学 计算机科学与工程学院，江苏 南京 211189）

摘 要：由于X86体系结构对虚拟机支持的先天不足，基于此体系结构的虚拟机需要修改操作系统的源代码，称为泛虚拟化技术。泛虚拟化需要修改操作系统的源代码，故只能支持开源的操作系统，且这种虚拟机的实现也是比较困难的。为了解决这个问题，Intel公司提出了VT－x技术，该技术可以使虚拟机不需要修改操作系统的源代码，也就是所谓的全虚拟化技术，可以支持非开源的操作系统，且虚拟机的实现也比较简单。XEN是业界广泛看好的一款基于X86体系结构开源的虚拟机监视器，XEN 3.0开始实现了基于VT－x的全虚拟化技术，具有优越的性能和良好的体系结构。文中讨论了Intel的VT－x技术，并从CPU虚拟化、内存虚拟化和设备虚拟化三个方面介绍XEN实现全虚拟化的关键技术。

关键词：泛虚拟化；VT－x；全虚拟化；XEN
中图分类号：TP393.01 **文献标识码**：A **文章编号**：1673－629X(2009)09－0242－04

Full－Virtualization Implementation of XEN Based on Intel VT－x

GU Xiao-feng, WANG Jian
(Computer Science and Engineering School, Southeast University, Nanjing 211189, China)

Abstract: As the X86 architecture has congenitally defective support for virtual machine, the virtual machine based on the architecture need to modify the operating system source code, called Para－virtualization. Para－virtualization need to modify the operating system, which can only support the open source operating system, and the virtual machine is also more difficult to implement. To solve the problem, Intel Corporation proposed the VT－x technology, which enables virtual machine do not need to modify the operating system source code, that is, the so－called Full－virtualization, can support for the non－open source operating system, and the implementation of the virtual machine is also relatively simple. XEN is an open source virtual machine monitor based on X86 architecture, whose future is widespread optimistic by the industry, and XEN 3.0 began to implement the Full－virtualization technology based on VT－x, which has a superior performance and good architecture. This paper focused on Intel's VT－x technology, and introduced the key technologies about XEN implementing Full－virtualization from the prospects of CPU virtualization, memory virtualization and device virtualization.

Key words: para－virtualization; VT－x; full－virtualization; XEN

0 引 言

IBM公司在20世纪六七十年代最早提出了虚拟机(Virtual Machine, VM)概念并将其运用到VM/370系统中[1]。Internet发展起来以后，随着新兴的虚拟机应用不断出现，虚拟机相关技术得到不断发展。现在硬件性能的大幅提升，云计算的提出，使得虚拟机技术获得了良好的发展基础和广泛的应用前景。

目前虚拟机有Bochs、Denali、Hyper－V、KVM、QEMU、Virtual PC、VMware、XEN等，虽然它们的实现平台和实现细节各不相同，但基本都是基于图1这个典型的虚拟机模型。虚拟机监视器(Virtual Machine Monitor, VMM)是虚拟机技术的核心，它介于物理硬件和虚拟机之间，为每个虚拟机虚拟出一套独立于实际硬件的虚拟硬件环境(包括CPU，内存，I/O设备)。

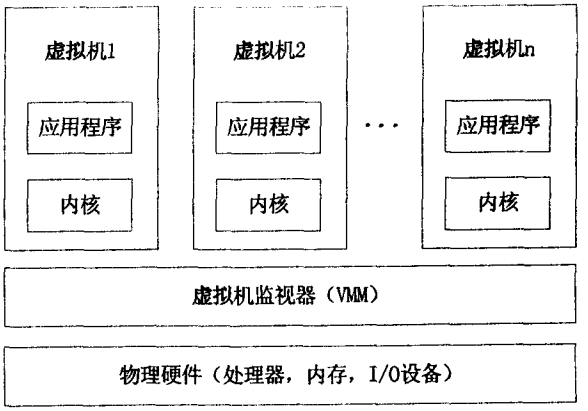

图1 典型的虚拟机模型

从应用程序的角度看，程序运行在虚拟机上同其

收稿日期：2008－12－28；**修回日期**：2009－02－28
作者简介：顾晓峰(1984－)，男，江苏南京人，硕士研究生，研究方向为虚拟机、嵌入式系统、高性能计算；王 健，副教授，研究方向为嵌入式系统、系统结构、高性能计算。

运行在对应的实体计算机一样,即运行在某一种特定的指令体系(Instruction Set Architecture, ISA)和/或操作系统上。VMM 抽象虚拟机的 ISA 可以等同于它运行的物理机器,也可以不完全相同。当虚拟的 ISA 与物理的 ISA 相同时,该虚拟机可以运行不需要修改的操作系统;而当二者不相同时,客户机的操作系统(Guest OS)就必须修改。根据 VMM 抽象虚拟机架构的不同,或根据是否需要修改 Guest OS,虚拟化技术又可以分为泛虚拟化(Para－virtualization)和完全虚拟化(Full－virtualization)两类。

1　XEN 虚拟机及 Intel VT－x 硬件虚拟技术

XEN 是剑桥大学教授 Ian 等领导开发的基于 X86 体系结构的一个开源的、优秀的虚拟机管理软件,得到业界的广泛支持,目前已集成到 BSD、Solaris、Linux 等操作系统中。X86 体系结构设计之初没有考虑对虚拟机的支持,这为基于 X86 的虚拟机技术带来了一些问题和挑战,Robin 和 Irvine 分析了 X86 体系结构在虚拟化时的问题[2]。所以早期的 XEN 采用泛虚拟化技术,需要修改 Guest OS 的内核源代码,避开那些因虚拟化后存在漏洞的指令。这样,XEN 上可以运行 Linux、Unix 等开源的操作系统,并可以获得很好性能,支持多约 100 个运行 Guest OS 的虚拟机,但无法运行那些非开源的操作系统,比如广受欢迎的 Windows。

1.1　Intel VT－x 硬件虚拟技术

由于 X86 体系结构对虚拟技术的支持存在先天不足,2005 年 1 月 20 日 Intel 向外界发布了代号为 Vanderpool 的硬件虚拟技术(Virtualization Technology),其中包括支持 X86 体系架构的 VT－x 技术[3]。Intel 的 X86 CPU 通过 CPUID. 1:ECX. VMX[bit 5]＝1 表示 CPU 支持 VT－x 技术。VT－x 提出了一种新的 CPU 操作,称为 VMX(Virtual Machine eXtensions)。VT－x 同时提出了两个新的 CPU 工作模式:VMX root 模式和 VMX non root 模式。root 模式与传统的 X86 工作模式没有太大的差别,只是增加了一些支持 VMX 的指令;而 non root 模式是在 VMM 控制管理下的 X86 环境,non root 模式下的 CPU 操作是受限的。VMX 可以在 root 模式和 non root 模式下运行。VMM 运行于 root 模式,而各个虚拟机运行于 non root 模式,虚拟技术就是通过 root 模式和 non root 模式之间切换实现的,由 non root 模式切换到 root 模式称为 VM 退出(VM Exit),这有可能是虚拟机中的操作系统执行了某些特权指令,而这些指令需要 VMM 代为执行;由 root 模式切换到 non root 模式称为 VM 进入(VM Entry),这有可能由虚拟机调度引起。

VT－x 技术的出现,简化了虚拟机的设计,并提高了 VMM 对虚拟机管理的灵活度和粒度。图 2 显示了在 VT－x 技术的应用下虚拟机的生命周期。首先,VMM 可以通过 VMXON 打开 VT－x 的硬件虚拟化功能,进入 VMX root 模式。接着,通过 VMLAUNCH 和 VMRESUME 发生 VM Entry,进入运行在 VMX non root 模式下的虚拟机(VM)。当 VM 中运行的操作系统执行某个指令或发生某个事件时,会自动触发 VM Exit,这时 VMM 重新获得控制权。VMM 根据 VM Exit 发生的原因做出相应的处理,之后再通过 VM Entry 进入虚拟机。最后,VMM 可以通过 VMXOFF 关闭 VT－x 的硬件虚拟化功能。

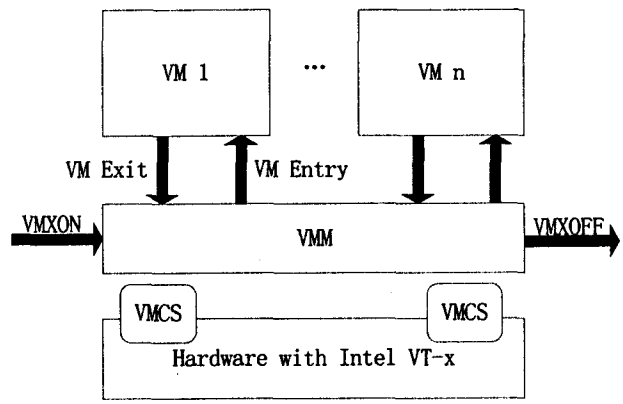

图 2　虚拟机的生命周期

VT－x 为每个 VM 设计了一个控制结构 VMCS(Virtual Machine Control Structure)来保存 VM 和 VMM 信息。VMCS 包括六个域,如图 3 所示:①主机状态域(Host State Area)保存 VMM 的各种状态信息;②客户状态域(Guest State Area)记录了 VM 的各种状态信息,包括寄存器状态,如段寄存器、控制寄存器等,还包括 VM 的虚拟 CPU 所处的状态;③虚拟机执行控制域(VM－Execution Control Fields)定义了 VM 在 non root 模式下的执行行为,也就是用来指定何种事件会触发 VM Exit,或何种事件不会触发 VM Exit;④VM Exit 控制域(VM Exit Control Fields)保存 VM Exit 相关的控制信息;⑤VM Entry 控制域(VM Entry Control Fields)保存 VM Entry 相关的控制信息;⑥VM Exit 信息域(VM Exit Information Fields)记录上一次发生 VM Exit 的信息,而且是只读权限,不能对这个域进行写操作。VMCS 由指令 VMPTRST, VMPTRLD, VMREAD, VMWRITE 和 VMCLEAR 来操作。VMM 可以对每个 VM 使用不同的 VMCS,也可以同一个 VM 内多个处理器使用不同的 VMCS。当执行 VM Entry 时将 VMM 的状态信息保存到 VMCS 的 Host State Area,并加载相应的 VM 的 VMCS 的 Guest State Area

到 CPU 中;执行 VM Exit 时则将当前 VM 的状态信息保存到 VMCS 的 Guest State Area,并加载 VMCS 的 Host State Area 到 CPU 中。VMCS 的存在使得 VMM 可以灵活地管理和配置 VM。

图 3 虚拟机控制结构

1.2 XEN 全虚拟化的体系架构

VT-x 硬件虚拟技术出现后,XEN 实现了对 VT-x 的支持。从 XEN 3.0[4]开始就是实现了具有硬件虚拟技术支持的全虚拟化,此时 Guest OS 不需要修改内核源代码,Windows 这种非开源的操作系统也可以在 XEN 上运行,弥补之前泛虚拟化的不足。图 4 为 XEN 3.0 的全虚拟化的体系架构示意图。XEN 中的虚拟机也称为域(Domain),图中运行着三个虚拟机,包括特权域 Domain 0、非特权域 VM 1 和 VM 2。其中,Domain 0 和 VM 1 是泛虚拟化的虚拟机,运行于其上的 Guest OS 称为 XenoLinux,特指内核修改过后的 Linux;VM 2 是一个全虚拟化的虚拟机,又称为 VMX Domain,它的 Guest OS 的内核源代码不需要修改,它必须运行在 VT-x CPU 上。

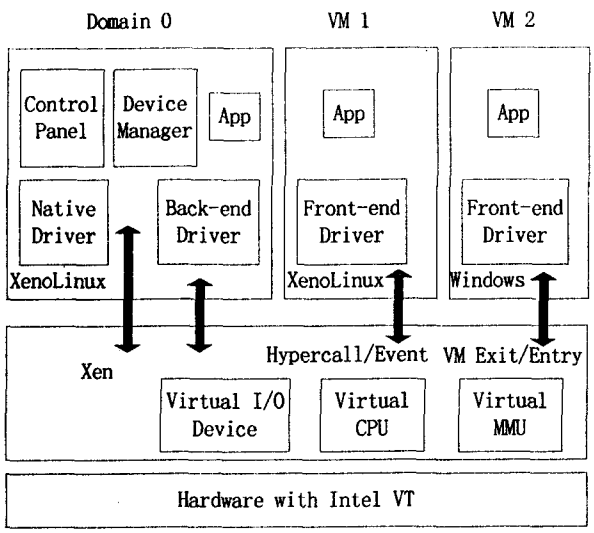

图 4 XEN 3.0 全虚拟化的体系架构

XEN,也叫 Hypervisor,亦即是 VMM,直接运行于硬件裸机上,运行在 root 模式下并处于最高特权级 ring 0 上,用于控制管理和虚拟机硬件资源,负责对各个虚拟机的调度以及对共享资源的控制访问等。Domain 0 的内核运行在 root 模式下并处于特权级 ring 1 上,又称为 Xen0,它是一个具有特殊地位的虚拟机,作为对 VMM 的扩展,拥有硬件的真实驱动程序,负责硬件设备的控制访问、管理和调度 Guest OS。Domain 0 中的控制面板(Control Panel,CP)是 XEN 的控制模块,作为一个特殊的应用程序通过 Hypercalls 来创建、保存、恢复、迁移和销毁各个 Domain。Domain 0 中的设备模块(Device Manager 或 Device Model,DM)[5]用来为虚拟机虚拟物理设备,为虚拟机提供访问物理设备的路径,实现物理的设备共享。

1.3 XEN 全虚拟化时的 CPU 虚拟化

由于 XEN 实现对 VT-x 技术的支持,当运行于 non root 模式下的虚拟机需要执行一些特权指令时,比如 I/O 访问、对控制寄存器的操作、MSR 的读写等指令,此时发生 VM Exit,进入 root 模式,XEN 取得控制权,通过读取 VMCS 中的 VM Exit Information Fields 得到发生 VM Exit 的原因,在 vmx_vmexit_handler 函数中开始执行相应处理。

目前 XEN 使用 BVT(Borrowed Virtual Time)[6]调度算法。虚拟机获得一个时间片后,在这个时间片内连续运行它的逻辑 CPU,时间片消耗完后,XEN 会调度下一个虚拟机运行。

1.4 XEN 全虚拟化时的内存管理虚拟化

在 VMM 中有一个叫 virtual Memory Management Unit(MMU)的模块,virtual MMU 模块主要用来维护一套完整的内存访问和管理机制,包括 PDT、PTE、TLB 和 CR3 等数据,完成 Guest OS 的虚拟地址和机器地址间转换,并确保 Guest OS 能够正确访问内存。

XEN 管理内存有两种方式:直接方式(Direct Mode)和影子方式(Shadow Mode)。其中,泛虚拟化时这两种方式都可以使用,而全虚拟化时,只能使用影子方式。这是因为,全虚拟化时,Guest OS 的内核源代码是未修改过的,VMM 对 Guest OS 是透明的。由于直接方式不能用于全虚拟化,故不讨论它,这里讨论的是影子方式。影子方式指 virtual MMU 通过维护影子页表(Shadow Page Table)实现对内存的管理。VMX Domain 中的 Guest OS 看到的内存是 VMM 为其分配的物理地址,而且是一段连续的空间,Guest OS 负责维护虚拟地址(Virtual Address,VA)到物理地址(Physical Address,PA)转换的页表,而 VMM 维护 PA 到真实的机器地址(Machine Address,MA)的映射,VMM 维护

的这个页表就是影子页表。VMM 要为 Guest OS 中的每个进程维护一个影子页表,并让真实的 CR3 指向此处。VMM 还要负责 Guest OS 的页表和影子页表的同步。VMX Domain 把 Guest OS 更新 PDT 和 PTE、刷新 TLB 和 CR3 操作设为触发 VM Exit 的条件,当 VM Exit 发生时 VMM 获得控制权,VMM 中的 virtual MMU 模块调用相应的处理函数,根据 PA 和 MA 的关系修改影子页表相应的表项。

1.5 XEN 全虚拟化时的设备管理虚拟化

XEN 只允许 Domain 0 访问真实的硬件,其他 Domain 都不能访问真实硬件,也就是说 I/O 设备的访问必须经由 Domain 0 来完成。Domain 0 中的 virtual I/O 设备模块(device models)向 VMX Domain 提供了各种 I/O 设备的抽象,包括显示器、键盘、鼠标、IDE 硬盘、软驱、光驱、网卡、声卡等等。

设备模块的实现借用了开源项目 QEMU 的设备模拟(Device Emulation)模块[7]。它的基本思想是 Domain 0 提供了一个操作平台和设备管理模型的环境,为每个 VMX Domain 运行一个设备模块的实例,提供虚拟 I/O 服务。设备模块 Device Model 的主要功能等待来自 VMX Domain 中 Guest OS 的 I/O 操作请求,并把请求转发给相应的设备模拟模块。一旦设备模拟模块完成请求,它就将结果返回。设备模块使用了 XEN 的轻量级的事件通道(Event Channel)发送和接收异步通知,这些通知包括虚拟中断请求、物理中断请求以及域间的通信。为提高设备访问的性能,通过共享内存页来实现数据交换。这个共享内存页采用授权表来进行访问控制。

2 XEN 全虚拟化时的性能评测

由于 XEN 的目标是提供高性能的虚拟机运行环境,因此其性能主要由运行于其上的 Guest OS 的性能来反映。于是,一切针对操作系统的 Benchmark 都可以用于检测 XEN 的性能,比较 XEN 中运行的操作系统的性能和同样的操作系统在真实的硬件机器运行时的性能。

SPEC CPU2000[8]是一组针对 CPU 和内存的测试。本实验用 SPEC 2000 对运行在 XEN 中和直接运行在物理机器上的 Windows XP SP2 性能进行测试。对 CINT 2000,CFP 2000 分别进行测试并加权平均,然后对两者取相同加权得到总的计算性能测试结果。图 5 给出 SPEC 2000 的测试结果。测试平台是:Intel Core 2 CPU,6320,1.86GHz;512M 内存;80G 硬盘。

图中三个柱状体分别代表使用 CINT 2000,CFP 2000 和两者都用时运行在 XEN 中的 Windows XP SP2 占直接运行在物理机器上的 Windows XP SP2 的性能百分比,这个百分比由纵轴表示。从图中可以看出,在这组针对 CPU 和内存的测试中,XEN 与实际的物理机器具有基本接近的性能,这表明在 CPU 和内存的虚拟化方面,XEN 具有很高的性能。

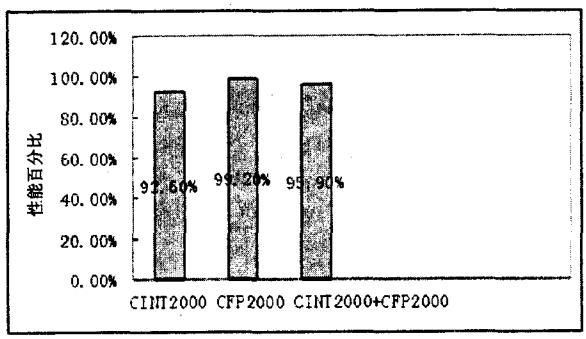

图 5　SPEC 2000 性能测试结果

3 结束语

XEN 由于其优越的性能、开源性和良好的架构,使得 XEN 成为业界最优秀的虚拟机之一。虚拟机技术正日益受到 IT 界的重视。虚拟技术很有可能在 XEN 和 Intel VT 技术的推动下给当前的计算机应用领域带来一次应用革命。此外,虚拟化技术是企业 IT 基础设施建设和管理上的一个重大进步,虚拟化技术降低了 IT 基础结构总成本,并为企业 IT 用户提供了更好的服务水平,显著提高了 IT 资源灵活性且极大地降低了 IT 基础设施的复杂性[9]。

参考文献:

[1] Creasy R J. The Origin of the VM/370 Time－sharing System[J]. IBM Journal of Research and Development, 1981,25(5):483－490.

[2] Robin J S, Irvine C E. Analysis of the Intel Pentiums Ability to Support a Secure Virtual Machine Monitor[EB/OL]. 2005[2007]. http://citeseer. ist. psu. edu/kiyanclar05survey. html.

[3] Intel Corporation. IntelR Virtualization Specification for the IA－32 Intel Architecture[EB/OL]. 2005[2006]. http://cache－www. intel. com/cd/00/00/19/76/197666_19766. pdf.

[4] Pratt I, Fraser K, Hand S, et al. XEN 3.0 and the Art of Virtualization[EB/OL]. 2005[2007]. http://www. linuxsymposium. org/2005/linuxsymposium_procv2. pdf.

[5] Fraser K, Hand S, Neugebauer R, et al. Safe Hardware Access with the Xen Virtual Machine Monitor[EB/OL]. 2004[2006]. http://www. cl. cam. ac. uk/Research/SRG/netos/papers/2004－oasis－ngio. pdf.

(下转第 249 页)

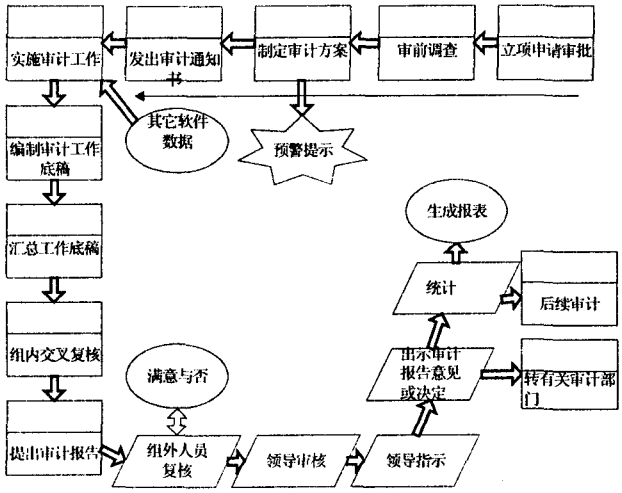

图 5　财务收支审计业务流程图

是一个事件驱动(event - driven)的、基于组件(component - based)的、用以丰富网络程序中用户界面的框架。ZK 包括一个基于 AJAX 事件驱动的引擎(engine),一套丰富的 XUL 和 XHTML,以及一种被称为 ZUML(ZK User Interface Markup Language,ZK 用户界面标记语言)的标记语言。后台采用了 DAO 模式进行设计和实现相关的功能类。本系统的关键部分是对审计业务流程中涉及的报表的处理,下面是一个报表的后台实现部分代码:

```
public interface Repor2Dao{
    public Report2 searchReport2ById(String id);
    publice Report2 searchReport2ByAuditId{String AuditId);
    public int searchAllReport2Count();
    public SplitPageResult<Report2>searchPageList(final int begin,
final int count);
    public void addReport2(Report2 report2);
    public void updateReport2(Report2 report2);
    public void deleteReport2(String id);}
public class Report2DaoImpl extends BaseDaoImpl implements Re-
port2Dao,ProxyManagerHandler{
    public void addReport2(Report2 report2) {
    report2,setId(this,objectGenerator,nextValue()};
    this,getSqlMapClienTemplate(),insert("Report2,addReport2",
report2);
    }
    public void updateReport2(Report2 report2) {
    this, getSqlMapClientTemplate ( ), update ("Report2,
updateReport2",report2)}
    public void deleteReport2(String id){
    getSqlMapClientTemplate(),delete("Report2,deleteReport2",
id);)}
```

5　结束语

电子政务的不断发展,提高了政务系统办事效率。采用 J2EE 和工作流技术设计了跨平台的、功能完善的、界面友好的、安全稳定的公安厅审计信息系统。该系统已经在某省公安厅投入使用,其运行效果良好,较好地完成对省市县三级审计业务工作,并且具有很好的可扩展性和可维护性。下一步的主要工作有:

①进一步研究网上审批工作流模型,增强系统性能,做好进一步的优化工作;

②根据实际需要,进一步完善和增加系统功能。

参考文献:

[1] 唐协平,张鹏翥.电子政务需求研究综述[J].计算机应用研究,2008,25(7):1921 - 1931.

[2] 赵　东,周明天.分布对象评述[J].计算机工程与应用,2000(12):7 - 10.

[3] Altendorf E,Hohman M,Zabicki R. Using J2EE on a Large,Web - Based Project[J]. IEEE Software,2002,19(2):81 - 89.

[4] Johnson R. J2EE Development Frameworks[J]. IEEE Computer,2005,38(1):107 - 110.

[5] 于孜清,冉蜀阳,李　胜.基于 MVC 模型的远程教材管理系统的设计与实现[J].计算机技术与发展,2006,16(1):18 - 22.

[6] 郑　刚.一种基于工作流技术的协同办公系统的设计[J].计算机技术与发展,2007,17(1):24 - 29.

[7] 辛　华,薛福任.工作流技术及其在网上审批中的应用[J].计算机工程与应用,2004(22):217 - 219.

[8] 句群慧,张华新,胡维华.基于 J2EE 的部门交互式审批平台的设计与实现[J].计算机应用与软件,2005,22(12):139 - 141.

(上接第 245 页)

[6] Duda K J, Cheriton D R. Borrowed - Virtual - Time (BVT) scheduling: supporting latency - sensitive threads in a general - purpose scheduler[C] // Proceedings of the 17th ACM SIGOPS. Symposium on Operating Systems Principles, volume 33(5) of ACM Operating Systems Review. New York, USA: ACM Press, 1999:261 - 276.

[7] Intel Corporation. Intel Itanium Architecture Software Developer's Manual[EB/OL]. 2006. ftp://download.intel.com/design/Itanium/manuals/24531805.pdf.

[8] Henning J L. SPEC CPU 2000: measuring CPU performance in the New Millennium[M].[s.l.]:IEEE Computer Society, 2000:28 - 35.

[9] 刘爱军,耿国华. 基于 x86 的虚拟机技术现状、应用及展望[J]. 计算机技术与发展,2007,17(11):250 - 253.