ISSN 1673-9418 CODEN JKYTA8
Journal of Frontiers of Computer Science and Technology
1673-9418/2010/04(01)-0089-08
DOI:10.3778/j.issn.1673-9418.2010.01.010

E-mail:fcst@public2.bta.net.cn
http://www.ceaj.org
Tel:+86-10-51616056

虚拟机全系统回放技术研究*

刘海坤[1+],金 海[1],崔 涛[2],廖小飞[1]

1. 华中科技大学 计算机科学与技术学院 服务计算技术与系统教育部重点实验室 集群与网格计算湖北省重点实验室,武汉 430074
2. 中兴通讯公司,北京 100191

Research on Virtual Machine Based Full System Replay*

LIU Haikun[1+], JIN Hai[1], CUI Tao[2], LIAO Xiaofei[1]

1. Services Computing Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
2. Zhongxing Telecommunication Equipment Co., Beijing 100191, China

+ Corresponding author: E-mail: haikunliu@smail.hust.edu.cn

LIU Haikun, JIN Hai, CUI Tao, et al. Research on virtual machine based full system replay. Journal of Frontiers of Computer Science and Technology, 2010,4(1):89-96.

Abstract: This paper studies the scheme of full system replay in virtualization infrastructure. A virtual machine based replay system (VMRS) is designed and implemented for para-virtualized device model. All the non-deterministic events occurred in the virtual machine are recorded at the hypervisor layer, and the system execution stream recurs during replay by event emulation. Experimental results show VMRS has good performance and scalability.

Key words: virtual machine(VM); system replay; event emulation; non-deterministic event

摘 要:研究了虚拟化体系结构下全系统回放的方法,设计实现了基于半虚拟化设备模型的虚拟机回放系统VMRS(virtual machine based replay system)。通过在虚拟机管理器中记录虚拟机内发生的不确定性事件并在回放时模拟不确定性事件的发生,从而完整重现系统的执行流。实验表明系统具有良好的性能和可扩展性。

关键词:虚拟机;系统回放;事件模拟;不确定性事件

文献标识码:A **中图分类号:**TP316.8

* The National Grand Fundamental Research 973 Program of China under Grant No.2007CB310900 (国家重点基础研究发展规划(973)).
Received 2009-06, Accepted 2009-08.

1 引言

随着系统软件的规模不断扩大,任务的复杂度不断提高,软件的并发性不断加强,软件的潜在错误越来越难以捕捉,软件系统的排错和调试也越来越困难,由于软件复杂性带来的安全性漏洞也越来越难以检测。回放技术可以重现系统的执行流,可以帮助发现系统中潜在的错误和安全漏洞。但是,目前对于该技术的研究还主要集中在用户层和操作系统层[1],无法回放完整的操作系统,而且对于系统内核程序及与系统不确定事件密切相关的多线程程序,尚没有一种很好的回放方法。虚拟机[2]的出现为解决以上问题提供了新的思路。在虚拟化体系构架下,虚拟机监视器(virtual machine monitor,VMM)通过对计算机底层硬件进行抽象,为上层的操作系统(即虚拟机)提供了一个模拟的硬件平台。虚拟机在执行过程中发生的所有事件都可以被 VMM 捕获到,通过 VMM 来完整记录操作系统上的不确定性事件,就可以对系统状态进行完全回放。系统级的回放结合检查点技术也因此被广泛应用于系统恢复、软件调试、入侵检测等领域。

尽管基于虚拟机的回放技术尚不成熟,但是近年来也涌现了一批典型系统,如 Revirt[3]、Retrace[4]以及 ExecRecorder[5]等。这类系统都基于这样一个原理:不需要记录系统中执行的每一条指令,仅仅记录影响进程执行的不确定事件以及这些事件发生的位置或者时间点,在回放的时候通过这些信息重构系统之前的运行状况。不确定事件记录大体上由两部分组成:时间和外部输入。时间是指在指令流中某个事件发生的精确位置,外部输入则指由一些外围输入设备产生的 I/O 中断。但这几个系统普遍存在的问题是需要修改客户操作系统,通用性差,需要记录的日志量大,系统开销大。文章结合以上系统,提出了一种基于虚拟机管理器 Xen[6]的操作系统回放机制,该机制能够在不修改操作系统以及应用程序的条件下对系统状态进行完整快捷的回放。

2 基于虚拟机的回放机制

为了对操作系统的执行流的进行完整回放,文章基于半虚拟化 Xen 实现了虚拟机回放的原型系统 VMRS(virtual machine based replay system)。Xen 提供的半虚拟化模型为每个操作系统构建了相互独立的虚拟硬件接口,该特性为从操作系统外围即 VMM 记录其行为提供了必要的环境支持。VMRS 对操作系统运行时的行为进行精确记录,而在以后某一时间,可以利用这些日志性息在同类操作系统上重现这些行为。

2.1 VMRS 系统架构

VMRS 主要包括日志记录模块、日志回放模块和控制模块。图 1 展示了基于半虚拟化 Xen 的虚拟机回放系统的总体构架。其中,日志记录模块用来获取非特权域操作系统的运行日志,日志回放模块根据日志记录执行具体的回放操作,而控制模块则是一系列超级调用的集合,用来为上述两个子系统提供功能接口。

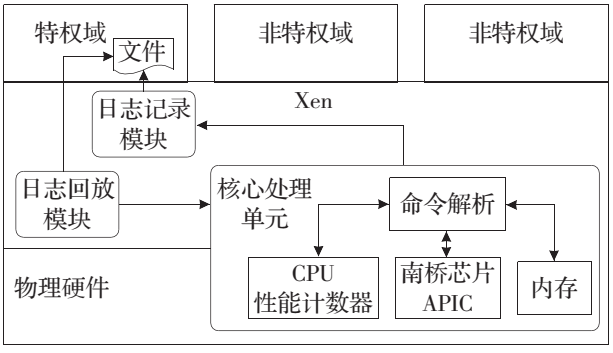

Fig.1 The system structure of VMRS

图 1 VMRS 系统总体构架

日志记录模块主要实现如下几个功能目标:

(1)分析记录日志内容。对影响系统行为的事件进行分析归类,只记录非确定性事件,避免了冗余数据,节省了日志存放空间;

(2)日志记录的标识。采用指令分支数[7]进行定位使得记录更加精确,且便于回放;

(3)日志的安全存储。VMRS 中记录的日志内容并没有像一般系统日志那样存放在被记录的操作系统中,而是存放在特权域(也称为 Domain 0)的文件系统中,避免了当目标操作系统被入侵时日志被篡改

的可能。

日志回放模块将日志分析后，批量写入 Xen 中的缓存区,通过对性能计数器的控制来获取事件发生的时间点,并操作特权域与非特权域之间的虚拟硬件接口来达到模拟回放的目的。其功能目标亦有三个:

(1)设计高效的日志分析程序与日志传输机制,建立特权域与 Xen 的共享内存,使日志文件以最高效的方式传输到 Xen 的缓存区中;

(2)对性能计数器与高级可编程中断控制器等硬件重新编程,以使其达到回放系统的配置要求,此外,由于性能计数器中断 PMI(performance monitor interrupt)是以非可屏蔽中断的方式通知 CPU 的,因此还需对 Xen 的非可屏蔽中断的处理方式做必要的修改,这样才能捕捉到过去事件发生的时间点;

(3)中断模拟程序的定位与设计,为了不破坏 Xen 的整体结构,VMRS 将模拟程序依旧放到特权域中,在其中配置对应与 Xen 的日志缓存区的影子缓存区。

2.2 VMRS 系统设计

在 Xen 的半虚拟化模型下非特权域(也称为客户机操作系统)的 I/O 设备模型如图 2 所示。运行在非特权域上的客户机操作系统并没有真实的设备驱动程序,它仅包含一套虚拟的驱动程序,即所谓的前端驱动程序,该驱动程序通过特权域与非特权域之间事先建立好的共享环缓冲区和事件通道,将客户机操作系统的 I/O 请求递交给特权域中的后端驱动程序,该程序调用真实设备驱动程序完成对 I/O 设备的操作后将执行结果按原路返回给客户机操作系统的前端设备驱动,这样客户机操作系统便完成了对真实物理设备的一次操作。由此过程,在前后端驱动之间,即特权域与非特权域之间的共享环缓冲区和事件通道上可以截获所有外部中断的中断号与数据，当然,也可以通过对共享环缓冲区和事件通道的操作来达到模拟中断的效果,这是 VMRS 回放系统的基本设计思路。

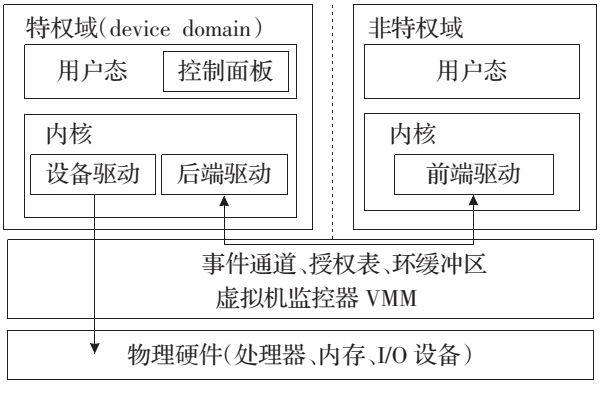

Fig.2 I/O device model of para-virtualized Xen

图 2 Xen 半虚拟化 I/O 设备模型

对于日志记录模块,首先需要在特权域中找到对应中断的后端驱动程序或使用相应设备驱动的守护进程,因为对共享内存区与事件通道的操作一般都在这里实现,而后在其中添加一个超级调用,即可将该中断发生时的数据与中断类型等相关信息写入 Xen 中的缓存区,从而达到日志记录的目的。当 Xen 中日志缓存区的剩余空间达到系统给定的一个阈值时,Xen 会主动向特权域发送一个虚中断,激活事先为该虚中断注册的中断服务程序,进而唤醒日志读取线程(内核线程)将 Xen 缓存区中的数据写入位于用户空间的日志文件中永久保存。详细流程如图 3 所示。

对于日志回放模块,需要先分析先前记录的日志文件,取出模拟该中断需要的一切信息,通过特权域内核写入 Xen 的缓存区。这里需要实现日志的分离,因为 Xen 的内存空间有限,不可能一次把所有的日志数据都写入到 Xen 的内存缓冲区，因此在特权域内核中将日志数据分离成两部分,一部分由性能计数器值组成,其他数据组成另一部分,前者写入 Xen 的缓存区,后者则放在特权域内核的影子缓存区中。每次回放之前都会检测 Xen 缓存区中可用记录数目,当记录数目不足时,Xen 会主动向特权域发送一个虚中断,激活事先为该虚中断注册的中断服务程序,进而唤醒日志写入线程(内核线程),将日志分析程序的结果写入 Xen 缓存区;当记录数目足够时,取出一条记录,将其写入性能计数器,当性能计数器溢出时会产生性能计数器中断,该中断的服务程序依然会向特权域发送一个虚中断,来激活事先为该虚中断注册的中断服务程序,进而唤醒中断模拟线程来完成历史中断的模拟任务。当中断模拟线程伪装成后端驱动程序

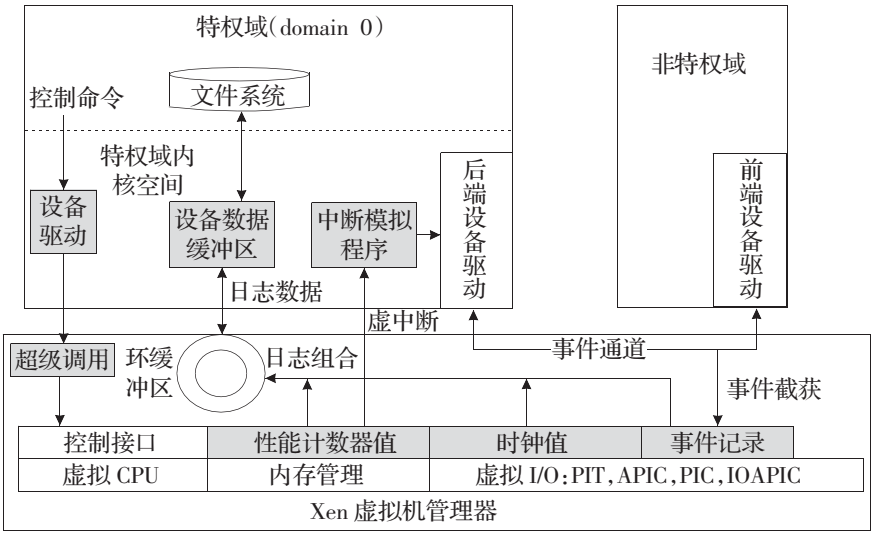

Fig.3 The theoretical implementation of VMRS

图 3 VMRS 系统原理框图

向非特权域发送事件通道消息后,非特权域被调度的时候就会处理本事件通道消息来完成事件的回放。

3 VMRS 系统实现关键技术

3.1 日志的传输

VMRS 采用了基于共享内存的日志数据传输方式来保证系统的高效性,其操作流程如图 4 所示。VMRS 在虚拟机回放系统初始化时在 Xen 与特权域内核之间建立一个共享的内存区域。具体创建流程为:首先,Xen 从名为 Xenheap 的堆上分配与预设缓冲区大小相同的内存页面(一般是一页内存),而后利用 Xen 提供的一系列宏操作,将此内存页的虚拟地址转化为机器地址页框号,然后利用此页框号建立 Xen 与特权域的共享内存映射,即操作 Xen 中的内存页面授权表, 使得特权域有权限访问此内存页面,最后以虚中断方式通知特权域来取走此页框号。当特权域获得此机器页框号后,经过一系列宏操作,将机器页框装化为自己可以识别的虚拟地址,然后就可以读写这块内存区域的数据了。

当目标操作系统有键盘等外部中断事件发生时,键盘值一类的外部数据会沿超调用的路径到达 Xen 内部,并与当前性能计数器的值合成日志,然后写入共享缓存区。在回放系统开启后,时钟日志数据会自动结合性能计数器值写入共享缓存区。当缓冲区中的数据量达到预设阀值时,缓存区控制模块会向特权域内核发送虚中断,特权域内核中相应的中断服务程序程序会唤醒使用日志记录系统虚拟设备的日志读取线程,将共享缓存区中的日志数据直接写入特权域的用户缓存区中,最后将用户缓存区中的数据写入日志文件,完成此次日志传输。

逻辑上讲,VMRS 中回放过程的日志传输路径是日志记录过程的逆过程,但又有所不同,回放过程中缓冲区中只存放性能计数器的值,而不具有日志记录过程中缓存区的日志数据结构。当回放模块需要日志信息时,它将控制缓存区控制模块向特权域内核发送日志传输虚中断,特权域接收到此虚中断后唤醒日志回放模块的虚拟设备的日志传输线程,该线程通过执行日志分离程序,将日志信息中的性能计数器数据写入共享缓存区,而将中断数据信息写入共享缓存区的影子内存区域。日志分离的原因一方面是由于 Xen 的内存空间有限(64 MB),而特权域内核的虚地址空间有 1 GB 之多,这样做可以使每次日志传输时最多的写入日志记录,从而减少日志传输次数,避免过多的超调用开销;另一方面是因为中断数据信息最终依然要写回特权域与非特权域之间的共享内存空间,这样做可以避免庞大数据信息在 Xen 与特权域内核之

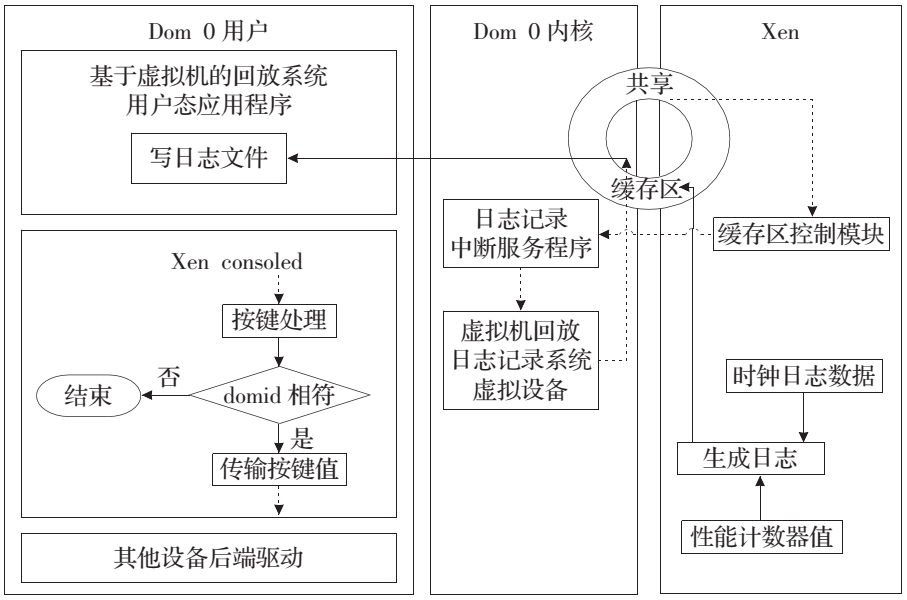

Fig.4 Data flow of log recording in VMRS

图 4 VMRS 日志记录模块数据传输示意图

间的重复拷贝,从而提高回放系统效率。日志回放模块在日志传输虚中断处理完毕后,每次从共享缓存区中读取一个性能计数器值并写入性能计数器;当性能计数器溢出时产生性能计数器中断 PMI,其中断服务程序中的回调函数向特权域内核发送虚中断,唤醒特权域内核中的中断模拟线程,将影子内存区域中的中断信息发送给相应设备的后端驱动,以完成回放的模拟工作。

3.2 回放时间点的确定

回放过程要按照偏序关系精确重现日志中记录的每一条不确定性事件,VMRS 采用 CPU 的性能计数器中断来定位此类事件的时间点。

由于性能计数器中断是一种非可屏蔽中断 NMI(non maskable interrupt),VMRS 正是通过设置自定义 NMI 回调函数来处理性能计数器中断的。而 X86 架构下 NMI 中断是由高级可编程中断控制器 APIC(advanced programmable interrupt controller)发送给 CPU 来处理的,所以必须先配置 APIC 使 CPU 可以接收到来自外部中断控制器的中断信号。性能计数器中断 PMI 发生在性能计数器溢出的时候,但是系统默认不产生这种中断,因此需要用 wrmsr 指令来设置性能计数配置控制寄存器 CCCR(counting configuration control registers),开启 CPU 对性能计数功能的支持。

为了确定不确定性事件的回放时间点,必须根据相邻两条日志记录的性能计数器的值差来确定这两次事件的发生间隔,将性能计数器设置为这个差值,这个值在经过计算的时间间隔后会发生溢出。如:当前后两次性能计数器的差值为1 000,则设置性能计数器的值为-1 000+1=-999,而后性能计数器的值自增 1 000 次时发生溢出。性能计数器的溢出会产生性能计数器中断 PMI 并发送给 APIC,而后 APIC 以 NMI 中断的形式发回 CPU,最后会调用处理 NMI 的接口函数进入 NMI 回调函数,回放模块的 NMI 回调函数 nmi_for_replay 通知特权域内核中的模拟中断程序进行日志的回放。以键盘事件的回放为例,只需要模拟键盘设备后端驱动的行为把键盘值写入设备缓冲区,向非特权域的前端设备驱动发送事件通道消息,那么非特权域就会认为发生了一次键盘中断,从而通过前端设备驱动将此次的键盘值取回,并显示在字符终端上,从而完成一次键盘事件的回放。

3.3 回放过程的实现

在 Xen 的体系构架下,客户操作系统通过虚拟的前端设备驱动和特权域操作系统中后端设备驱动来共同完成对真实设备的 I/O 操作。前端设备驱动与后端设备驱动之间包括一块共享内存区域和两个用来通信的事件通道端口号,这块内存区域的共享映射和事件通道端口号是在非特权域创建之时由虚拟机管理器 Xen 来建立的。

以键盘设备为例来说明后端设备驱动处理非特权域字符输入的流程。键盘前端设备驱动共享缓冲区结构包含两个字符数组缓存区和两对生产者、消费者指针:in 字符数组用来存放由后端驱动写入的字符数据;out 字符数组用来存放由非特权域发送向特权域后端驱动的字符数据;in_prod 和 out_cons 分别为特权域后端驱动的生产者和消费者指针;in_cons 和 out_prod 为非特权域前端驱动的消费者和生产者指针。当特权域后端设备驱动获取到按键值后,处理流程如图 5 所示。

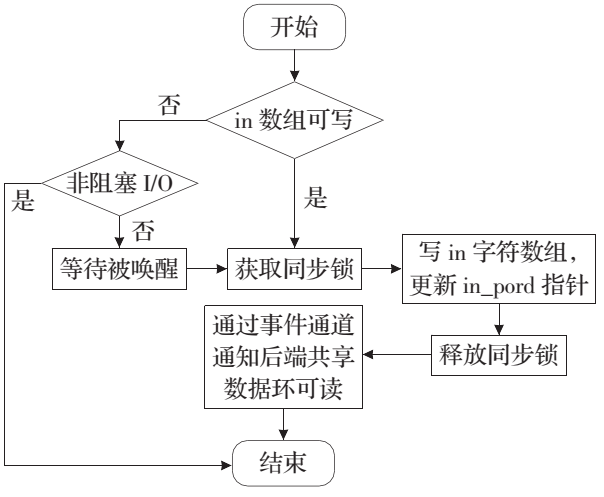

Fig.5 Process of char input at the backend device

图 5 后端驱动处理字符输入的流程图

首先检查 in 字符数组是否还有剩余空间来写入,若有则获取同步锁,而后将按键值写入 in 字符数组并更新 in_prod 指针,然后释放同步锁并以事件通道方式来通知后端驱动 in 字符数组可读。当前端驱动程序接收到特权域的事件通道消息后,首先获取同步锁,读 in 字符数组并更新 in_cons 指针,然后释放同步锁并通过事件通道消息通知特权域键盘值获取完毕,缓冲区可写。至此,外部输入的按键值经过后端驱动设备,前端驱动设备的传递,最终到达非特权域。

由以上的分析可知,若模拟键盘设备后端驱动的行为,即写缓冲区与发送事件通道消息,那么非特权域就会认为又一次发生了键盘中断,从而通过前端设备驱动将此键盘值取回。为此还必须获取非特权域共享内存区的地址,以及用于事件通道通信的端口号。在 Xen 系统中可以通过 Xenstore[8]来获取这两项信息。回放过程中,当特权域收到中断模拟的虚中断后,唤醒中断模拟线程首先将日志中的键盘值拷贝到特权域内核的影子缓存区内,然后将缓存区中的数据拷贝到共享内存区中,最后通过超级调用,向非特权域前端驱动设备发送键盘处理虚中断。至此,便实现了非特权域的键盘输入过程的模拟。

4 系统测试

VMRS 在 Xen3.3 环境下实现了对字符界面的 Linux 操作系统 Ubuntu8.04 的精确回放。然而对于虚拟机全系统的回放,不仅仅只关注系统的功能实现,还要衡量回放过程给原系统运行带来的性能开销。接下来,将对日志记录模块和日志回放模块的性能开销进行分别测试,并对测试结果进行对比与分析。

首先需要关注的是在日志记录过程中的日志增长速度。对于日常的桌面应用(主要为键盘操作),经过多次测试,VMRS 的平均日志增长速度为 0.97 MB/h,其对于一个拥有几十乃至几百 GB 存储容量的系统的磁盘空间消耗几乎是微乎其微的。

其次,需要关注的是日志记录过程中产生的系统开销。通过分别测试在有无日志记录模块运行时录入 10 个字符的时间消耗如图 6 所示:在未经修改的Xen 环境下,特权域后端设备驱动程序对每个键盘中断的平均处理时间约为 446 μs,VMRS 处理每一个按键中断的平均处理时间约为 565 μs,日志记录带来的额外开销约为 119 μs,由此得到一次按键记录所造成的额外开销约为 26%。

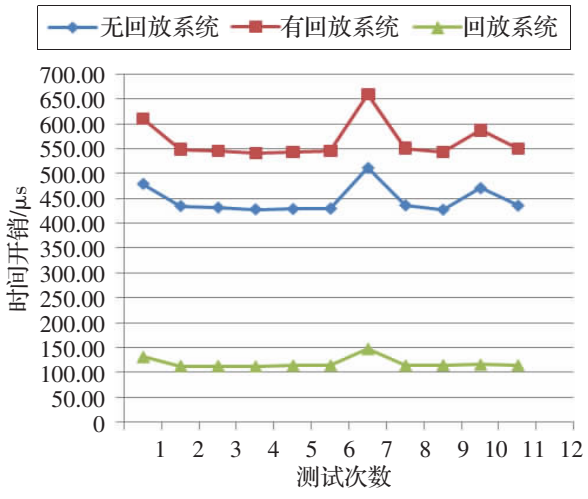

Fig.6 Performance overhead caused by event logging

图 6 日志记录过程的性能开销

对于 VMRS 回放过程的系统开销是通过对比中断模拟程序与 Xen 原始后端驱动程序处理每个键盘中断的平均时间开销来进行的。测试进行了 11 次,每次录入 20 个按键,计算每次的程序平均执行时间。测试结果如图 7 所示:中断模拟程序平均执行时间为 130.66 μs,Xen 原始后端驱动程序处理每个按键的平均时间开销为 445.86 μs。由此可见,回放时所执行的程序所需的中断数据总会及时到达,其回放过程的系统开销甚至远远小于非特权域处理键盘中断的系统开销。

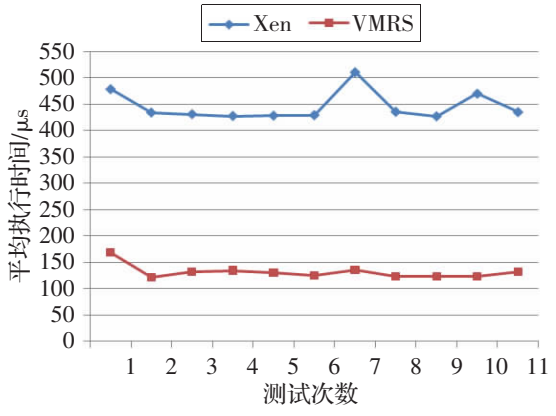

Fig.7 Time overhead of VMRS replaying and time caused by keybord operation in Xen

图 7 回放模拟程序与 Xen 对键盘处理的时间开销

5 结论

该文在 Xen VMM 的环境下设计并实现了基于半虚拟化设备模型的虚拟机回放系统 VMRS。系统具有以下特点:

(1)保持了客户操作系统的完整性:VMRS 能够对任意基于 Linux 的半虚拟化操作系统进行有效记录和回放,不需要修改客户操作系统的代码或添加模块程序。

(2)系统具有较好的可扩展性:目前 VMRS 只能回放字符界面的操作系统,但良好的接口设计只需要添加少量代码就可以支持其他的外部设备。

(3)系统性能高效:VMRS 只产生很小的日志记录和回放性能开销,并且记录日志消耗的存储空间非常之少。

虽然目前的研究已经支持完整系统的回放,但是还有一些亟待解决的问题:为了对具有图形化界面和网络连接的 Linux 进行回放,需要对鼠标、网络等其他中断事件进行支持;VMRS 目前只适用于半虚拟化的虚拟机,如何支持全虚拟化的硬件虚拟机尚需要更加深入的研究。

References:

[1] Geels D, Altekar G, Shenker S, et al. Replay debugging for distributed applications[C]//Proceedings of USENIX Annual Technical Conference (USENIX'06), Boston, MA, USA, May 30-June 3, 2006:289-300.

[2] Smith J E, Nair R. Virtual machines: Versatile platforms for systems and processes[M]. Morgan Kaufmann: Elsevier Inc, 2006:1-26.

[3] Dunlap G W, King S T, Cinar S, et al. ReVirt: Enabling intrusion analysis through virtual-machine logging and replay[C]//Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI'02), Boston, MA, USA, December 8-11, 2002. [S.l.]: ACM Press, 2002:211-224.

[4] Xu M, Malyugin V, Sheldon J, et al. ReTrace: Collecting execution trace with virtual machine deterministic replay[C]//Proceedings of the Third Annual Workshop on Modeling, Benchmarking and Simulation (MoBS'07), California, USA, June 10, 2007.

[5] de Oliveira D A S, Crandall J R, Wassermann G, et al. Ex-

ecRecorder: VM-based full-system replay for attack analysis and system recovery[C]//Proceedings of the First Workshop on Architectural and System Support for Improving Software Dependability (ASID'06), San Jose, California, USA, October 21, 2006. [S.l.]: ACM Press, 2006:66-71.

[6] Barham P, Dragovic B, Fraser K, et al. Xen and the art of virtualization[C]//Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP'03), Lake George, New York, USA, October 19-22, 2003. [S.l.]: ACM Press, 2003: 164-177.

[7] Intel Corporation. The IA-32 Intel architecture software developer's manual, Volume 3: System programming guide[R]. 2001:1861-1898.

[8] David C. The definitive guide to the Xen hypervisor[M]//Prentice Hall Open Source Software Development Series. Boston, MA, USA: Pearson Education, Inc, 2008.

LIU Haikun was born in 1981. He is a Ph.D. candidate at Huazhong University of Science and Technology. His research interests include virtualization technology for computing system and distributed computing.

刘海坤(1981-),男,湖北随州人,华中科技大学计算机学院计算机系统结构专业博士研究生,主要研究领域为计算系统虚拟化,分布式计算。

JIN Hai was born in1966. He received his B.S., M.S. and Ph.D. degrees in Computer Engineering from Huazhong University of Science and Technology (HUST) in 1988, 1991 and 1994, respectively. Now he is a professor and the Dean of School of Computer Science and Technology at HUST. He is the chief scientist of the 973 Project "Basic Theory and Methodology of Virtualization Technology for Computing System" and the Largest Grid Computing Project, ChinaGrid, in China. He is a senior member of CCF, IEEE, and member of ACM. He is the member of Grid Forum Steering Group (GFSG). His research interests include virtualization technology for computing system, cluster computing and grid computing, peer-to-peer computing, network storage and network security, etc.

金海(1966-),男,博士,华中科技大学教授、博士生导师,CCF 高级会员,华中科技大学计算机学院院长,国家973 计划"计算系统虚拟化基础理论与方法研究"首席科学家,教育部重大专项"中国教育科研网格 ChinaGrid"计划的专家组组长,湖北省计算机学会理事长。主要研究领域为计算系统虚拟化,集群与网格计算,对等计算,网络存储系统,网络安全等。

CUI Tao was born in 1983. He received his M.S. degree in Computer Science and Engineering from Huazhong University of Science and Technology in 2009. His research interest is virtualization technology for computing system.

崔涛(1983-),男,河北张家口人,2009 年于华中科技大学获得硕士学位,主要研究领域为计算系统虚拟化。

LIAO Xiaofei was born in 1978. He received his Ph.D. degree in Computer Science and Engineering from Huazhong University of Science and Technology in 2005. He is now an associate professor in School of Computer Science and Engineering at HUST. He is a member of the CCF and IEEE. His research interests include virtualization technology for computing system, peer-to-peer system, cluster computing and streaming services, etc.

廖小飞(1978-),男,湖北黄冈人,2005 年于华中科技大学获得博士学位,目前为华中科技大学副教授,CCF 会员,主要研究领域为计算系统虚拟化,对等计算,集群计算,流媒体服务等。