一种基于脏页面延迟拷贝的虚拟机动态内存迁移方法

张 伟 张晓霞 王汝传

(南京邮电大学计算机学院 南京 210046)

摘 要 通过对虚拟机动态内存迁移过程中内存页面状态的分析,针对预拷贝算法中脏页面可能被重传这一特点,提出一种基于脏页面预测的迭代传送方法。该方法基于时间局部性原理,在页面迭代传送前对页面未来的变脏程度进行预测,根据预测结果对页面进行状态分类,并对处于不同状态的页面采取相应的传送策略。实验证明,与Xen采取的动态迁移方法相比,提出的方法减少了近10%的迁移总量。

关键词 虚拟机,内存动态迁移,脏页面,预测

中图法分类号 TP393 **文献标识码** A

Live Memory Migration for Virtual Machine Based on Dirty Pages Delayed Copy Method

ZHANG Wei ZHANG Xiao-xia WANG Ru-chuan

(College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210046, China)

Abstract Through the analysis of the state of the memory pages in the live memory migration for virtual machine, a novel method was proposed to avoid the dirty pages to be retransmitted in the Pre-Copy algorithm. In this method, the next dirty extent of pages is predicted based on the principle of temporal locality before the pages' transmitting in the iteration stage. According to the result of prediction, the pages are classified into different states to take appropriate transmission strategy. Experiments show that the method has effectively reduced the total migration memory nearly 10% compared to the live migration approach taken by Xen.

Keywords Virtual machine, Live memory migration, Dirty pages, Prediction

1 引言

虚拟机迁移过程是指虚拟机在运行状态下从一个物理结点迁移到另一个物理结点的过程,虚拟机的迁移技术可以有效解决计算环境对硬件平台的依赖,是虚拟化领域的研究热点。虚拟机迁移过程分为静态迁移和动态迁移两类,静态迁移是在虚拟机暂停的情况下迁移所有状态数据,主要面向无实时性要求的应用,例如Collective系统[1]和Zap系统[2]。动态迁移则要求在迁移过程中虚拟机上的服务不能中断,用户对物理机器的切换没有感知,主要面向实时要求性强的应用,例如VMware虚拟机的VMotion机制[3]和Xen的Live-migration方案[4]。

动态迁移能够将整个虚拟机的运行状态完整、快速、无缝地在不同宿主机器之间迁移,并且在整个迁移过程中能够保证运行在其上的服务质量。因此,动态迁移广泛地用于数据中心、虚拟机群的管理、云计算平台等,以解决服务器之间的负载均衡、容错入侵以及节能管理等问题。动态迁移过程需要迁移的内容包括:内存映像、CPU状态和网络配置等,具体研究方向包括:内存迁移的研究、文件系统迁移的研究、跨域迁移的研究、迁移标准的确定和迁移过程的安全问题等。其中,内存状态迁移需要考虑操作系统内核的执行状态和应用程序的状态,涉及数据量大且动态变化,还涉及到内存状态的一致性和迁移效率等问题,如何提高内存迁移的效率一直是动态迁移的关键问题。本文基于内存页面访问的局部性特点,将预测思想引入内存拷贝算法,实现迁移性能的优化。

2 相关工作

目前国内外主流的虚拟机内存迁移的策略主要有3种:基于虚拟机系统事件记录与重放的方法、基于后拷贝的方法(Post-Copy)和基于预拷贝的方法(Pre-Copy)。

基于虚拟机系统事件记录与重放的方法通过在源虚拟机上迭代记录虚拟机执行过程中影响系统执行流程的事件,并在目的虚拟机上重放这些事件日志来同步源虚拟机和目的虚拟机的状态。Revirt系统[5]基于此技术实现虚拟机动态迁移过程,并通过对系统执行状态的动态监测发现可能的入侵行为。Jin H等人[6]基于Linux系统在此技术上做出改进,即采用检查点/恢复和记录/重放技术实现快速透明的虚拟机迁移,并采用按Copy-on-Write(COW)方式设置检查点。该方案可以大大减少迁移停机时间和网络带宽的消耗,但该技术发展并不成熟,没有被主流的虚拟机应用所采用。

基于Post-Copy的方法首先将所有的处理器状态发送到目标机器,开启目标虚拟机,再将虚拟机上发生执行错误的内

到稿日期:2012-07-10 返修日期:2012-10-28 本文受国家自然科学基金(61003236,61272422)资助。

张 伟(1973-),男,博士,副教授,主要研究方向为网络安全、逆向分析、数据流检测,E-mail:zhangw@njupt.edu.cn;张晓霞(1986-),女,硕士生,主要研究方向为虚拟化技术;王汝传(1943-),男,教授,博士生导师,主要研究方向为无线传感器网络、移动代理和虚拟现实技术。

存页面动态地从源虚拟机推到目标虚拟机。Michael R 等人[7]结合 Adaptive Pre-Paging 和 Dynamic Self-Ballooning 技术提出 Post-Copy 方法。Hirofuchi T[8]利用此迁移方法为 IaaS 数据中心实现虚拟机的动态整合，使虚拟机群集运行在较少的服务节点中，从而降低能耗。这种方案在迁移过程中很大程度上增加了对源虚拟机的依赖性，停机时间较长。

基于 Pre-Copy 的方法是先传送所有的内存页面到目标结点并在迁移过程中记录内存的修改状态，之后每轮仅传送上一轮修改过的页面，最后暂停虚拟机并传送修改特别频繁的页面及所需的网络状态。Clark C Fraser K 等人[4]提出的 Pre-Copy 方法已经被一些主流的虚拟机采用，如 Xen、VMware、KVM，该方法将停机效应分配到整个迁移过程中，整个系统停机时间显著减少，但是对脏页面的频繁重传将导致迁移时间的延长和网络吞吐量性能的下降，制约了预拷贝方法优势的发挥。虚拟机的动态迁移要求在虚拟机上运行的服务不能中断，因此虚拟机的每个内存页面都可能变脏，而动态迁移的主要应用场景的数据中心是内存读写和网络通信密集型操作环境，预拷贝迁移方法的脏页面反复重传会大量增加迭代拷贝的次数和每次拷贝传输的网络吞吐量。许多研究者在该算法的基础上提出了改进，Xen 虚拟机[9]采用 Xen 的影子页表(shadow pages)记录每个页面被修改的状态，将内存页面分为 3 类脏位图页面，在每轮迭代中仅选择上轮被修改且本轮未被修改的页面进行传送。Menon A 等人[10]提出在基于 Xen 虚拟机迁移过程中增加退出迭代拷贝的条件，提前进入 stop-and-copy 阶段，但这样又会延长停机时间，影响运行服务的质量。Jin H 等人[11]分析迁移内存页面的不同特征，并对迁移数据采用自适应的压缩算法，实现虚拟机的快速迁移，然而若当内存压缩算法选择不当时，可能会为迁移系统带来额外开销。

Pre-Copy 模型平衡了停机和整体迁移时间的矛盾，成为目前主流虚拟机采用的迁移模型。本文提出的优化模型基于 Pre-Copy 模型，针对内存操作的局部性特点，对可能变脏的内存页面进行预测并推迟传送，以提高整体迁移效率。本文第 3 节介绍迁移技术及预拷贝算法；第 4 节针对基本 Pre-Copy 模型提出基于脏页面预测及分类处理算法；第 5 节对该模型的参数和性能问题进行分析；第 6 节给出虚拟机内存迁移仿真实验；最后总结全文。

3 迁移技术介绍

3.1 内存迁移的过程

典型的虚拟机内存动态迁移过程由 3 个步骤组成[4]。

步骤 1　Push：页面从源虚拟机通过网络被 Push 到目的虚拟机，而源虚拟机运行不间断，其间被修改了的页面需要重传，该步骤会多次迭代。

步骤 2　Stop-and-Copy：迭代 Push 步骤中止后，冻结源虚拟机，页面通过最后一次迁移传送到目的结点，并启动新虚拟机。

步骤 3　Pull：新的虚拟机在目的机器上运行，若页面出现与原来虚拟机不一致的状态，则从源虚拟机处执行 Pull 操作来获取正确页面。

目前大部分的迁移策略只包含上述的一个或者两个步骤。例如，单纯 Stop-and-Copy 就是虚拟机的静态迁移；将 Stop-and-Copy 和 Pull 结合是 Post-Copy 迁移方案；Push 和 Stop-and-Copy 结合是第 3 种内存迁移方案，即预拷贝方法。

虚拟机内存的动态迁移需要考虑内存状态的一致性和迁移效率问题，迁移效率最重要的指标是虚拟机停机时间和整个系统的迁移时间。停机时间是指用户停止执行的时间段；总迁移时间是指从迁移开始到迁移的虚拟机在目的宿主主机开始运行的时间。要提高迁移效率，就需要尽可能地减少停机时间和迁移时间，这是用户可以感受到的，其不宜超出用户能够容忍的范围。

$$T_{total}=t_{preparation}+t_{iteration}+t_{down} \tag{1}$$

式中，$t_{preparation}$ 是源虚拟机发起迁移信号和目的虚拟机预留资源过程需要的时间；$t_{iteration}$ 包括第一次拷贝所有的内存页面到目的端和每轮迭代拷贝上一轮的变脏的页面的累计时间；t_{down} 是指挂起虚拟机，将不一致的页面一次性地拷贝到目的虚拟机需要的时间。要缩短总的迁移时间，$t_{iteration}$ 相对固定，就要降低从 $t_{iteration}$ 和 t_{down} 入手。

$$M_{total}=m_0+\sum_{i=1}^{n}m_{dirty} \tag{2}$$

式中，所有要传送的内存页面包括第一轮发送的在源虚拟机所有页面的最初状态 m_0 和以后每轮要迭代拷贝的脏页面。其中，每轮迭代要重传的脏页面是导致整个内存页面传送量增大的重叠因素，为了减少迁移过程中的传送量，应该考虑减少脏页面的重传数量。

3.2 预拷贝算法

基于预拷贝算法的虚拟机内存迁移流程如图 1 所示。完成第 1 阶段迁移工作前的准备工作之后进入第 2 阶段的迭代拷贝过程，第 1 轮迭代拷贝所有的内存页面，第 2 轮迭代只拷贝在第 1 轮迭代过程中修改过的页面，即变脏页面。以此类推，第 n 轮拷贝的是在第 $n-1$ 轮迭代过程之后修改过的页面。当迭代终止条件满足，如迭代到达一定次数时，进入第 3 阶段，即停机阶段，此时停机并把剩下的脏页面以及运行状态等信息一次性拷贝过去。

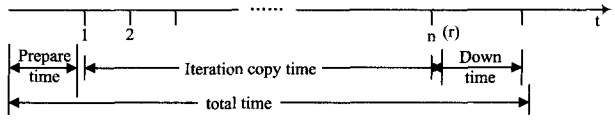


图 1　Pre-Copy 算法 3 个阶段

Xen 虚拟机动态迁移采用了附带简单预测能力的预拷贝方法，其采用 Xen 的影子页表(shadow pages)记录被修改的页面，将内存页面分为 3 类脏位图页面：to_skip 记录本次迭代中可以跳过不传的页面，to_send 记录在上次迭代中出现的脏页，to_fix 记录在最后停机拷贝才传送的页面。在每轮迭代结束之前，将脏页面位图拷贝到 to_send 中并清空脏页位图。在每轮迭代开始后，将脏页面位图拷贝到 to_skip 中，由 to_skip 和 to_send 页位图中的页面对应比特位共同决定是否发送该页面。其判定规则如表 1 所列，(1)当 to_send 位且 to_skip 位均为 0 时，表示这两轮迭代内存页面没有被修改，因此该页面不传送；(2)当 to_send 位为 0 且 to_skip 位为 1 时，表示这两轮迭代中内存页面虽有变脏，但可能继续变脏，因此该页面不传送；(3)当 to_send 位为 1 且 to_skip 位为 0 时，表示这两轮迭代中内存页面变脏的过程已结束，需要传送该页面；(4)当 to_send 位且 to_skip 位均为 1 时，表示在这两轮迭代中页面变脏较为频繁，因此该页面不传送。

表 1　Xen 内存迁移决策规则

To_send	0	0	1	1
To_skip	0	1	0	1
Send or not	N	N	Y	N

4　基于访问局部性预测的迭代传送方法

4.1　内存迁移系统描述

定义 1　虚拟机内存中一个页面为 p_i，则虚拟机内存中所有的页面集合 $P=\{p_1,p_2,\cdots,p_m\}$。

定义 2　D 为一个用于预测的数据结构，以迭代传送过程中的页面状态存放，以便实现对这些页面操作的支持。

定义 3　设每轮迭代传送的页面量为 m_i，整个内存迁移过程的页面传送量集合 $M=\sum_{i=1}^{d}m_i$，共 d 次迭代，其中迭代拷贝阶段页面重传的总量为 M_d，$M_d\subset M$。设总的页面迁移时间为 $T_{total}=t_{preparation}+t_{iteration}+t_{down}$，$t_{down}$ 为停机拷贝的时间。

定义 4　内存迁移度量 α 为页面重传比，$\alpha=M_d/M$，β 为停机时间比，$\beta=t_{down}/T_{total}$。

定义 5　一个内存迁移系统 $S=\{P,D,M,T,\alpha,\beta\}$，其中 P 如定义 1 所示，是系统的输入，提供一个具体调用序列；D 如定义 2 所示，是中间辅助结构。

本文设计的优化内存迁移系统如定义 5 所示，对要迁移的内存页面不同状态进行分析并相应传送，使内存迁移度量标准 α 和 β 的值尽量减小。假设一个以 α 和 β 为参数的目标函数 $f(\alpha,\beta)$ 为综合优化指标，在保证最终成功迁移的前提下，我们以得到 Min $f(\alpha,\beta)$ 的系统 S 为目标。

4.2　内存迁移过程

本文提出的虚拟机内存动态迁移方法基于 Pre-Copy，包含 Push 和 Stop-and-Copy 两个步骤，也需要迭代重传变脏的页面和停机过程，但对脏页面重传采用了优化措施。优化方法根据内存页面被修改的历史记录，及时预测各个页面在下轮迭代中可能变脏的程度，并针对不同页面采取不同的传送策略，以实现优化目的。

基于脏页面延迟迁移的内存迁移分为 4 个过程，如图 2 所示。

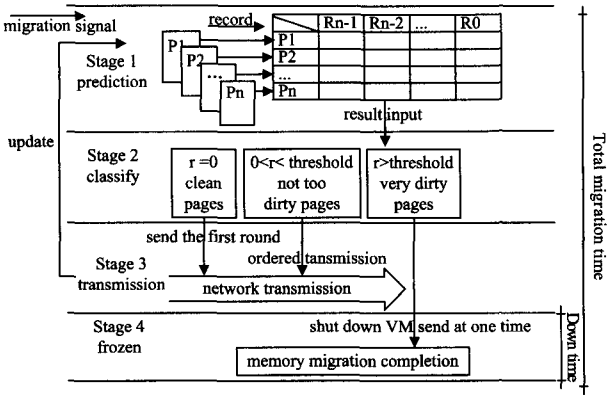


图 2　内存迁移优化模型

第 1 个过程对虚拟机中所有的内存页面进行监控，并为每个页面设立寄存器向量记录其被访问的情况；第 2 个过程根据寄存器记录的结果对页面进行计算和分类；在第 3 个过程，针对不同类别的页面采取分步传送策略，并更新寄存器数据；然后，判断迭代终止条件，如果不满足终止条件，则返回第 1 个过程，否则，进入第 4 个过程，暂停虚拟机，并传送剩余页面。

4.3　基于访问局部性的移位寄存器

页面置换算法中的最近最少使用算法(Least Recently Used，LRU)考虑到内存页面被使用的情况，最近被访问过的页面以更大的概率在接下来时间内被访问到，因此我们可以根据页面过去的访问记录来预测页面将来可能变脏的概率，并设计了移位寄存器来实现。

定义 6　单个页面的 n 位寄存器序列 $c_i=(R_{n-1},R_{n-2},\cdots,R_1,R_0)$，$X_i$ 为 R_i 的值，$X_i=0$ 或 $X_i=1$，寄存器序列值 $r_i=\sum_{i=0}^{n-1}X_i$。

定义 7　内存页面 n 位寄存器集合 $C=\{c_1,c_2,\cdots,c_m\}$，且寄存器值的集合 $r=\{r_1,r_2,\cdots,r_m\}$，m 为页面总数。

定义 8　对寄存器 C 的操作 $O_p(C)=\{O_{p1}(C),O_{p2}(C)\}$，$O_{p1}(C)=\begin{cases}\text{Set } R_{n-1}=1, & c_i\in C\\ \text{do nothing}, & c_j\in C,i\neq j\end{cases}$，$O_{p2}(C)=\begin{cases}\text{Shift Right Logical}, & c_i\in C\\ \text{Shift Right Logical } c_j\in C,i\neq j\end{cases}$，$\forall R_i\in C$ 都满足 $\sum_{i=1}^{m}X_i=1$。

定义 9　设内存迁移过程中单个页面 p_i 的访问次数为 m_i，其访问率为 $\gamma_i=m_i/\Delta t$，则整个页面集合 $P=\{p_1,p_2,\cdots,p_n\}$ 的访问率为 $\gamma=\{\gamma_1,\gamma_2,\cdots,\gamma_n\}$，该过程页面集合的平均访问率 $\bar{\gamma}=\frac{1}{n}\sum_{i=0}^{n-1}\gamma_i=\frac{1}{n}\sum_{i=0}^{n-1}\frac{m_i}{\Delta t}$。

定理 1　在一个时间相对较短的页面调用序列中(相对于全局调用序列)，某些特定页面的访问率会高于平均访问率，即页面具有访问局域性。

证明：在预测方案中，首先为每个页面设立一个移位寄存器，如定义 6 所示，因此内存中存在一个所有页面的寄存器集合，如定义 7 所示。当某个页面被访问时，寄存器执行定义 8 的操作。在自然情况下，页面被访问的频率是随机分布的，在一次迁移过程中，每个页面的访问率应近似等于 $\bar{\gamma}$。而由内存访问的局部性特性可知，在虚拟机内存实时迁移中，某些特定页面被访问的次数 m_i 会很大，其访问率 γ_i 可能会大于 $\bar{\gamma}$。证毕。

本文设计的内存迁移系统 $S=\{P,D,M,T,\alpha,\lambda\}$ 中，对高访问率页面的反复重传会增大迭代拷贝阶段页面重传的总量 M_d，从而增大内存页面重传比 α，降低内存迁移的整体效率。在迭代拷贝之前预测高访问率页面并采取相应的传送策略，能够提高内存迁移系统 S 的效率。

4.4　内存页面的状态迁移

动态内存迁移要求运行在其上的服务是不能中断的，因此内存中每个页面都有可能变脏。而迁移关注的仅是在迭代拷贝过程结束前页面的最终状态。为了避免脏页面中间状态大量重传，我们使用一个有穷状态机来分析内存页面的状态迁移。

定义 10(有穷状态机，Finite automaton，FA)　M 是一个五元组，$M=(Q,\Sigma,\delta,q_0,F)$。

式中，Q 为内存页面状态在迁移过程中的非空有穷集合，$\forall q\in Q$，q 为某页面的一个状态；Σ 为触发页面产生某种状态变化的页面访问；δ 为页面状态发生转移的条件函数，$\delta:Q\times\Sigma\to Q$；$q_0$ 为内存页面的开始状态；F 为内存页面的终止状态集合，$F\subseteq Q$，$\forall q\in F$，q 为某页面的一个终止状态。

由定义 10 可知，在内存迁移的迭代拷贝过程中，内存页

面的状态 $Q=\{q_0,q_1,q_2,q_3,q_4\}$，$q_0=\langle unsend,clean\rangle$，$q_1=\langle unsend,relative\ dirty\rangle$，$q_2=\langle unsend,more\ dirty\rangle$，$q_3=\langle send,clean\rangle$，$q_4=\langle send,relative\ dirty\rangle$，$\Sigma=\{a,b,c,d\}$，$\delta=\{\delta_1,\delta_2,\delta_3,\delta_4,\delta_5,\delta_6\}$，$F=\{q_2\}$。

内存页面在迭代拷贝过程中的状态迁移变化如图 3 所示，共有以下 6 种情况：(1)$\delta_1\langle q_0,a\rangle=q_3$，在该过程中未被修改的页面；(2)$\delta_2\langle q_3,b\rangle=q_1$，已经传送过的干净页面变脏，但 $r<r_t$，r 为页面寄存器值，r_t 是进入最后冻结阶段拷贝的页面阈值，仍要被传送；(3)$\delta_3\langle q_3,c\rangle=q_2$，已经传送过的脏页面变脏，但由于 $r\geqslant r_t$，等待虚拟机停机再被传送；(4)$\delta_4\langle q_1,d\rangle=q_4$，还未传的较脏页面，根据历史统计值 $r<r_t$，要被传送到目的端；(5)$\delta_5\langle q_4,b\rangle=q_1$，指(4)中传送过的较脏页面又变脏了，下轮迭代继续传送；(6)$\delta_6\langle q_4,c\rangle=q_2$，指(4)中传送过的较脏页面变得很脏，下轮不传送，待到停机阶段再传送。

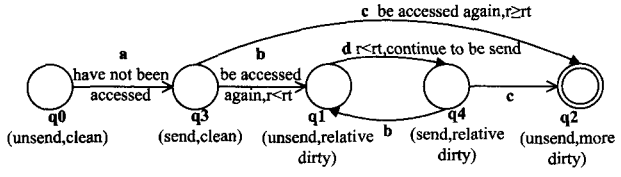


图 3　内存页面状态迁移图

4.5　多阶段迭代传送

若选择顺序传送内存页面，必然会引起不必要的重传。多阶段传送策略如下：在源虚拟机发起迁移信号后进入预测模块，根据页面寄存器的数值设定进入冻结阶段才传送的阈值 r_t。选出干净页面($r=0$)进行第一轮传送。对较脏页面($r<r_t$)以从小到大的顺序依次传送，进入下一轮迭代，并更新预测模块中寄存器的值。最后很脏的页面($r\geqslant r_t$)进入停机传送过程后再传送。当内存页面传送完毕，将网络重定向到目的虚拟机，源和目的虚拟机的状态达到一致时，迁移完成。

迭代过程的伪代码如下：

```
If the iteration stopping condition is fulled
    Exit iteration stage and skip to the frozen stage
Else
    For each page
        Check and compute the registers value r
    If(r>0 && r < threshold)
        Transfer the page in this round
    Update registers of all pages
Endif
```

5　算法参数设置

本文采用了基于访问局部性预测方法，并用移位寄存器来实现，根据定理 1 可知，迁移过程中仅需要很少的寄存器位数。该方法中寄存器的移位是随着程序访问新页而进行的，因为程序执行过程中，每个页面的执行时间是由计算机硬件和程序结构决定的，如果移位 Clock 设置不当，会影响执行效果。改进后的方法实现简单，移位 Clock 与页面的执行时间成正比。

在对内存页面传送中，阈值 r_t 大小的设置关系到迭代次数及停机时间，如果设置得太大，会增加总的虚拟机迁移时间；若设置得太小，则可能增加冻结模块的虚拟机停机时间。因此实验过程中应兼顾二者，这个值阈值的设置灵活考虑。r 的值代表寄存器中“1”的个数，而 r_t 代表被访问较为频繁页面的阈值，本文设定 r_t 在 $\bar{\gamma}<r_t<n$ 这个范围内，其中 $\bar{\gamma}$ 为页面访问过程平均访问率，n 为寄存器的位数。

参考 Xen 虚拟机实时迁移退出迭代拷贝的条件并结合本文设计的寄存器，我们考虑 3 个迭代终止条件：(1)本轮迭代满足整体迭代的收敛性且对网络造成的负担已达到最大值；(2)超出设定最大的迭代次数；(3)发送的总页数大于 Guest OS 的最大页数。满足以上 3 个条件中的任意一个，进入冻结阶段，完成一次性拷贝剩余页面到目的端。

6　仿真实验

为了验证本文所提方法的有效性，我们选择 3 种算法：传统预拷贝的算法(Pre-Copy)、Xen 虚拟机采用的脏页位图算法和本文提出的基于移位寄存器的统计预测算法，对页面访问随机分布和局部性分布两种测试进行比较。仿真实验分为 4 项，第 1 项实验测试 3 种算法在总页面数固定、局部性程度固定时的性能；第 2 项实验测试总的待迁移页面数目对算法性能的影响；第 3 项实验测试待迁移页面的访问局部性程度对性能的影响；第 4 项实验测试寄存器大小与门限设置对算法性能的影响。为了精确模拟迁移过程中页面访问逐步减少的特征，即页面整体的变脏率是逐渐递减的，实验假设在每次观察页面的传送量时，页面变脏率从 50%逐次以 5%的幅度递减。

第 1 项实验测试 3 种算法的效率，迁移内存 256M，每个页面为 4k，页面数为 65536，移位寄存器位数设置为 8，其间如果页面修改次数大于 3，即认为属于频繁访问集合，需要延时传输。迁移迭代 10 次，比较页面拷贝总量，分为随机分布和局部性分布两种情况，局部性度量为 0.2，即 80%的访问量集中在 20%的页面上。

由图 4、图 5 可知，本文改进后的方法在迁移过程中所要传送总的内存页面量比传统的预拷贝方法好很多，与 Xen 虚拟机采用的脏页位图算法相比，在第 5 轮迭代，Xen 传送的页面数目为 9047，而改进后方法仅需 7189，有明显的降低，该对比在图 5 的局域性分布下，Xen 传送的页面为 4146，而改进后方法仅需 2455，效果更明显；且两个图中，Xen 传送的总页面曲线在往后的迭代中还有上升的趋势，而本文提出的方法每轮传送的页面量较均等且逐渐减少，相应的曲线较平稳。

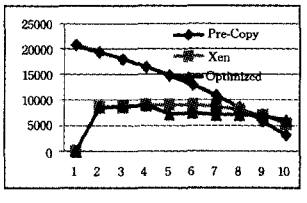


图 4　随机访问中 3 种算法不同阶段传输页面数

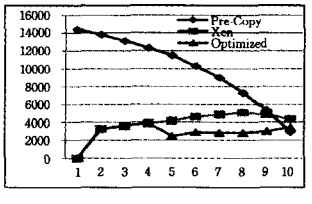


图 5　局部性访问中 3 种算法不同阶段传输页面数

第 2 项实验测试总的待迁移页面数目对算法性能的影响。我们在迁移页面总数变化的情况下测试 3 种算法迭代过程中的页面重传比 α 和停机时间比 β，本实验将停机时间比 β 转换成停机后仍未传送的页面与总页面的比值，如定义 4 所示，该数值越大，说明效率越低。

如图 6、图 7 所示，本项实验改变总的待迁移页面数目依次为 65536/4、65536、65536 * 2。在这 3 种情况下，比较 3 种

算法的内存迁移的度量标准 α 和 β，其中横坐标的 1,3,5 是比较 α。从实验结果可以看出，总迁移页面量越大，度量 α 的值越小，迁移效率越高。随机分布的第 3 种页面情况如横坐标 5 所示，本文的优化方法的 α 值为 0.12，相对预拷贝方法的 0.55和 Xen 方法的 0.2 较低，说明迁移效率较高，其他 2 种页面情况也是如此；在局域性分布的第 3 种页面情况如横坐标 6 所示，本文的优化方法的 β 值为 0.03，相对预拷贝方法和 Xen 方法的 0.01 较高，说明停机时间较长。

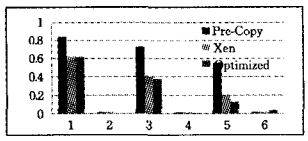


图 6 随机访问中 3 种算法的页面重传比 α 和停机时间比 β

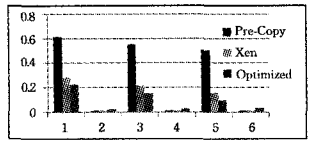
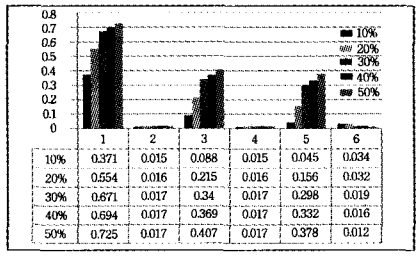


图 7 局域性访问中 3 种算法页面重传比 α 和停机时间比 β

第 3 项实验测试待迁移页面的访问局部性程度对迁移性能的影响，页面数为 65536，移位寄存器位数设置为 8，进入停机阶段的阈值设定为 3，迭代次数为 10，局部性度量从 0.1 到 0.5，如果局部性度量为 0.5，退化为随机均匀分布。

如图 8 所示，我们改变局部性度量从 0.1 到 0.5，在图中的排列自左向右。在这 3 种情况下，比较 3 种算法的内存迁移的度量标准 α 和 β，其中横坐标的 1、3、5 是比较 α。由实验结果可看出局部性越小，即访问越集中在某些页面的情况下，α 值越小；对于度量 β 的值，可以从横坐标 6 观察到局部性越大，β 值越小。

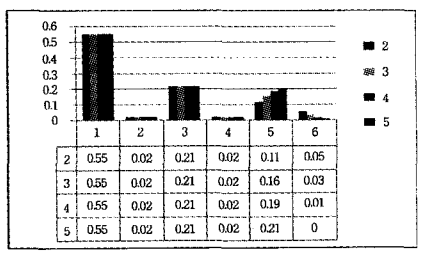


	1	2	3	4	5	6
10%	0.371	0.015	0.088	0.015	0.045	0.034
20%	0.554	0.016	0.215	0.016	0.156	0.032
30%	0.671	0.017	0.34	0.017	0.298	0.019
40%	0.694	0.017	0.369	0.017	0.332	0.016
50%	0.725	0.017	0.407	0.017	0.378	0.012

图 8 不同偏斜度局部性访问中 3 种算法的页面重传比 α 和停机时间比 β

第 4 项实验测试较脏页面阈值的设置对算法性能的影响，页面数为 65536，迭代次数为 10，局部性度量为 0.2，移位寄存器位数设置为 8，较脏页面的阈值从 2 逐步上升到 5。

如图 9 所示，我们将较脏页面阈值从 2 变化到 5，在图中从左到右依次变化，横坐标的 1,3,5 是比较度量标准 α。由横坐标 5 可知，脏页面阈值越小，α 值越小；由横坐标 6 可知，脏页面阈值越大，β 值越小。



	1	2	3	4	5	6
2	0.55	0.02	0.21	0.02	0.11	0.05
3	0.55	0.02	0.21	0.02	0.16	0.03
4	0.55	0.02	0.21	0.02	0.19	0.01
5	0.55	0.02	0.21	0.02	0.21	0

图 9 不同传输阈值局部性访问中 3 种算法的页面重传比 α 和停机时间比 β

由上述 4 项实验对比可知，本文提出的基于移位寄存器的统计预测算法比传统预拷贝的算法的各项度量标准都要强很多，而与 Xen 虚拟机采用的脏页位图算法相比，本文的优化方法在总的页面传送量和页面重传比率 α 上要强一些，而停机时间比率 β 相对弱一些。由后 2 项实验可知，本文提出的算法所涉及的相关参数也显得尤为重要，迁移效率与局部性分布的局部性度量及脏页面的阈值密切相关，局部性度量及脏页面的阈值越小，对页面重传比率 α 越有利，但不利于停机时间比率 β，设定阈值参数时应综合考虑二者的优劣。

结束语 本文针对 Pre-Copy 算法中对脏页面反复重传的这一关键点，提出一种优化的内存迁移方案，即根据基于时间局部性原理的移位寄存器对可能成为脏页面的页面进行预测，并根据预测值对页面在迁移过程中不同状态进行分析，对处于不同状态的页面采取相应的传送策略。实验证明该方案在使用占用空间有限的寄存器后，有效减少了迁移时要拷贝的内存页面数量，特别是对于读写密集型的内存页面。下一步工作中将考虑对每轮传送的页面采取适当压缩，以减轻每轮传送的网络负担。

参 考 文 献

[1] Chandra R, Zeldvovich N, Sapuntzakis C, et al. The collective: a Cache-based system management architecture[C]// Proc. of the Second USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI 2005). Boston, MA, 2005: 259-272

[2] Osman S, Subhraveti D, Su G, et al. The Design and Implementation of Zap: A System for Migrating Computing Environments [C]// Proc. of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002). Boston, MA, December 2002

[3] Nelson M, Lim B H, Hutchins G. Fast transparent migration for virtual machines [C]// Proc. of the 2005 USENIX Annual Technical Conference. VMware Press, 2005: 391-394

[4] Clark C, Fraser K, Hand S. Live Migration of Virtual Machines [C]// Proc. of the 2nd Conference on Symposium on Networked Systems Design and Implementation. Berkeley: USENIX Association, 2005: 273-286

[5] Tamura Y, Sato K, Kihara S, et al. Kemari: virtual machine synchronization for fault tolerance[C]// Xen Summit. Boston: USENIX Poster Session, 2008

[6] Liu H K, Jin H, Liao X F, et al. Live Migration of virtual machine based on full system trace and replay[C]// ACM International Symposium on High Performance Distributed Computing. 2009: 101-110

[7] Hines M R, Gopalan K. Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning [C]// Proc. of the ACM/ Usenix International Conference on Virtual Execution Environments (VEE'09). 2009: 51-60

[8] Hirofuchi T, Nakada H, Itoh S, et al. Reactive consolidation of virtual machines enabled by postcopy live migration[C]// Proc. of the 5th International Workshop on Virtualization Technologies in Distributed Computing. 2011: 11-18

[9] Barham P, Dragovic B, Fraser K, et al. Xen and the art of virtualization[C]// Bolton Landing: Proceedings of the 19th ACM Symposium on Operating System Principles. 2003: 164-177

(下转第 135 页)

$ID_n^*\}$、门限值 t、消息 m^* 以及门限环签密接收者身份为 ID_R^* 下的伪造门限环签密 C^*。如果在整个过程中算法 B 没有失败退出，那么算法 B 检查下列条件是否成立：

①$F(ID_i^*)=0 \bmod p$ 对于所有的 $i\in(1,\cdots,n)$ 都成立；

②$K(M^*)=0 \bmod p$，其中 $M^*=H_m(L,m^*)$。

如果上述条件不同时成立，那么算法 B 将失败退出；否则，B 可计算：

$$\left(\frac{\sigma_4^*}{(\sigma_5^*)^{L(M^*)}}\right)^{1/t}=\left(\frac{g_2^{u+\sum_{i=1}^{n}r_i(U_i)}(m'\prod_{i\in\mathcal{M}}m_i)^{r_m^*}}{g^{r_m^*\cdot L(M^*)}}\right)^{1/t}$$

$$=(g_2^{ai})^{1/t}=g_2^{a}=g^{ab}$$

这就是 CDH 问题的解。

因此，如果存在一个敌手能够以不可忽略的概率伪造一个有效的门限环签密，那么就存在一个有效的算法能够以不可忽略的概率解决 CDH 问题，而这与 CDH 问题是一个困难问题相矛盾，故方案是 EUF-IDTRSC-CMIA 安全的。

3.4 方案效率

相对于群元素的点乘运算和指数运算等运算而言，双线性对计算所花费的计算成本较高，故这里只考虑双线性对运算的计算量。在本方案中，因 $z_1=e(g_1,g_2)$ 和 $z_2=e(g,g_2)$ 可以进行预计算，所以在对方案的运算量进行计算时，忽略这部分的计算量。通过对方案进行分析可知，在门限环签密产生阶段不需要双线性对计算。而在解签密阶段仅需要 3 次双线性对计算，虽然文献[15]也是在解签密阶段需要 3 次双线性对计算，然而该方案是基于随机预言模型的，相对于该方案而言，本文所提方案具有更高的安全性。

结束语 本文利用秘密共享技术提出了一种门限环签密方案，而现有基于身份环签密方案的安全性大多是在随机模型下证明的，同时对于门限环签密方案的研究也不多。本文在标准模型下设计了一个基于身份的门限环签密方案，通过对方案的安全性进行证明，指出方案在 DBDH 和 CDH 困难问题的假设下满足适应性选择密文攻击下的不可区分性以及适应性选择消息和身份攻击下的不可伪造性，因此，本文所提方案是安全可靠的。

参 考 文 献

[1] Shamir A. Identity-based cryptosystems and signature schemes [C]// Proceedings of Crypto 1984. volume 196 of LNCS, 1984: 47-53

[2] Boneh D, Franklin M. Identity-based encryption from the Weil pairing[C] // Proceedings of Crypto 2001. volume 2139 of LNCS, 2001: 213-229

[3] Florian Hess. Efficient identity based signature schemes based on pairings[C] // Proceedings of SAC 2002. volume 2595 of LNCS, 2002: 310-324

[4] Paterson K G, Schuldt J C N. Efficient identity-based signatures secure in the standard model[C]// Proceedings of ACISP 2006. volume 4058 of LNCS, 2006: 207-222

[5] Zheng Yu-liang. Digital signcryption or how to achieve cost(signature & encryption)≪cost(signature)+cost(encryption) [C]// Advances in Cryptology-Crypto 1997. volume 1294 of LNCS, Springer-Verlag, 1997: 165-179

[6] Duan S S, Cao Z F, Lu R X. Robust id-based threshold signcryption scheme from pairings[C]// Proceedings of the 3rd International Conference on Information Security. ACM, 2004: 33-37

[7] Peng C G, Li X. An identity-based threshold signcryption scheme with semantic security[C]// Proceedings of CIS 2005. volume 3802 of LNCS, Springer-Verlag, 2005: 173-179

[8] Li F G, Yu Y. An efficient and provably secure id-based threshold signcryption scheme[C]// Proceedings of ICCCAS 2008. IEEE Press, 2008: 488-492

[9] Zhu Z C, Zhang Y Q, Wang F J. The analysis of an efficient and provably secure id-based threshold signcryption scheme and its secure version [C] // Proceedings of the Second International Conference on Provable Security. volume 5324 of LNCS, Springer Verlag, 2008: 210-225

[10] Huang Xin-yi, Susilo W, Mu Yi, et al. Identity-based ring signcryption schemes: cryptographic primitives for preserving privacy and authenticity in the ubiquitous world[C]// Proceedings of the 19th International Conference on Advanced Information Networking and Application 2005. volume 2, 2005: 649-654

[11] Zhang M W, Yang B, Zhu S L, et al. Efficient secret authenticatable anonymous signcryption scheme with identity privacy[C]// Proceedings of IEEE ISI 2008. volume 5075 of LNCS, Springer-Verlag, 2008: 126-137

[12] Zhun L J, Zhang F T. Efficient id-based ring signature and ring signcryption schemes [C] // Proceedings of CIS 2008. IEEE Press, 2008: 303-307

[13] Zhu Z C, Zhang Y Q, Wang F J. An efficient and provable secure identity-based ring signcryption scheme[J]. Computer Standards & Interfaces, 2009, 31: 1092-1097

[14] Sharmila Deva Selvi S, Sree Vivek S, Pandu Rangan C. On the security of identity based ring signcryption schemes [C]// Proceedings of the 12th International Conference on Information Security. volume 5735 of LNCS, Springer-Verlag, 2009: 310-325

[15] 罗大文，何明星，李琥. 基于身份的门限环签密方案[J]. 计算机工程与应用，2011，47(33)：65-67

(上接第 130 页)

[10] Menon A, Santos J R, Turner Y, et al. Diagnosing Performance Overheads in the Xen Virtual Machine Environment[C]// Proc. of the First ACM/USENIX International Conference on Virual Execution Environments. 2005: 13-23

[11] Jin H, Deng L, Wu S, et al. Live virtual machine migration with adaptive memory compression[C]// Proc. of the 2009 IEEE International Conference on Cluster Computing (Cluster 2009). 2009

[12] Zhao W M, Wang Z L, Luo Y W. Dynamic Memory Balancing for Virtual Machines[C]// Proc. of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments. 2009: 37-47

[13] Du Y Y, Yu H L, Shi G G, et al. Microwiper: Efficient Memory Propagation in Live Migration of Virtual Machines[C]// Proc. of the 2010 39th International Conference on Parallel Processing. 2010: 141-149

[14] 陈阳，怀进鹏，胡春明. 基于内存混合复制方式的虚拟机在线迁移机制[J]. 计算机学报，2011，34(12)：2279-2291