第 38 卷 第 4 期
Vol.38 No.4

计 算 机 工 程
Computer Engineering

2012 年 2 月
February 2012

• 软件技术与数据库 •　　文章编号：1000—3428(2012)04—0070—03　　文献标识码：A　　中图分类号：N945

# 面向安腾架构的分层内存故障注入方法

王 波，左德承，钱 军，张 展

(哈尔滨工业大学计算机科学与技术学院，哈尔滨 150001)

**摘 要：**为研究内存故障对高可用服务器的影响，针对安腾架构的计算机提出一种多层次的内存故障注入方法，设计并实现一种新的故障注入器(HMFI)，通过在物理层、操作系统内核层和进程层注入内存故障，考察目标系统对内存故障的容错能力。实验结果表明，HMFI 注入的内存故障能够有效验证与分析复杂计算机系统的容错性能。

**关键词：**故障注入；安腾架构；容错机制；内存故障；评测

# Hierarchical Memory Fault-injection Method for Itanium Architecture

WANG Bo, ZUO De-cheng, QIAN Jun, ZHANG Zhan

(School of Computer Science & Technology, Harbin Institute of Technology, Harbin 150001, China)

**【Abstract】**In order to figure out how memory faults affect high-performance servers, this paper presents a multi-layer fault injection method for Itanium architecture computers. HMFI(Hierarchical Memory Fault Injector) is designed and developed to inject memory faults in physical level, operating system kernel level and process level. Experimental results demonstrate the effectiveness of HMFI in evaluating the fault-tolerant properties of complex computer system.

**【Key words】**fault-injection; Itanium architecture; fault-tolerant mechanism; memory fault; evaluation

**DOI:** 10.3969/j.issn.1000-3428.2012.04.023

## 1 概述

内存故障是计算机系统中最常见的故障之一。在可用性要求极高的关键系统中，需要评测其对内存故障的容错能力。故障注入是评测容错机制最有效的办法。目前的内存故障注入技术主要针对嵌入式系统，注入层次也比较单一，不适用于安腾架构等高可用计算机。本文针对安腾架构计算机的特点，提出多层次故障注入的方法，通过带外监控等多种手段监控系统状态，获取故障日志，实现自动化程度高，覆盖层面广的内存故障注入。

## 2 多层次内存故障注入方法

图 1 给出了容错机制层次。从容错的观点看，系统可以抽象为应用层、容错机制层和基础系统层。容错机制层检测并阻止基础系统层中发生的故障，从而保证应用层能够可靠地运行。对于不同的容错机制，基础系统层和应用层的划分也是不同的，因此需要在多个层次注入内存故障，验证不同的容错机制的有效性[1]。

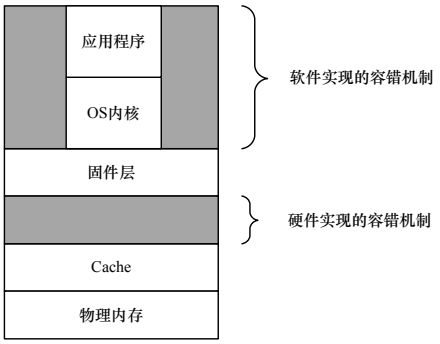


**图 1 容错机制层次**

本文在物理层、内核层和进程层分别进行故障注入，故障类型分瞬时故障和永久故障，错误类型有随机 1 位翻转、随机多位翻转等。

### 2.1 物理层故障注入原理

传统的物理层注入方法有管脚信号修改、电源干扰、粒子轰击等，这些方法不仅难以实现，而且不易控制，费用昂贵，还会对硬件造成损害。安腾架构处理器的固件中集成了丰富的调试接口，可以通过固化的程序触发硬件错误，能够触发错误的部件有寄存器、TLB、Cache 等。

作为存储器中使用频率最高的部件，Cache 的可靠性对系统有很大影响。因此，本文选择 Cache 作为物理层的注入目标。虽然无法对内存注入物理故障，但从软件的角度来说，Cache 错误和内存错误对系统的影响是等效的，因此 Cache 故障可以很好地评测物理层故障对系统的影响。

### 2.2 软件层故障注入原理

当硬件层容错机制没有检测到内存故障，或者对检测到的故障无法处理时，内存故障就会传播到软件层，影响系统内核和应用进程的正常运行。

已有的 Ferrari、DOCTOR 等故障注入工具大都在进程的虚拟地址空间上注入故障。由于虚存和物理地址间需要转换，因此往虚存空间注入的故障很难被触发，注入位置也不够精确，更是无法对内核代码段等区域进行故障注入，具有很大的局限性。本文提出一种新的思路，利用 Linux 提供的

**基金项目：**国家“863”计划基金资助重点项目(2008AA01A204, 2009AA01A404)；国家自然科学青年基金资助项目(61003047)；国家科技部国际科技合作计划基金资助项目(2010DFA14400)

**作者简介：**王 波(1987－)，男，硕士研究生，主研方向：容错计算；左德承，教授；钱 军，博士研究生；张 展，讲师

**收稿日期：**2011-08-09　**E-mail：**wbhitren@gmail.com

/dev/ mem 虚拟设备直接向物理内存地址进行注入故障，能够更精确地控制故障注入的位置。注入的位置有内核代码段、进程代码段和进程数据段等。

## 3 HMFI 注入器的设计与实现

HMFI 故障注入器采用 C/S 的设计，客户端与待测系统通过高速网络通信，客户端提供 GUI 界面给实验操作员，同时监控目标系统状态，并分析处理实验数据。系统各模块的工作流程如图 2 所示。

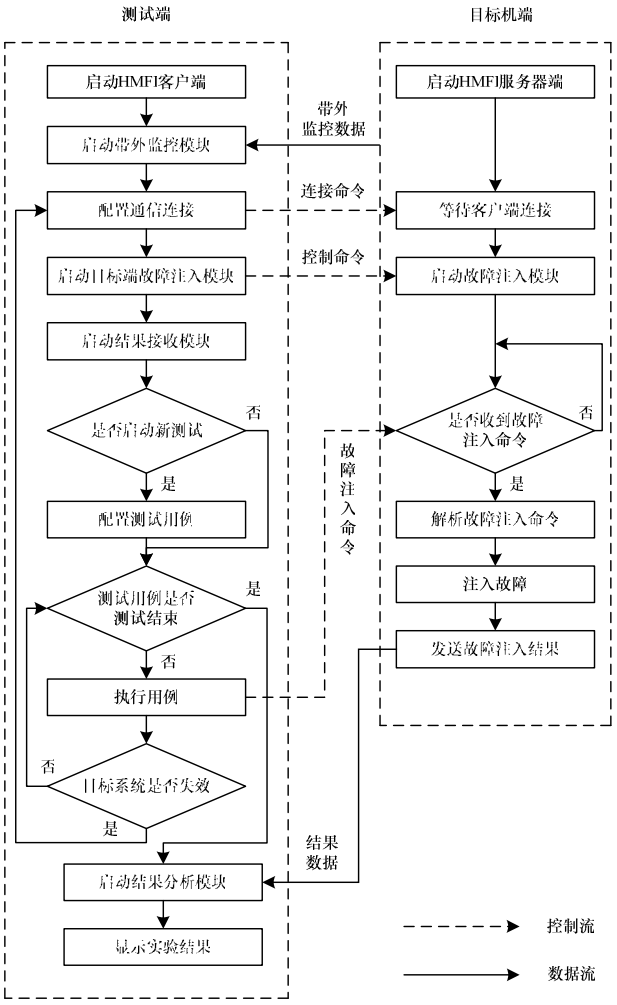


**图 2 系统工作流程**

为实现自动化测试，HMFI 通过带外监控的方式对目标系统状态进行监控，当目标系统的操作系统失效时，可以继续进行监控，保证测试不因系统失效而中断，相比于以往的故障注入工具有很大提高[2]。带外监控采用 IPMI[3]实现，系统监控模块通过局域网与目标系统的 BMC(Baseboard Management Controller)模块进行通信，获取目标系统的基本状态数据，如图 3 所示。

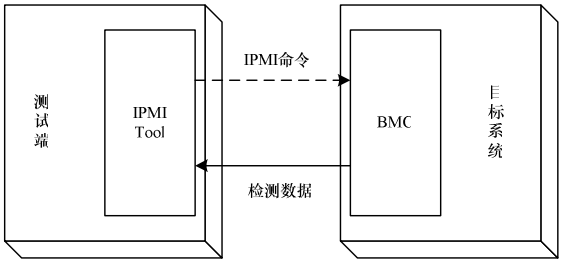


**图 3 带外监控的自动化测试**

### 3.1 物理层故障注入工具

安腾处理器的 PAL 固件中集成了丰富的故障注入例程，用以处理器的内部测试。通过构建 PAL 例程的调用环境，可以实现向 Cache 注入物理故障。注入故障后，利用安腾架构的机器错误检查机制(MCA)，可以实现对故障的跟踪和结果回收。

如图 4 所示，该工具使用/sys 虚拟文件系统作为用户态程序与内核模块交互的接口，用于传递故障注入参数和流程控制。/sys 文件系统方便地在每个处理器核下创建调用接口，比 proc 文件系统更适合于 Cache 的故障注入参数传递。故障注入内核模块通过调用 PAL_MC_ERROR_INJECT 函数实现对固件中故障注入例程的调用。故障被注入后，Linux IA64 操作系统提供内核日志和 SAL 日志(/var/log/salinfo 目录下)用于记录故障信息。目标系统的 BMC 模块将部件级的故障信息记录在 FPL 日志和 SEL 日志中，可以通过远程 IPMI 工具获取得到。

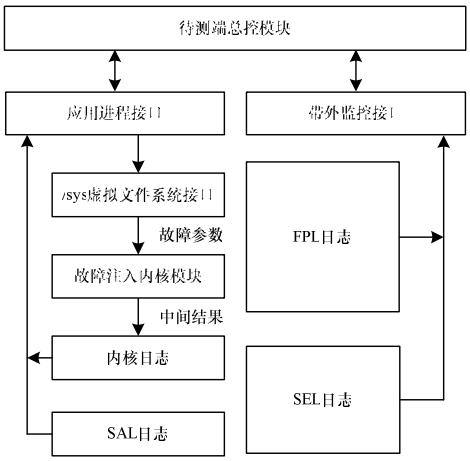


**图 4 基于 PAL 固件的内存故障注入工具设计**

### 3.2 物理地址内存故障注入工具

Linux 操作系统提供的/dev/mem 虚拟设备是物理内存的全镜像，可以用来访问物理内存，利用 mmap 可以将/dev/mem 中一段物理内存地址映射到故障注入程序的线性地址空间，然后通过线性地址的读写实现对物理内存的读写，以模拟物理内存发生故障。物理地址故障注入工具的总体设计如图 5 所示。

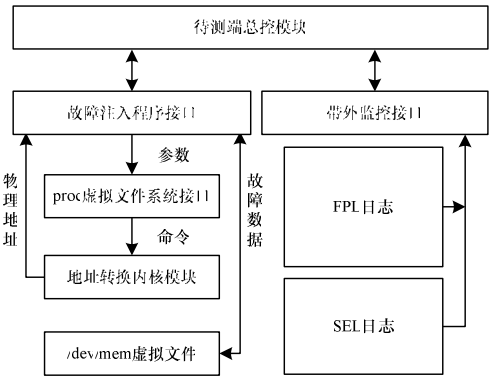


**图 5 物理地址内存故障注入工具设计**

要进行物理地址内存故障注入，首先需要通过物理地址转换模块，将内核代码段线性地址和进程线性地址转换为物理地址。内核代码段线性地址属于一一映射区，可以使用__pa()转换得到物理地址，对于进程线性地址，则需要查询页表获取物理地址[4]。

故障注入后，可以通过内核日志和带外监控日志等跟踪分析故障。对于向进程注入的内存故障，还可以通过 Linux 的 waitpid 系统调用[5]来跟踪进程的运行状态，分析进程退出或异常终止的原因。

## 4 故障注入实验

利用 HMFI 故障注入器，以 HP Superdome 服务器作为待测目标系统，验证故障注入器的有效性，对目标系统的容错能力进行测试分析。Superdome 服务器配置如表 1 所示。

**表 1 待测系统配置**

| 类别 | 属性 | 备注 |
|---|---|---|
| 节点 | 4 个 | 每个节点配有 8 个 2 GB 的内存条 |
| 处理器 | 安腾 II | 共 8 个处理器 |
| 1 级 Cache | 32 KB | 16 KB 指令 Cache，16 KB 数据 Cache |
| 2 级 Cache | 1 280 KB | 1 MB 指令 Cache，256 KB 数据 Cache |
| 3 级 Cache | 9 216 KB | 处理器间共享 |
| 内存 | 64 GB | 每个节点 16 GB |

### 4.1 物理内存故障实验

利用物理层故障注入工具，向处理器分别注入进程中 1 位随机翻转、内核中 1 位随机翻转、进程中多位随机翻转和内核中多位随机翻转故障，统计结果如表 2 所示。

**表 2 物理内存故障注入情况**

| 故障 | 注入结果 |
|---|---|
| Cache 随机 1 位翻转 | 无异常，日志记录故障被纠正 |
| Cache 随机多位翻转 | 故障被检测到，操作系统试图修复<br>故障失败，系统重启 |
| 物理内存 | 64 GB |

下面以针对 17 号 CPU 的随机多位翻转 Cache 故障注入为例，分析故障日志。由于相关日志内容复杂，为便于说明，只从日志中抽取关键语句列出。

(1)BMC 日志

#1 Machine Check initiated

#2 MC timestamp

#3 BR_TO_OS_MCA

#4 OS_MCA_UNCORRECTED_MC

#5 OS MCA did not correct the Machine Check

(2)内核日志

#1 Entered OS MCA handler

#2 MCA: kernel context not recovered

#3 MCA/INIT might be dodgy or fail

#4 Restart

(3)固件日志

#1 data cache level 2 error

#2 PAL recovery status: error was isolated and contained, continuable if sw can recover

带外监控日志记录表明，硬件检测到硬件故障，开始进行错误检查(Machine Check)，并记录检测到故障的时间；该故障被上传给操作系统(BR_TO_OS_MCA)，操作系统的错误修复程序没能纠正该故障(OS_MCA_UNCORRECTED_MC)。

内核日志显示了当故障上报到操作系统时，错误修复程序开始运行(Entered OS MCA handler)，但是无法恢复该故障，开始重启。固件层日志表明，PAL 层固件定位到了故障发生在 17 号 CPU 的 2 级数据 Cache，但无法纠正，只能等软件层处理后再继续，而软件层的 MCA handler 程序选择了重启。该故障最终导致了系统失效。

### 4.2 内核代码段内存故障实验

随机选择内核代码段中的一个字，向该字注入随机 1 位翻转故障，实验结果如图 6 所示。可以看出，当内存故障传播到内核代码段时，其结果往往是致命的，会导致内核崩溃，系统失效。

| 实验次数 | 内核崩溃，重启 | Swapper 进程崩溃，尝试修复失败，重启 |
|---|---|---|
| 10 | 8 | 2 |

**图 6 内核代码段内存故障实验结果**

### 4.3 进程内存故障实验

运行 unixBench 作为负载，分别向负载进程的代码段和数据段注入故障，错误类型为随机 1 位翻转，各进行 100 次故障注入。发生在进程代码段的内存故障有 68%的概率引起访存异常，致使负载进程收到 SIGSEGV 信号中断；有 1%的情况下会触发机器错误检查处理程序，但故障修复失败，导致系统失效重启；另有 1%的概率直接导致内核崩溃，系统失效；只有 30%的情况下负载运行正常，如图 7 所示。发生在进程数据段的内存故障有高达 19%的概率触发极其错误检查处理程序，同样无法修复故障，导致系统失效重启；其他情况下，负载运行正常，如图 8 所示。从结果可以看出，传播到进程中的内存故障有 10%的概率导致系统失效，只有 56%的情况不受影响。

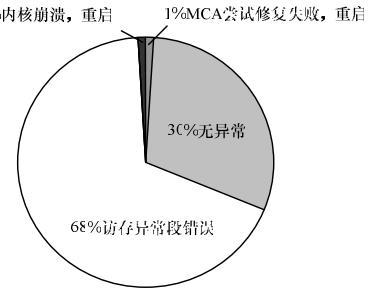


**图 7 进程代码段 1 位翻转故障注入结果**

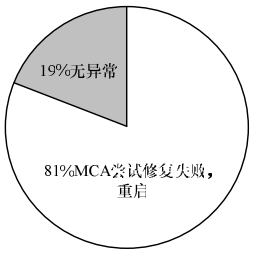


**图 8 用户进程数据段 1 位翻转故障注入**

## 5 结束语

本文提出一种多层次内存故障注入的方法，设计并实现 HMFI 内存故障注入器。为验证故障注入方法的正确性，本文对 Superdome 服务器进行多层次的故障注入实验，通过对内核日志、BMC 日志和固件层日志等的分析，系统地测试目标系统对传播到各个层次的内存故障的容错能力。本文只讨论内存故障，下一步将针对 CPU 故障、I/O 故障等进行容错评测的研究，并对共享内存多处理器架构计算机的可靠性进行研究。

## 参考文献

[1] 孙峻朝, 李运策, 杨孝宗. 故障注入研究的一种理论框架[J]. 小型微型计算机系统, 1999, 20(11): 816-819.

[2] 叶俊民, 熊华根. 运行时软件故障注入器的设计与实现[J]. 计算机工程, 2008, 34(24): 4-6.

(下转第 75 页)