

系统可用性基准测试技术综述

翟季冬

dijd03@mails.tsinghua.edu.cn

清华大学计算机科学与技术系高性能所

摘要：可用性和可维护性是现代高性能计算机系统设计的目标。基准测试程序在引导计算机科学系统研究和发展的过程中起到了历史性的关键作用，但是传统的测试程序忽略了可用性（availability）、可维护性（maintainability）和可改进性（evolutionary growth）等领域，最近这些测试领域在高端系统设计中已经变得越来越重要。本文总结了关于系统可用性和可靠性方面测试的最新工作，并着重分析了 Berkeley 大学针对 ISTORE 项目设计的可用性基准测试程序的设计原理。

关键词：可用性基准测试程序，故障插入

A Summary for System Availability Benchmark Technology

Zhai Jidong dijd03@mails.tsinghua.edu.cn

Department of Computer Science & Technology

Tsinghua University 100084, China

Abstract: Availability and maintainability have becoming more and more important in high performance computing system design. Benchmarks have historically played a key role in guiding the progress of computer science systems research and development, but have traditionally neglected the areas of availability, maintainability, and evolutionary growth, these areas that have recently become critically important in high-end system design. In this paper, the latest technologies about availability benchmark are summarized, and the availability benchmark designed for the system of ISTORE in Berkeley University of California is analyzed in detail.

Key words: Availability benchmark, Fault injection

1 介绍：

现在对高性能计算机系统的需求越来越大，系统的硬件成本开始逐渐降低，但是系统的维护管理费用却大幅度的增加。对于一个希望提供 24*365 小时不间断服务的大型商务系统，其管理成本远远大于它的硬件成本。对许多使用高性能计算机系统的部门，如大型电子商务网站，他们对系统的可用性和可靠性的需求特别高。在 2000 年的一个统计中，像 EBay 这样的电子商务网站

down 机一小时会造成 22.5 万\$的损失。单单是从经济学角度出发，系统的可用性和可管理性就已经变得非常重要了。

为了对系统的可用性和可管理性进行比较和测试，我们需要一套完整的基准测试程序对系统进行科学评估。在过去，系统测试主要集中在系统的性能测试，如 SPEC CPU2000, SPEC HPC2002, TPC, Linpack 等等。它们可以回答系统在正常工作情况下哪个系统速度更快，但是却无法回答哪个系

统具有更好的可靠性，哪个系统全年 down 机次数最少。计算机系统的可用性与系统的性能是同等重要的，应该也是衡量高性能计算系统性能的重要指标。但是目前国际上对高性能计算系统可用性和可管理性的测试技术与开发还处于刚刚起步阶段，尚无成熟的测试方案和基准测试程序。John Hennessy 在一次报告说：我们不应该在一味的过分强调系统的性能，而诸如可用性、可管理性、可扩展性等系统的品质也应引起我们足够的重视。

现在 Berkeley 大学针对他们研制 ISTORE 系统开发了一套用于测试可用性的基准测试程序。SUN 微系统有限公司也开发了一套用于测试系统恢复特性的测试程序。本文重点分析 Berkeley 大学设计的测试程序的原理和方法，并总结了一些通用的测试方法。

2 可用性基准测试程序

2.1 可用性定义

在我们研究可用性之前，首先需要对系统的可用性下个完整的定义。在过去，系统的可用性一般用来指在一段时间内系统处于 UP 状态的时间百分比。在这个定义中，系统在 UP 和 DOWN 两种状态中必居其一，UP 状态指系统的完全正常状态，DOWN 状态指系统的故障状态。所以经常有系统被描述为 5 个 9 的可用性（99.999%）目标，就是按照这种定义来的。

由于系统在 UP 和 DOWN 状态之间可以有很多降级状态（Degraded States），当系统处于这些降级状态时，尽管系统的性能会受到影响，但是系统仍然是可以使用的，所以需要重新定义系统的可用性。因此一个好的可用性基准测试程序必须可以捕捉到系统这些降级状态，而不是仅仅可以测量系统是否处于正常状态，同时也应该记录系统处于降级状态的效率和提供服务的质量。

其次，系统的可用性不应该仅仅用一个时间值或某段时间内的平均时间值来衡量。比如从用户的角度看，有一个系统在每分钟内有两秒钟不可用，另一个系统在一个月有一整天不可用。虽然两个系统平均可用时间是相同的，但从用户使用的角度来看系统的可用性却是不同的，可用性测试程序应该能捕捉到这两种系统的差异。因此，系统的可用性必须用某段时间内系统服务质量的某种对应关系来衡量。

在综合考虑这两种需求下，Brown 提出一种测试可用性方法：测量系统随着时间服务质量的变化情况，服务质量的测量具体依赖于所研究的系统类型，一般测量系统的性能和故障承受程度。比如对于一个 web 服务器，就是每秒中响应用户的请求数，以及它的存储子系统可以容纳的故障数，其它可能的测量包括：

完整性：对于一个类似于搜索引擎的系统，当系统处于降级状态，可以测量它能返回多少可用的数据库结果。

容量 :当一个系统处于降级状态的时候它可能允许连接的用户数和可接受的任务数都变少了。

2.2 测试方法

Brown 提出的这种测试计算机系统可用性方法的**核心思想**是 :使用了一种故障插入 (Fault Injection) 技术 , 首先充分利用已有的标准测试程序 (他们使用的是 SPECWeb99) 给系统产生一定的负载 , 然后通过人为的方式在系统中插入故障使系统的可用性受到影响 , 同时记录系统在处于各种故障的情况下所能提供的服务质量的情况。

为了建造一个可用性测试程序环境 , 需要有一种方式产生实际的工作负载来度量服务质量。在该系统中充分使用了现有的测试程序来为给定的系统产生一个有代表性的工作负载。该测试环境还能产生模拟的故障和维护事件 , 比如模拟磁盘阵列失败 , 存储器失效 , 进程杀死 , 电源断电等等都应该可以模拟。

作者在给系统插入故障的时候 , 把故障负载 (fault workload) 分为两种 : 单故障负载 (Single-fault workloads) : 仅仅包括一个故障 , 一旦系统进入稳定状态 , 单点故障就可以注入系统 , 系统的行为也就是服务质量情况被记录。多点故障负载 (Multi-fault workloads) : 多点故障由一系列故障和维护事件组成 , 用来模拟现实世界中的故障事件 , 可以对组合故障进行较好的模拟。

测试的过程主要分两步 : 第一、标准的性能测试程序在一个没有故障插入的环境下执行 , 同时一些必要的性能指标被记录 , 这一步建立了一个对于没有故障的系统的基本测量。第二 , 负载运行的同时插入故障负载 , 同时记录各项性能指标 , 这一步是整个可用性测试的关键部分。因为它可以反映一个系统在有故障干扰的情况下服务质量的变化情况 , 也就是我们所要测的系统的可用性度量。

2.3 测试过程

Brown 选择软件 RAID 系统作为测试系统可用性的对象 , 选择软件 RAID 的主要原因是 : 软件 RAID 的实现可以使用各种操作系统版本 , 包括商业的和免费的 , 便于在各个系统之间进行比较系统的可用性。更重要的是 RAID 很好的定义了可用性的目标 , 是一个理想测试系统可用性的应用程序。同时软件 RAID 的系统是许多网络服务应用的底层 , 所以软件 RAID 可以很好的代表应用。

RAID-5 是有四块磁盘组成的冗余备份存储系统 , 最多可以允许一个磁盘出错 , 如果有一块新盘代替出错磁盘 , 系统可以根据其它三块磁盘的数据重新构建 RAID 系统。作者使用三块实际的磁盘和用一个 PC 机安装相应的软件模拟物理磁盘来构建一个 RAID-5 系统。测试的时候通过软件的方法控制那块模拟的磁盘出现故障 , 实现插入单点故障和多点故障。作者应用该方法测量了

一个分别由 Linux、Window 2000、Solaris 服务器组成的软件 RAID 系统，这种方法不仅可以量化各种故障情况对系统可用性的影响，同时也可以发现瞬时误差和恢复策略的情况。

实验结果：在研究软件 RAID 系统的可用性后，作者发现使用不同的操作系统实现的 RAID 会有很大的区别。Linux 系统和 Solaris 系统发现 RAID 磁盘出现故障的时候，如果有空闲的磁盘可供使用，系统可以自动重新构建 RAID，而 Windows 系统需要人工干预来重新构建一个 RAID 系统。Linux 系统会对系统的瞬时故障反映很灵敏，它会处理所有的瞬时故障，即使该故障不会引起系统的性能下降。Linux 在从系统的故障状态恢复到正常状态的过程，对整个系统的性能影响比较小，可以同时对外提供服务，但是需要的时间比较长。Solaris 系统在从故障恢复的时间比较短，但是对系统的性能影响比较大。Windows 系统处于上述两种系统之间，在出现磁盘故障后需要人工干预才可以恢复。图 1 就是 Linux 和 Solaris 系统分别从故障恢复的过程，x 轴表示时间，y 轴表示系统的每秒相应数，可以理解为系统的性能或服务质量。上图是 Linux 系统恢复的过程，可以看出系统恢复的时间比较长，系统影响比较小。下图是 Solaris 系统恢复过程，可以看出系统恢复的时间比较短，系统性能影响比较大。

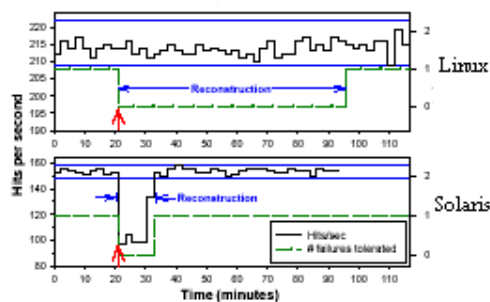


图 1 系统恢复可用性图表

3 可恢复性基准测试程序

对可恢复性基准测试的研究主要是 Sun Microsystems, Inc. 在他们的文章里面描述了一个用来测量系统从故障恢复能力的基准测试程序。主要是为了区分不同系统在处于故障的情况下，哪个系统可以更快的从故障恢复过来，用来衡量系统的可恢复性。

他们把系统的可用性定义为三个因素：速率（rate）、健壮性（robustness）、可恢复性（recovery）。Rate 是指系统在一个给定的时间里面故障和维护事件出现的次数。Robustness 是指系统所能检测和处理故障的能力，以及系统在面对各种故障情况下仍然具有的工作能力。Recovery 是指系统从故障状态恢复到正常状态所需要的速度。

以前关于系统可用性的测试一般都是集中在系统的健壮性上，其实本文上面谈到的 Berkeley 大学的可用性测试程序就是集中在健壮性的研究。而 Sun 的研究人员开发的系统 SRB (System Recovery Benchmark) 主要是注重在系统的恢复特性上。该系统主要是用来比较不同系统在处于相同的故障下，哪个系统可以更快恢复过来，它主要集

中在系统可用性的第三方面 恢复性。在该测试中也使用了故障插入技术,他们把负载进行扩展,分为故障负载(fault load)和工作负载模型(workload model)。

4 结论

本文主要是对系统可用性和可靠性的测试工作进行了简单的总结,由于这方面的研究处于刚刚起步阶段,研究成果不是很多。本文主要围绕 Berkeley 的可用性基准测试程序为中心,重点分析了可用性测试领域普遍使用的故障插入技术,并总结了相关的一些测试方法,希望本文能对下一步工作做一点铺垫。

5 参考文献

- 【1】A. Brown and D. A. Patterson, Towards Availability Benchmarks: A Case Study of Software RAID Systems”, USENIX 2000.
- 【2】I. Pramanick, J. Mauro, J. Zhu, “A System Recovery Benchmark for Clusters” Proceedings of the IEEE International conference on Cluster Computing. 2003
- 【3】J. Mauro, J. Zhu, I. Pramanick, “The System Recovery Benchmark” Proceedings of the 10th IEEE Pacific Rim International Symposium on Dependable Computing. 2004.
- 【4】D. P. Sieworek et al, “Development of a Benchmark to Measure System Robustness”, Proceedings of the 1993 International Symposium on Fault Tolerant Computing. June 1993.
- 【5】T. Tsai et al, “An Approach Towards Benchmarking of Fault Tolerant Commercial Systems”, Proceedings of the 1996 Symposium on Fault Tolerant Computing, June 1996.