HP Performance Engineering 最佳实践系列

适用于性能工程师和经理

性能监控最佳实践

文档发布日期：2014 年 3 月

法律声明

担保

HP 产品和服务的唯一担保已在此类产品和服务随附的明示担保声明中提出。此处的任何内容均不构成额外担保。HP 不会为此处出现的技术或编辑错误或遗漏承担任何责任。

此处所含信息如有更改，恕不另行通知。

受限权利声明

机密计算机软件。必须拥有 HP 授予的有效许可证，方可拥有、使用或复制本软件。按照 FAR 12.211 和 12.212，并根据供应商的标准商业许可的规定，商业计算机软件、计算机软件文档与商品技术数据授权给美国政府使用。

版权声明

商标声明

Adobe® 是 Adobe Systems Incorporated 的商标。

Java 是 Oracle 和 / 或其子公司的注册商标。

Microsoft® 和 Windows® 是 Microsoft Corporation 的美国注册商标。

文档更新

此文档的标题页包含以下标识信息：

- 软件版本号，用于指示软件版本。
- 文档发布日期，该日期将在每次更新文档时更改。
- 软件发布日期，表示此版本软件的发布日期。

要检查是否有最新更新或验证所使用的文档是否为最新版本，请访问：

http://h20230.www2.hp.com/selfsolve/manuals

此站点需要注册 HP Passport 才能登录。要注册 HP Passport ID，请访问：

http://h20229.www2.hp.com/passport-registration.html

或单击 “HP Passport” 登录页面上的 **New users - please register** 链接。

此外，如果订阅了相应的产品支持服务，则还会收到更新的版本或新版本。有关详细信息，请与您的 HP 销售代表联系。

支持

请访问 HP 软件支持网站：

http://www.hp.com/go/hpsoftwaresupport

此网站提供了联系信息，以及有关 HP 软件提供的产品、服务和支持的详细信息。

HP 软件联机支持提供客户自助解决功能。通过该联机支持，可快速高效地访问用于管理业务的各种交互式技术支持工具。作为尊贵的支持客户，您可以通过该支持网站获得下列支持：

- 搜索感兴趣的知识文档
- 提交并跟踪支持案例和改进请求
- 下载软件修补程序
- 管理支持合同
- 查找 HP 支持联系人
- 查看有关可用服务的信息
- 参与其他软件客户的讨论
- 研究和注册软件培训

大多数提供支持的区域都要求您注册为 HP Passport 用户再登录，很多区域还要求用户提供支持合同。 要注册 HP Passport ID，请访问：

http://h20229.www2.hp.com/passport-registration.html

要查找有关访问级别的详细信息，请访问：

http://h20230.www2.hp.com/new_access_levels.jsp

目录

目录

欢迎使用本指南

欢迎使用*性能监控最佳实践*。

本指南介绍在各种环境中最佳实现性能测试监控有关的概念、准则和实践示例。

此章节包括：

关于 HP 性能监控

HP 是自动化性能测试的市场领导者。自动化性能测试是利用产品、人员和过程来降低应用程序、升级或修补程序部署风险的一种规程。其核心功能是将生产工作负载应用于预部署系统，同时度量系统性能和最终用户体验。如欲构建结构合理的性能测试，请回答如下问题：

- 应用程序能够为预期的用户作出足够快速的响应吗？
- 应用程序是否能够处理预期的 （以及超预期的）用户性能测试？
- 应用程序可以处理业务所需的事务数吗？
- 应用程序在预期和非预期的用户性能测试下是否稳定？
- 确定上线当天用户会有良好的体验吗？

回答这些问题后，自动性能测试即可量化业务方面的变更带来的影响。这反过来又说明了部署风险。有效的自动化性能测试过程帮助您进行更成熟的发布决策，防止出现系统停机和可用性问题。

HP 在自动化性能测试方面提供两种产品 —— HP LoadRunner 和 HP Performance Center。每种产品都侧重于不同的市场，但两种产品都构建在已经证实并共享的基础之上 （包括：支持的协议和监控器等）。

HP LoadRunner 支持在受控和峰值性能测试条件下测试系统。要生成性能测试，LoadRunner 运行分布在网络上的数千虚拟用户 (Vuser)。 Vuser 可以在基于 UNIX 和基于 Windows 的平台上运行。这些 Vuser 通过使用最低限度的硬件资源，提供一致、可重复和可度量的性能测试以测试接受测试的应用程序 (AUT)，就象真实用户所做的那样。

HP Performance Center 是 Application Lifecycle Management (ALM) 套件的一部分，是安装在您组织自身的基础结构上的企业全局性能测试工具。

- Performance Center 支持跨地域管理多个并发性能测试项目，无需在两地间穿行。
- Performance Center 管理所有内部性能测试需要。
- 使用 Performance Center，可以从可通过 Web 访问的中心位置管理大型性能测试项目的所有方面，包括资源分配和计划。
- Performance Center 可帮助简化测试过程，减少资源成本，提高运行效率。
- Performance Center 帮助查明性能瓶颈。
- Performance Center 允许您确定测试中的应用程序可以扩展到的最大用户数。（此数字是应用程序的性能开始降级的**临界点**。）这些信息为您提供线索，以找到增加应用程序性能测试容量的应对措施。

为了满足性能监控团队的需要，并减少配置和部署相关监控器的所需时间，我们在本指南中包含了性能监控准则，并且还提供一组由默认度量、默认阈值（如适用）和主动测试（如适用）构成的预置监控器。所有这些都是以各种来源（包括 HP 操作系统管理员、HP 专业服务组织、技术文档和业界专家著作）的最佳实践数据和专业知识研究而得。使用这些准则监控系统性能将有助于识别性能瓶颈，而性能瓶颈将成为系统问题的根源。

本指南的目的是提供简单易用的全面性能监控准则，而使 Performance Center 用户或 IT 组织无需成为应用程序方面的专家。

本指南的组织方式

*HP 性能监控最佳实践*包含以下部分：

第 I 部分　简介

介绍性能监控和解决方案。

第 II 部分　操作系统

提供监控 Windows 和 UNIX 操作系统的最佳实践。

第 III 部分　运行时平台

提供监控 Java 和 .NET 运行时平台的最佳实践。

第 IV 部分　Web 服务器监控

提供监控 Apache 和 IIS Web 服务器的最佳实践。

第 V 部分　应用程序服务器监控

提供监控 WebLogic 和 WebSphere 应用程序服务器的最佳实践。

第 VI 部分　数据库资源监控

提供监控 Oracle 和 MSSQL Server 数据库资源的最佳实践。

第 VII 部分 虚拟化技术

提供监控 Microsoft Hyper-V 和 VMWare 管理程序平台的最佳实践。

本指南面向的读者

本指南面向以下对象：

- 性能工程师
- Performance CoE 经理
- QA 经理
- QA 工程师

其他联机资源

疑难解答和知识库可访问 HP 软件支持网站上的疑难解答页，并在页面上搜索自解决知识库。选择**帮助 > 疑难解答和知识库**。此网站的 URL 是 http://h20230.www2.hp.com/troubleshooting.jsp。

HP 软件支持可访问 HP 软件支持网站。通过此站点，可浏览自解决知识库。您还可以在用户讨论论坛发帖和搜索信息、提交支持请求、下载修补程序和更新的文档等。选择**帮助 > HP 软件支持**。此网站的 URL 是 www.hp.com/go/hpsoftwaresupport。

大多数提供支持的区域都要求您注册为 HP Passport 用户再登录，很多区域还要求用户提供支持合同。

要查找有关访问级别的详细信息，请访问：

http://h20230.www2.hp.com/new_access_levels.jsp

要注册 HP Passport 用户 ID，请访问：

http://h20229.www2.hp.com/passport-registration.html

HP 软件网站可访问 HP 软件网站。此站点提供了有关 HP 软件产品的最新信息。这里包含新的软件发布、研讨会及展览会、客户支持和其他更多信息。选择**帮助 > HP 软件网站**。此网站的 URL 是 www.hp.com/go/software。

文档更新

HP 软件不断地使用新信息更新其产品文档。

要检查是否有最新更新或验证所使用的文档是否为最新版本，请访问 HP Software Product Manuals （HP 软件产品手册）网站 (http://h20230.www2.hp.com/selfsolve/manuals)。

第 I 部分

简介

1

介绍性能监控

性能监控是更大范围的性能测试规范的一部分，涉及测试中的应用程序的性能度量。

此外，性能监控还可用于验证系统的其他质量属性，如可扩展性、可靠性和资源使用性能。

此章节包括：

性能监控概述

性能监控确保您能得到有关性能测试中的应用程序运行状况的最新信息。通过分析多个性能测试得到的系统性能数据，就可以定义基线，即表示典型运行条件下可接受性能的一系列度量。该基线提供了参考点，当发生问题时就更容易识别。

此外，排除系统故障时，性能数据能提供有关问题发生时系统资源的行为的信息，这有助于查明原因。

最后，监控应用程序性能可为您提供数据，以预测未来增长、规划系统配置中的更改可能如何影响未来的操作。

性能监控通过收集反映不同工作负载条件（性能测试、负荷或单用户操作）下应用程序行为的度量，帮助您识别瓶颈，验证应用程序是否符合其性能目标。这些度量随即应与性能目标中定义的那些相关联。此类度量的示例有：响应时间、吞吐量和资源利用率（即 CPU、内存、磁盘 I/O、网络带宽）。如果没有很好地了解这些度量，分析性能结果时就很难得出结论并查明瓶颈。强烈建议您学习专业知识以便能执行正确的分析。

为达到最佳性能而配置和调整应用程序，是应用程序开发者和 IT 公司中持续关注的问题。指出**为什么**特定应用程序运行缓慢的能力是一种很令人向往的技能，它是科学与艺术的结合。无论您的技能或技艺水平如何，收集性能数据都是诊断和解决各种问题必需的第一步。

性能术语

性能测试的数量方面是在监控阶段收集的。请仔细看看性能监控中使用的主要术语。

系统行为的两个最重要度量是**带宽**和**吞吐量**。带宽是数量度量，是工作*可*完成的比率，而吞吐量则度量完成工作请求的*实际*比率。

吞吐量可随应用于测试中的系统的用户数而不同。它通常以每秒请求数度量。在某些系统中，有很多并发用户时吞吐量可能下降，而在其他系统中，它在压力下保持为稳定，但通常由于排队开始出现严重延迟。计算机系统的各种资源的繁忙程度称为**利用率**。

执行特定任务所花时间的关键度量是**队列时间**、**服务时间**和**响应时间**。

服务时间和队列时间

服务时间度量处理特定的客户工作请求需要多久。

工作请求到达时资源繁忙，不能立即处理时，请求将入队。请求被处理之前在队列中等待时，受到**队列时间**延迟的影响。

响应时间

响应时间是最重要的度量，并将在整本书中经常用到，指的是服务时间和排队时间的总和。它可以在服务器或客户端如下划分成响应时间：

- **在服务器上度量的延迟。**这是服务器执行完请求所需的时间。不考虑客户端到服务器的延迟，后者包括请求和响应穿越网络需要的额外时间。
- **在客户端上度量的延迟。**客户端上测得的延迟包括请求队列、服务器执行请求所花的时间和网络延迟。深入了解应用程序的使用是必需的，这样才能形成活动与其在用户中的热门程度的正确混合。

工作负载配置文件、容量和可扩展性

另一个影响性能监控结果的重要术语是**工作负载配置文件**，是给定的测试中应用程序内执行各种操作的用户的混合。

容量描述利用率达到最大时，每个资源可以处理多少工作，而**可扩展性**通常定义为计算机或系统的吞吐量，是请求服务的用户总数的函数。

影响性能的因素

人们已经知晓多年，尽管软件开发总是向着持续改进努力，但它将永远不会是 100% 完美的。反过来，应用程序的性能好坏也只能相对其性能目标而言。

性能问题影响所有类型的系统，不管它们是客户端 / 服务器还是 Web 应用程序系统。在着手处理任务之前，必须了解影响系统性能的因素。

一般而言，影响性能的因素可分为两大类：面向**项目管理**和**技术**的。

影响性能的项目管理因素

在现代软件开发生命周期 (SDLC) 中，主要的阶段受到时间限制影响，这是为了应对日益激烈的竞争。这导致出现项目管理问题：

- 开发中写代码的时间更短了，可能导致由于缺少对性能的关注而降低产品质量。
- 由于快速开发途径导致的丢失信息的可能性会使性能目标无法达到。
- 产品部署之后可能会发现不一致的内部设计，例如，屏幕导航的对象和序列太多太乱。
- 违反编程标准的概率增加，导致代码未优化，占用过多资源。
- 由于设计是特定于项目的，导致不可能在将来的项目中重用模块。
- 模块无法针对可扩展性进行设计。
- 系统可能由于用户性能测试中规模突然增加而崩溃。

影响性能的技术因素

项目管理的相关问题对输出有重大影响，而技术问题可能严重影响应用程序的总体性能。问题可能源自技术平台的选择，后者可能为特定目的而设计，在不同条件下表现不佳。

但是通常技术问题的出现是由于开发者疏忽了对性能的考虑。很多开发者的通用做法是在开发阶段不优化代码。这样的代码可能会不必要地使用稀缺的系统资源（如**内存**和**处理器**）。这样的编程习惯可能导致严重的性能瓶颈，诸如：

- 内存泄漏
- 数组绑定错误
- 缓冲区效率低
- 处理循环太多
- HTTP 事务数很大
- 内存和磁盘之间文件传输太多
- 低效率的会话状态管理
- 由于最大并发用户数导致线程冲突
- 对于峰值性能测试，体系结构规模不足
- 低效率的 SQL 语句
- 数据库表上缺少正确的索引
- 服务器配置不当

一旦封装了代码以便进行部署，这些问题就很难跟踪了，需要特殊的工具和方法。

影响性能的另一组技术因素是**安全**。应用程序的性能和它的安全性经常有冲突，因为增加安全层 （SSL、私钥 / 公钥等等）会使计算量非常大。

还必须考虑**网络**相关问题，特别是与 Web 应用程序相关的问题。它们可能来自各种来源，例如：

- 较早或未优化的网络基础结构
- 缓慢的网站连接会增加网络流量，因而使响应时间变长
- 服务器上影响性能的不平衡性能测试

性能目标

要成功监控性能测试中的系统，监控性能的途径和监控自身都必须和性能项目的环境相关。因此作为性能测试生命周期 (PTLC) 的一部分，监控的第一步应是定义**性能目标**。这些目标是指通过性能测试过程收集，且预期在决定或改进产品质量时有参考价值的数据。但是，这些目标不一定是量化的，也不一定直接和其他已表述的性能条件相关。

这些目标通常包括以下特征中的全部或部分：

- **契约。**通常在业务客户和测试实体之间正式定义性能目标为：
  - **必备。**由于法律责任、服务级别协议 (SLA) 或固定的业务需要，必须达成的条件。
  - **可协商。**对产品发布而言最好能达成的条件，但某些情况下也可以修改。这些通常 （但不一定）是关注最终用户的。

- **精度。**描写性能目标的数量方面的措辞有：
  - **准确。**应完全达到目标中所写的条件，例如 “50% CPU 利用率”。
  - **接近。**条件落在某个范围内或仅有一个限制，例如，“每个进程使用内存不超过 50MB”、“事务 X 的至少 90% 的响应时间不应长于 3 秒。”
- **边界**。性能目标频繁定义有关测试中的应用程序的特定值：
  - **目标。**这是一组特定条件下的资源的理想值，通常在响应时间、吞吐量和资源利用率水平方面指定。
  - **阈值。**这表示资源的最大可接受值，通常在响应时间、吞吐量（每秒事务数）和资源利用率水平方面指定。

性能目标及其服务属性派生自业务要求。通过测量捕获的被监控度量，显示朝向或离开性能目标的进展情况。

性能监控准则

准备性能监控时，要记住以下的简单常规准则：

- 从标准采样间隔开始。如果问题较具体，或能查明怀疑的瓶颈所在，则缩短时间段。
- 根据采样间隔决定整个监控会话的长度。只应对较短的运行执行间隔频繁的采样。
- 尝试在监控的对象数和采样频率之间取得平衡，使收集的数据保持在可管理的限度内。

- 仅选取与测试中应用程序性质相关的监控，以全面覆盖测试场景，避免冗余地部署不同名称的相似监控。
- 部署过多的计数器可能会造成分析和性能开销负担过重。
- 务必不要忽视正确的系统配置 （例如虚拟内存大小）。尽管准确地说这不是监控规定的一部分，但它可能极大地影响测试结果。
- 决定有关远程计算机的策略。要么在每个远程计算机上定期运行监控服务以收集结果，然后在运行结束时将结果批量传输给管理员，要么连续收集度量并通过网络发送给管理员。根据测试中的应用程序和定义的性能目标选择策略。
- 设置阈值时，对任何测试和应用程序都要考虑由硬件和/或操作系统供应商设置的任何 “常规”建议 （例如，平均 CPU 利用率在一段时间内应该低于 80%，或者磁盘队列长度应小于 2）。

  这并不意味着不符合这些 “常规”建议总是错误的，但它确实意味着将监控结果和性能测试响应时间与其他度量一起检查，总是值得的。
- 选择将监控应用程序及其目标的最有价值活动的参数。数据太多可能导致分析进程负担过重。
- 监控目标不仅可以通过使用内置系统或应用程序对象和计数器达成，也可以通过查看特定于应用程序的日志、脚本、 XML 文件等达成。
- 这可能是个好主意：有少量基本的监控持续运行（例如，在 HP SiteScope 中），并为测试执行期间的性能测试场景定义更详细的监控。

不仅测量性能测试下的度量，也测量性能测试前后某些时段的度量，以创建“本地基线”并验证测试中的应用程序在性能测试完成后是否回到基线。

对监控的误解

性能监控的完整目标可能大致定义为收集度量数据供以后分析用，最终目标则是识别瓶颈的根源。

尽管这种说法通常并无争议，但这个目标可能产生一些常见的误解，产生高开销并增加成本。它们有：

- **监控基本的基础结构就足够了。**
  监控系统度量（如 CPU、内存和磁盘）很重要，但这些度量提供的信息对于了解实际用户或应用程序是否正遇到性能问题还不够。今天大多数性能问题的原因通常是应用程序组件而非某些硬件的问题。因此，系统监控本身仍很关键，但不能提供真实应用程序性能的准确或完整描述。

- **监控应用程序的进程或服务就足够了。**
  今天的应用程序，无论是封装的、J2EE、.NET 还是自定义 SOA 应用程序，都很复杂，并跨越多个系统和各种技术。为了透彻了解应用程序的运行状况，详细的组件监控和诊断对了解各种服务间的复杂交互是必需的。 HP Diagnostics 使您能从最终用户业务流程开始，然后深入到应用程序组件和系统层中，从而确保能够得到对业务影响最大的问题的快速解决方案，并满足服务水平协议的要求。

- **监控系统或应用程序的所有可用度量是最佳途径。**
  收集太多数据会导致分析负担加重，扭曲对真实性能问题的揭示。但 100% 的覆盖率并不需要，甚至也不可取。著名的 80/20 规则（80% 的问题通常是由 20% 的系统或应用程序组件造成的）对于性能监控也成立。解决办法在于了解哪些系统与关键的业务功能相关，哪些不相关。

- **所有测试都可以用同一组度量来完成。**
  尽管大多数性能测试很可能都会选择某些度量，但好的性能监控会根据执行的测试类别而包含不同的度量组。

- **监控 Web 服务器通常就足够了。**
  监控今天复杂的应用程序时，了解其体系结构对于了解性能因素的真实情况是必需的。标准的 Web 应用程序部署至少由 Web 服务器、应用程序服务器和数据库服务器组成，在多数情况下散布在多个物理计算机甚至多个物理位置上。随着 SOA 的涌现，更多的基础结构和服务可能会在生成最终用户响应时涉及到。因此监控所有相关服务器，尤其是数据库计算机，就非常重要了。有时可能还需要监控客户端工作站。

瓶颈和调整

为了让应用程序符合性能目标，就必须连续监控其性能。通过监控获取的性能数据可用于诊断类似生产的条件下出现的性能问题。此数据可能指示出瓶颈的存在，即整个系统的性能或容量受单个组件的严重限制的情况。

正规一些说，瓶颈位于系统的关键路径上，并提供最低吞吐量。在客户端 - 服务器系统尤其是基于 Web 的系统中，可能有许多缓慢点，如 CPU、内存、数据库、网络链接等等。其中某些可以通过监控操作系统的相关计数器识别，而另一些只能通过度量应用程序查明。

HP 提供的产品 HP Diagnostics for J2EE/.Net 允许 IT 专业人员：

- 主动检测生产中的问题。
- 迅速将问题隔离到系统或应用程序层。
- 查明特定应用程序组件的根源。

应用程序可能在开发和 QA 环境中表现很好，但在生产中无法扩展或出现性能问题。了解运行应用程序的基础结构的影响和许多应用程序组件在性能测试下交互时的行为很重要。从诊断角度，能按应用程序体系结构的层次和应用程序组件隔离问题、具有逐步深入 J2EE/.Net 性能问题和 J2EE/.Net 环境的可见性及带有足以判断问题根源的详细信息的实际逻辑，是很重要的。

而从业务角度，看到系统资源被充分利用是目标所在：毕竟所有这些 CPU 单元、大量内存和磁盘的开支是为了让它们尽可能忙碌。因此瓶颈的一个非正式定义是这样的情况：某资源已充分利用，*但*仍有进程 / 线程队列等待得到服务。

分布式环境尤其容易受到瓶颈的影响，这是因为：

- 每个应用程序组件都可能驻留在多个操作系统中。
- 组件之间的网络配置。
- 防火墙及其他安全措施。
- 数据库故障，其中糟糕的架构设计、缺少正确索引和存储分区都可能严重拖慢总体的系统响应时间。
- 低效的线程管理导致并发使用率下降。
- 未经验证的许多连接。

- 由于糟糕的线程池大小管理，线程数快速增长。
- 数据库连接池大小配置错误。
- 未优化而频繁使用的 SQL 语句。
- 无物理和共享内存调整，而这是大量事务处理所必需的。

如上所述，理想情况下性能监控能够识别并消除瓶颈和 / 或调整应用程序。

上面提到的 80/20 规则的另一个应用就是，在任何给定的应用程序中，20% 的操作消耗 80% 的资源。不用说，这些最频繁使用的操作最有可能是导致瓶颈的操作。因此改进这 20% 的代码能极大地降低对总体性能的要求。

性能调整的过程就自身而言部分是科学，部分是艺术，因为它可能涉及设计级别、编译级别、组装级别和运行时的干预。它的完成通常不可能没有取舍，优化时一般只有一两个方面能照顾到，例如：执行时间、内存使用、磁盘空间、带宽、能耗或某种其他资源。例如，缓存（和请求执行时间）增加会导致更大的内存消耗，多处理器的使用可能使源代码复杂化，等等。

2

HP 监控解决方案

HP 的组合产品包括许多用于处理监控所有方面的多用途监控解决方案。在性能验证领域中，HP LoadRunner 和 HP Performance Center 与这两个解决方案（HP SiteScope 和 HP Diagnostics）集成，便于实现全面和完整的监控和瓶颈分析解决方案。

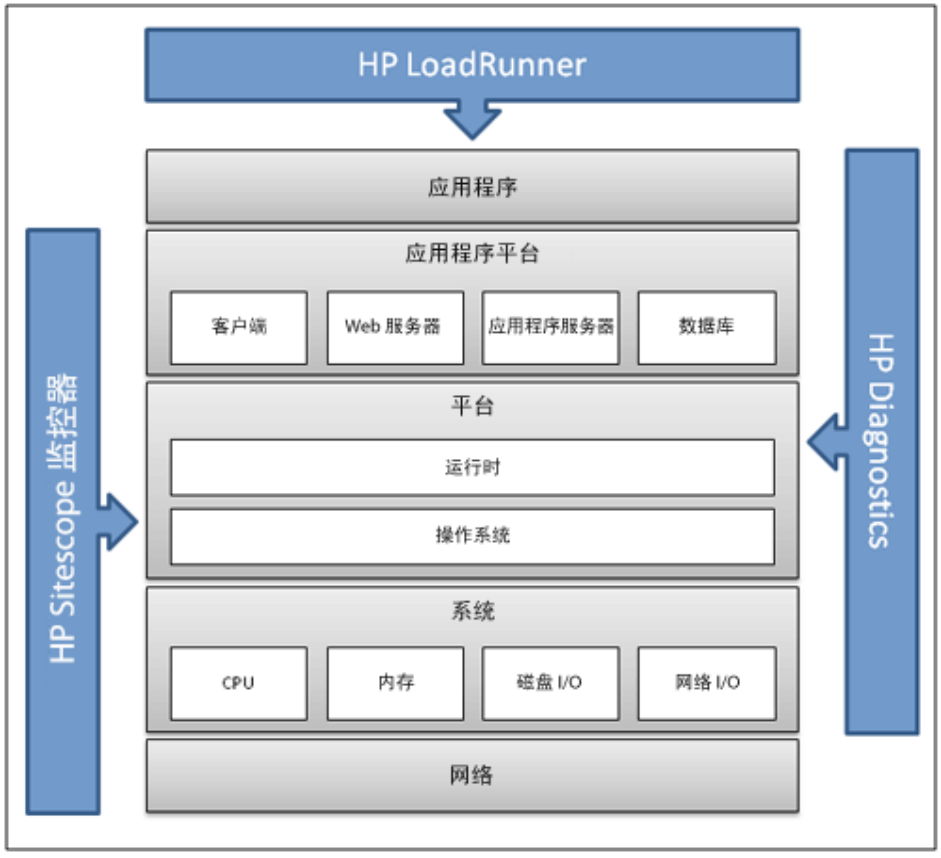
此章节包括：

概述

与 HP SiteScope 和 HP Diagnostics 集成时，LoadRunner 和 Performance Center 提供全面、完整和整体性的监控解决方案。这是通过结合以下两种产品的优点实现的：

- **LoadRunner 和 Performance Center。**通过模拟典型工作负载以及监控事务形式的用户操作来验证负载下的性能。
- **HP SiteScope。**监控测试中系统的不同层，收集有意义的数据以侧重于瓶颈分析过程。
- **HP Diagnostics。**通过将事务响应时间分解为不同的应用程序层来隔离出性能瓶颈，为问题的解决提供可操作的数据。

下图展示了用于测试中系统的不同层的 HP 监控解决方案：



从实践途径来说，相关计数器必须根据监控的特定类型选择。各种度量类型可分组为以下类别：

- **应用程序。**应用程序度量包括自定义性能计数器。
- **平台。**平台度量是关于 Microsoft Windows 上的 .NET 公共语言运行时 (CLR) 和 J2EE 环境中的 JVM 的。操作系统也视为平台。
- **系统。**系统度量是关于处理器、内存、磁盘 I/O 和网络 I/O 的。
- **网络。**网络度量是关于网络带宽使用情况和延迟的。

对于面向验证的测试，建议使用 LoadRunner 和 SiteScope 监控 AUT，以便识别事务响应时间或资源利用率中的潜在瓶颈。一旦识别出这样的瓶颈，则建议使用 HP Diagnostics，以使用更有针对性、更短的测试分离出问题所在，最后提供给开发团队可操作的数据。

对于面向优化的测试，建议从一开始就使用 HP Diagnostics，以便更快识别出潜在优化点。此途径最适合于这样的测试：压力测试、针对应用程序的小子系统测试运行、容量测试，等等。

HP LoadRunner

LoadRunner 和 Performance Center 包括涵盖性能测试直接需要的本机监控功能。

其中包括：

- **LoadRunner 数据点监控。**针对基于 Web 的应用程序运行时，包括由 VuGen 脚本生成的事务监控和自动生成的数据点，诸如每秒点击数、吞吐量等等。
- **测试中的系统监控。**包括应用程序相关度量，如系统资源、Web 服务器、数据库和网络度量。

运行性能测试时，LoadRunner 事务监控是应当应用的最重要的基本监控，因为它们反映端到端的用户体验。这样就能从业务角度进行事务验证，而这又有助专注于测试和瓶颈分析。建议使用 LoadRunner 的服务级别协议，根据性能目标度量实际性能。下图展示了事务标记为度量 Web 链接鼠标单击的 LoadRunner 脚本。

```
ManageResourcePool()
{
    web_reg_find("TEXT=Select a Resource Pool",LAST);
    lr_start_transaction("TM_ResourcePool_T01_ClickManageResourcePools");
    web_link("Manage Resource Pools",
        "Text=Manage Resource Pools",
        "Snapshot=t4.inf",
        LAST);
    lr_end_transaction("TM_ResourcePool_T01_ClickManageResourcePools", LR_AUTO);
    lr_think_time( 10 );
    web_reg_find("TEXT=Resource Requests",LAST);
    lr_start_transaction("TM_ResourcePool_T02_ClickOnResourcePool");
    web_link("ResPool_{UserNum}",
        "Text=ResPool_{UserNum}",
        "Snapshot=t5.inf",
        LAST);
    lr_end_transaction("TM_ResourcePool_T02_ClickOnResourcePool", LR_AUTO);
    .
    .
    .
```

事务计数器

所有事务计数器都以单事务和聚合值 （总计）的粒度提供。

计数器	描述
事务响应时间	不同负载下的不同响应时间值。 平均响应时间、最大值、百分比等等。
每秒事务数	每秒生成的事务数。
事务成功率	通过、失败或停止的事务数。

Web 资源相关计数器

其他基于数据点的监控由 LoadRunner 提供，与基于 Web 的应用程序相关。这些是评估应用程序维持模拟工作负载能力的重要计数器。每秒点击数

- 每秒点击数
- 吞吐量
- 每秒 HTTP 响应数
- 每秒下载的页数
- 连接数
- 每秒 SSL

LoadRunner 允许从 VuGen 脚本生成用户定义的数据点。这是功能非常强大的工具，帮助创建自定义并特定于环境的监控，而只需投入少量时间。这是用 VuGen 的 **lr_user_data_point** 函数完成的；度量值可从不同数据源捕获，然后在 LoadRunner Controller 或 Performance Center 联机图中显示，以及在 LoadRunner Analysis 中进行脱机调查并与其他度量关联。

下图展示 JBoss 自定义监控。VuGen 脚本配置为关联来自 JBoss 性能统计页面的数据。相关值随即报告至 Controller 中或 Performance Center 运行页上的“用户定义的数据点”图。

最后，如上所述，LoadRunner 和 Performance Center 还允许借助内置于产品中的本机监控或使用与 SiteScope 的集成，监控系统资源利用率、数据库、 Web 服务器、应用程序服务器等等。

HP SiteScope

LoadRunner 和 Performance Center 产品可配置为与 SiteScope 一起使用，后者是行业领先的监控解决方案，它可以作为独立产品运行，也可以作为各种 HP 产品的监控模块运行，如 HP HP Business Availability Center 和上面提到的性能测试解决方案。

SiteScope 是无代理程序监控解决方案，为确保分布式 IT 基础结构的可用性和性能而设计，如服务器、操作系统、网络设备、应用程序和应用程序组件。这一基于 Web 的基础结构监控解决方案是轻型的，可灵活自定义，并且不需要在生产系统上安装数据收集代理程序。

有了 SiteScope，您可以获得验证基础结构操作所需的实时信息，在发生问题时得到通知，并在问题变得严重之前解决瓶颈。SiteScope 还包括允许开发标准化监控组织和加速监控部署的模板。SiteScope 还有可用于以多种媒体交流和记录事件信息的警报类型。可自定义警报模板以满足组织的需要。

尽管 Performance Center 中的本机监控可覆盖大多数组织的一般需要，但唯有 SiteScope 具有庞大的监控器集和预封装的模板，设计为能满足所有可能的监控要求。不论操作系统度量还是应用程序服务器度量、各种 UNIX 风格或文件检查器， SiteScope 全有。

SiteScope 是业内第一个推出无代理程序监控解决方案的先驱。SiteScope 用户已受益于其业内公认的无代理程序监控体系结构。和基于代理程序的监控途径不同， SiteScope 通过以下方式降低总拥有成本：

- 收集基础结构组件的详细性能数据
- 通过将所有监控组件合并到中央服务器来减少维护的时间和成本
- 消除不稳定的代理程序影响系统性能的可能性

HP Diagnostics

HP Diagnostics 隔离应用程序性能问题，缩短应用程序性能瓶颈问题的平均解决时间 (MTTR)。它提供可操作信息来解决性能问题。

HP Diagnostics 扩展了 LoadRunner 和 Performance Center，能应对跨应用程序生命周期测试复杂的 J2EE、.NET、企业资源计划 (ERP) 和客户关系管理 (CRM) 应用程序的独特挑战。

HP Diagnostics 允许您：

- 在生命周期的较早阶段发现并解决更多问题
- 通过在应用程序投入使用之前发现最常见的应用程序问题来获得更高质量
- 收集具体数据以支持是否使用应用程序的决策
- 以能够快速解决问题的基于角色的可视性，在应用程序投入使用后管理和监控它们

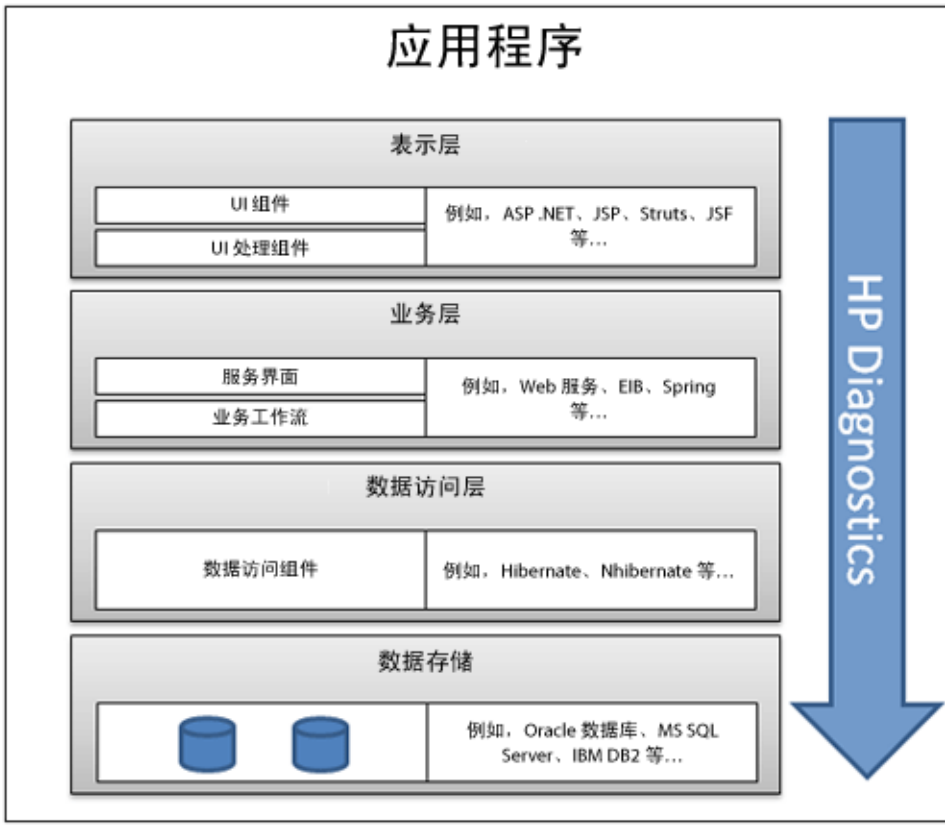
在性能测试期间， HP Diagnostics 在基础结构的所有层面从客户端跟踪 J2EE、.NET、 ERP 和 CRM 业务流程。模块随即将每个事务响应时间分解为在各层中和各组件中花费的时间。

您获得：

- 各层、组件、内存和 SQL 语句如何在负载条件下影响业务流程的总体性能的直观而简单易用的视图。在性能测试期间或之后，您可以通知应用程序团队应用程序规模不当，并向他们提供可操作数据。
- 能在业务环境下有效分流和找出问题，使团队能集中注意力于影响业务流程的问题。

- 能更方便地找出与测试中特定业务流程相关的组件。因为 J2EE、 ERP 和 CRM 应用程序可能使用数千种组件，这可能很有挑战性。HP Diagnostics 软件自动检测哪些组件在执行给定事务时是 “活动”的，并对其收集数据供分析用。业务流程未触及的组件被筛选掉，使您能专注于将工作完成而不是配置系统。

下图演示 HP Diagnostics 测量的应用程序层的示例：



HP Diagnostics 的关键功能和优势：

- 从缓慢的最终用户事务深入到瓶颈组件、方法或 SQL 语句，帮助解决内存、异常及其他常见问题
- 无需用户干预即可自动检测业务流程相关的所有组件并跟踪它们
- 在应用程序生命周期中提供完整的应用程序可见性，使应用程序投入使用时具有更高的质量
- 缩短在 J2EE、.NET、ERP 或 CRM（Siebel、Oracle、PeopleSoft 或 SAP）环境中解决问题的平均时间 (MTTR)
- 与 HP HP Business Availability Center、LoadRunner 和 Performance Center 完全集成

第 II 部分

操作系统

3

Windows 监控

Performance Center 提供处理 Windows 平台上运行的应用程序的性能测试行为的全面监控解决方案。

此章节包括：

概述

由于 IT 组织使用的绝大多数应用程序都基于 Windows，使用 Performance Center 使您能够使用 Windows 操作系统性能计数器跟踪测试中的应用程序行为。

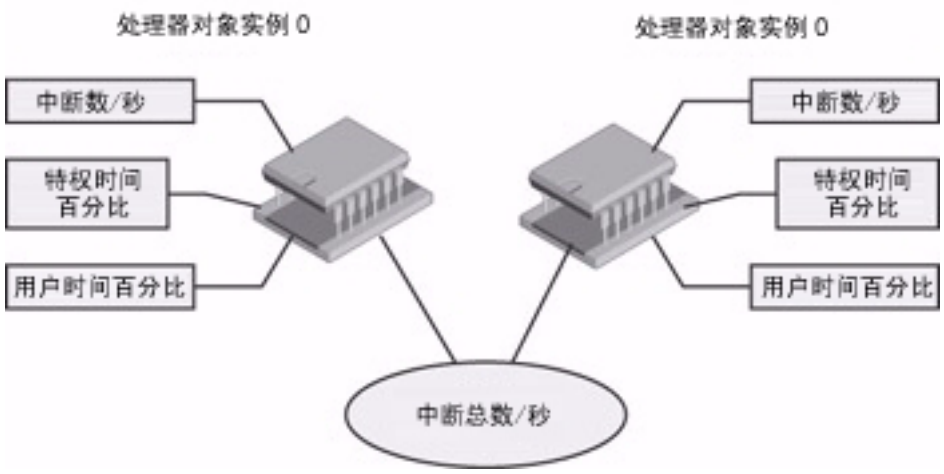
本章对由默认度量和默认阈值 （适用时）组成的几组预选监控器进行描述，所有这些都是使用各种来源（包括 HP 操作系统管理员、HP 专业服务组织、技术文档和业界专家著作）的最佳实践数据和专业知识研究而得。

现代 Windows 平台（从 Windows 2000 开始）提供各种内置工具可方便地收集、显示和重用与性能相关的信息。这些工具使用多种采样技术生成在性能问题诊断中非常有用的间隔性能监控数据。它们设计得极其高效，可以以最小影响连续运行。

体系结构

对象

性能相关的统计信息组织到**对象**中。例如，与总体处理器使用率相关的度量，比如**中断数 / 秒**和**用户时间百分比**，在处理器对象中提供。一个性能对象可能有一个或多个**实例**，每个实例都有名称以便唯一标识每个实例。例如，在有多个处理器的计算机上，每组处理器度量有多个实例。每个处理器性能计数器与处理器对象的特定名称的实例关联。实例名称是与该实例相关的一组计数器的唯一标识符，如下所示：



计数器

对每个度量间隔可用的各个性能统计信息都是数字性能**计数器**。您选择的每个性能计数器按其路径进行唯一标识，通常采用以下语法：

```
\\ 计算机名称 \Object(Parent/Instance#Index)\Counter
```

路径的**计算机名称**部分是可选的。

只有单个对象实例与其关联的简单对象 （比如系统或内存）采用以下语法：

```
\Object\Counter
```

计数器类型

每个计数器都有计数器类型。知道计数器类型很有用，因为它指示性能统计信息是如何得出的。

下面是一些最重要的计数器类型：

- **实例化计数器**。显示最近度量的简单数字值
- **间隔计数器。**显示随时间变化的活动速率
- **已用时间计数器**。按间隔收集，不能进行汇总
- **平均值计数器。**提供给定间隔派生的平均值

处理器 —— 最重要的计数器

程序执行线程消耗处理器 (CPU) 资源。这些线程可以是用户模式进程或操作系统内核的一部分。可用性能计数器度量线程和其他可执行工作单元耗用多少 CPU 处理时间。这些处理器利用率度量使您能确定哪些应用程序应对 CPU 占用率高负责。

计数器	描述
处理器时间百分比计数器	指示处理器执行非空闲线程所用的已用时间百分比
特权时间百分比计数器	指示进程线程以特权模式执行代码所用的已用时间百分比
中断时间百分比计数器	指示处理器在采样间隔内接收和处理硬件中断所用的时间
处理器队列长度计数器	指示处理器队列中的线程数
上下文切换次数计数器	指示计算机上的所有处理器从一个线程切换到另一个线程的综合速率
系统运行时间计数器	指示系统总体可用性的指示器

处理器时间百分比计数器

正式名称	处理器 (_Total)\ 处理器时间百分比计数器
计数器类型	间隔 （繁忙百分比）
描述	该间隔内的总体处理器平均利用率。处理器**不**在运行空闲线程的每个间隔，假设处理器代表某个实际工作负载而繁忙。
使用备注	总体处理器使用情况的主要指示器。值落在 0-100% 繁忙程度的范围内。处理器对象的 _Total 实例表示所有处理器利用率实例的**平均**总计值。
性能	确定处理器是否是潜在瓶颈的主要指示器。
操作	持续的 100% 利用率可能意味着失控的进程。通过查看进程 (n)\ 处理器时间百分比计数器确认失控进程线程是否在无限循环，以作进一步调查。
阈值	对于面向响应的工作负载，利用率持续高于 80-90% 时应该小心。对于面向吞吐量的工作负载，除容量约束以外，长时间的高利用率一般不用担心。
相关度量	➤ 处理器 (_Total)\ 特权时间百分比（请参见（第 49 页）） ➤ 处理器 (_Total)\ 用户时间百分比 ➤ 处理器 (n)\ 处理器时间百分比 ➤ 进程 (n)\ 处理器时间线程百分比 (n/索引号)\ 处理器时间百分比

注：在计算机上观察到处理器的利用率很高并不总是指示存在需要解决的问题。如果与处理器相关的其他计数器（如**特权时间百分比**或**处理器队列长度**）正在线性增加，则可能需要调查 CPU 利用率高的原因。

特权时间百分比计数器

正式名称	处理器 (_Total)\ 特权时间百分比计数器
计数器类型	间隔 （繁忙百分比）
描述	在间隔内以特权或内核模式发生的总体处理器利用率平均值。所有操作系统函数都以特权模式运行。特权模式包括涉及启动设备输入/输出操作和启动用于完成中断处理的延迟过程调用的设备驱动程序代码。
使用备注	处理器对象的 _Total 实例表示所有处理器利用率实例的**平均**总计值。**特权时间百分比**与总体**处理器时间百分比**的比率 （特权模式比率）与工作负载相关。
性能	确定操作系统是否功能正常 （包括设备驱动程序是否正常）的辅助指示器，负责潜在的处理器瓶颈。
操作	失控进程线程在无限循环时，处理器的状态可以指出问题是否还牵涉系统模块。
阈值	值始终高于 75 % 指示存在瓶颈。
相关度量	➤ 处理器 (_Total)\ 中断时间百分比 ➤ 处理器 (_Total)\DPC 时间百分比 ➤ 进程 (n)\ 特权时间百分比

注：特权模式比率不分好坏。但是，如果相同工作负载的此比率突然变化，则应该查明其原因。

中断时间百分比计数器

正式名称	处理器 (_Total)\ 中断时间百分比计数器
计数器类型	间隔 （繁忙百分比）。
描述	该间隔的中断模式下发生的总体处理器平均利用率。只有中断服务例程 (ISR) 作为设备驱动程序的函数在中断模式中运行。
使用备注	处理器对象的 _Total 实例表示所有处理器利用率实例的**平均**总计值。ISR 中断处理是最高优先级的处理。中断处理是系统函数，没有关联的进程。过高的中断时间百分比可能指示设备发生故障但无法指明是哪个设备。使用 Kernrate（内核调试器）可以确定分派最频繁的 ISR。
性能	此计数器指示处理器接收和处理硬件中断所用的时间百分比。此值是生成中断的设备活动的间接指示，如网络适配器。此计数器显著增加表示有潜在硬件问题。
操作	确定故障设备是否正引起潜在处理器瓶颈的辅助指示器。
阈值	取决于处理器。
相关度量	➤ 处理器 (_Total)\ 中断数 / 秒 ➤ 处理器 (_Total)\DPC 时间百分比 ➤ 处理器 (_Total)\ 特权时间百分比

处理器队列长度计数器

正式名称	系统 \ 处理器队列长度计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	观察到在处理器就绪队列中延迟并等待计划执行的线程数。处理器 “就绪队列”中等待的线程按优先级排序，处理器空闲时，就计划让优先级最高的线程运行。
使用备注	许多程序线程在自愿等待状态中休眠。活动线程子集设定了可观察到的处理器队列长度的实际上限。
性能	确定处理器是否是潜在瓶颈的重要辅助指示器。
操作	容量约束可能正导致过多应用程序延迟的指示。
阈值	在处理器非常繁忙的单处理器计算机上，重复观察到处理器队列长度 > 5 则警示工作量经常超出处理器的正常处理能力。此外，就绪队列长度 > 10，同时处理器利用率也接近饱和，也有力表明处理器约束。在多处理器计算机上，将处理器队列长度除以物理处理器数。在配置为使用硬处理器相似性进行不对称运行的多处理器计算机上，较大的处理器队列长度值可能是一个不平衡配置的信号。
相关度量	线程 （父进程 \ 索引号） \ 线程状态

上下文切换次数计数器

正式名称	系统 \ 上下文切换次数 / 秒计数器
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	一个运行中的线程由另一个代替时，就发生上下文切换。因为 Windows 支持多线程操作，所以上下文切换是系统的正常行为。当用户模式线程调用任何特权操作系统函数时，在用户模式线程和以特权模式执行被调用函数的相应内核模式线程之间发生上下文切换。
使用备注	上下文切换是正常的系统功能，上下文切换率是工作负载的副产品。上下文切换率高通常并不表示有问题。它也并不意味着 CPU 容量不足。此外，系统管理员通常对发生的上下文切换率也无能为力。相对于历史正常值，每秒上下文切换率大幅增加可能反映出有问题，例如设备故障。请将上下文切换次数 / 秒与其通常相关的处理器 (_Total)\ 中断数 / 秒计数器进行比较。
性能	上下文切换率很高通常表示应用程序设计有问题，也可能预示着可扩展性方面的难题。
操作	上下文切换发生在以下情况中：高优先级的线程抢占了正在运行的低优先级线程，或者高优先级线程阻塞。当很多线程有着相同优先级级别时，可能会发生很高的上下文切换率。这通常指示系统上有太多线程争用处理器。如果未看到很高的处理器利用率，但看到非常低的上下文切换率，则可能指示线程被阻止

阈值	当与历史标准值存在极大偏差时生成重要服务器计算机的警报。一般而言，上下文切换率小于 5,000/ 秒 / 处理器则不必担心。如果上下文切换率**超过** 15,000/ 秒 / 处理器，则存在约束。
相关度量	线程 \ 上下文切换次数 / 秒。

系统运行时间计数器

正式名称	系统 \ 系统运行时间计数器
计数器类型	已用时间。
描述	显示计算机自上次重新启动以来一直在运行的时间（秒）。
使用备注	系统可用性的主要指示器。
性能	N/A
操作	报告系统可用性。
阈值	N/A
相关度量	处理器 (n)\ 已用时间

注：在度量性能之前，确保服务器和服务器应用程序已启动、正在运行并可以使用。

内存 —— 最重要的计数器

Windows 维护物理和虚拟内存。当向磁盘过度分页耗用太多可用磁盘带宽时，RAM 不足常常会间接导致磁盘性能问题。因此，磁盘分页率是重要的内存性能指示器。在 32 位系统上，虚拟内存限制为 4 GB，其中 2 GB 为专用区域，2 GB 为共享区域。拥有大量物理内存并不能防止虚拟内存不足，反而在应用程序在使用分配的内存后不释放该内存时可能会因**内存泄漏**而导致致命崩溃。

观察到可用 RAM 不足时，重要的是确定如何使用分配的物理内存，并对称为其工作集的有问题进程的驻留页数进行计数。

计数器	描述
可用字节数计数器	指示可用于正在计算机上运行的进程的物理内存量
工作集计数器	指示每个进程的驻留页数
页数 / 秒计数器	指示从磁盘读取页或向磁盘写入页以解决硬页故障的速率
页读取数 / 秒计数器	指示进程工作集对物理内存来说太大而正在向磁盘分页
未分页池字节数计数器	指示无法写入到磁盘而在分配后必须保留在物理内存中的对象的系统内存区域大小 （由操作系统使用的物理内存大小）
分页池字节数计数器	指示内存泄漏
分页池失败次数计数器	指示从分页池进行的分配失败的次数
缓存字节数计数器	指示静态文件缓存的大小

计数器	描述
系统缓存驻留字节数计数器	指示分配到系统文件缓存的驻留页数
已提交字节数计数器	指示导致慢和不正常响应时间的极端分页

可用字节数计数器

正式名称	内存 \ 可用字节数计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	进程的驻留页集合。RAM 中此进程可以处理而不会导致分页故障的分配页数。
使用备注	其计算方法是在零、可用和备用内存列表上添加空间量。可用内存是可供使用的内存；零内存是填充零以防止后面进程看到前面进程使用的数据的内存页；备用内存是从进程的工作集 （其物理内存）中中途移除到磁盘上但仍可再次调用的内存。
性能	如果内存不足，则进程 (n)\ 工作集会告诉您每个进程正在使用多少 RAM。
操作	N/A
阈值	值持续小于安装的 RAM 的 20ñ25% 则指示内存不足。
相关度量	➤ 内存 \ 可用字节数 ➤ 内存 \ 已提交字节数 ➤ 进程 (n)\ 专用字节数 ➤ 进程 (n)\ 虚拟字节数 ➤ 进程 (n)\ 池分页字节数

工作集计数器

正式名称	进程 (*)\ 工作集计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	进程的驻留页集合。RAM 中此进程可以处理而不会导致分页故障的分配页数。
使用备注	进程 (n)\ 工作集跟踪活动进程当前使用的 RAM。某些服务器应用程序（比如 IIS 、Exchange 和 SQL Server）管理其自己的进程工作集。监控进程对象中的进程 (_Total)\ 工作集以查看所有进程地址空间中是如何分配 RAM 的。
性能	如果内存不足，则进程 (n)\ 工作集会告诉您每个进程正在使用多少 RAM。
操作	N/A
阈值	持续增加 10% 或更多则警告物理内存受限。
相关度量	➤ 内存 \ 可用字节数 ➤ 内存 \ 已提交字节数 ➤ 进程 (n)\ 专用字节数 ➤ 进程 (n)\ 虚拟字节数 ➤ 进程 (n)\ 池分页字节数

页数 / 秒计数器

正式名称	内存 \ 页数 / 秒计数器
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	该间隔内磁盘的分页操作数。页数 / 秒是页读取数 / 秒和页写入数 / 秒的总和。
使用备注	页读取数 / 秒计数器指示硬页故障。正在运行的线程已引用虚拟内存中不在进程工作集中的页。它也不是标记为正在转换的整理页，而是仍驻留在内存中的页。线程从磁盘获取页时延迟了 I/O 操作的持续时间那么长时间。操作系统将页从磁盘复制到 RAM 中的可用页，然后重新分派线程。
性能	确定物理内存是否是潜在瓶颈的主要指示器。
操作	过多的分页可导致响应时间缓慢和不稳定。
阈值	当每个分页磁盘的页数 / 秒超过 50 时请小心。
相关度量	➤ 内存 \ 可用字节数 ➤ 内存 \ 已提交字节数 ➤ 进程 (n)\ 工作集

注： 过高的分页率通常可通过增加 RAM 来降低。磁盘带宽有限。用于分页操作的容量对其他面向应用程序的文件操作不可用。

页读取数 / 秒计数器

正式名称	内存 \ 页读取数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	此计数器指示进程工作集对物理内存来说太大而正在向磁盘分页。它显示读取操作数，而不考虑每次操作中检索的页面数。较高的值表示有内存瓶颈。
使用备注	如果页读取操作率较低，而物理磁盘 \ 磁盘时间百分比和物理磁盘\平均 磁盘队列长度较大，则可能存在磁盘瓶颈。如果队列长度增加时页读取率不减少，则存在内存不足。
性能	确定物理内存是否是潜在瓶颈的主要指示器。
操作	过多的分页可导致响应时间缓慢和不稳定。
阈值	值持续超过 5 指示读取请求存在大量页故障。
相关度量	➤ 内存 \ 页数 / 秒 ➤ 内存 \ 页写入数 / 秒

未分页池字节数计数器

正式名称	内存 \ 未分页池字节数计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	从未分页池分配的页始终驻留在 RAM 中。
使用备注	有关每个 TCP 连接的状态信息都存储在未分页池中。除以页的大小计算出已分配的页数。
性能	如果内存不足，未分页池字节数会告诉您正在使用多少不可分页的 RAM 系统函数。
操作	N/A
阈值	观察内存 \ 未分页池字节数的值比在系统启动时的值是否高出 10% 或更多。如果确实高出这么多，则发生重大内存泄漏。
相关度量	➤ 池分页字节数 ➤ 池分页驻留字节数 ➤ 系统缓存驻留字节数 ➤ 系统代码驻留字节数 ➤ 系统驱动程序驻留字节数 ➤ 进程 (_Total)\ 工作集

分页池字节数计数器

正式名称	内存 \ 分页池字节数计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	系统分页池中已提交的虚拟内存页的数量。系统函数从分页池中分配可将页调出的虚拟内存页。进程调用的系统函数也从分页池中分配虚拟内存页。
使用备注	内存\分页池字节数报告在系统分页池中分配了多少虚拟内存。内存 \ 分页池驻留字节数是当前驻留在 RAM 中的分页池页数。其余是调出的页数。
性能	N/A
操作	主要用于识别正在泄漏内存的进程。
阈值	特定进程的进程 (n)\ 分页池字节数增加超过 10% 可能指示泄漏内存行为。
相关度量	➤ 内存 \ 提交限制 ➤ 内存 \ 正在使用的已提交字节数百分比 ➤ 进程 (n)\ 池分页字节数 ➤ 进程 (n)\ 虚拟字节数

注：某些逃逸进程可能会泄漏系统分页池中的内存。进程 (n)\ 分页池字节数计数器帮助您识别这些泄漏应用程序。

分页池失败次数计数器

正式名称	服务器 \ 分页池失败次数计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	服务器服务自初始化以来经历的分页池分配失败累计次数。
使用备注	文件服务器服务有很多从分页池分配虚拟内存页的函数。如果内存泄漏耗尽分页池，文件服务器服务可能遇到从分页池分配虚拟内存的困难。如果分配虚拟内存的调用失败，则文件服务器服务从这些失败中完美恢复，并进行报告。很多其他应用程序和系统函数不从虚拟内存分配失败中完美恢复，所以此计数器可能是内存泄露导致这些分配失败的唯一可靠指示器。
性能	N/A
操作	主要用于识别分页池中虚拟内存不足。
阈值	此计数器的任何非零值都指示存在瓶颈。
相关度量	➤ 内存 \ 池分页字节数 ➤ 内存 \ 提交限制 ➤ 内存 \ 正在使用的已提交字节数百分比 ➤ 服务器 \ 池分页字节数 ➤ 进程 (n)\ 池分页字节数

缓存字节数计数器

正式名称	内存 \ 缓存字节数计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	系统工作集中的驻留页的集合。RAM 中内核线程可以处理而不会导致分页故障的分配页数。
使用备注	系统工作集像任何其他工作集一样与页替换有关。
性能	如果内存不足，缓存字节数告诉您正在使用多少可分页的 RAM 系统函数。
操作	N/A
阈值	N/A
相关度量	➤ 未分页池字节数 ➤ 池分页驻留字节数 ➤ 系统缓存驻留字节数 ➤ 系统代码驻留字节数 ➤ 系统驱动程序驻留字节数 ➤ 进程 (_Total)\ 工作集

系统缓存驻留字节数计数器

正式名称	内存 \ 系统缓存驻留字节数计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	分配到系统文件缓存的驻留页数。此计数器跟踪文件缓存中当前驻留在 RAM 中的虚拟内存页数。
使用备注	在文件打印和服务器上，系统缓存驻留字节通常是 RAM 的最大耗用者。它是系统工作集（缓存字节数）的一部分，与可用字节数减少时的页整理有关。
性能	当系统文件缓存效率低时，会影响依赖缓存的服务器应用程序的性能。其中包括服务器、重定向器、 NTFS 和 IIS。
操作	主要用于识别正在泄漏内存的进程。
阈值	N/A
相关度量	内存 \ 缓存字节数

已提交字节数计数器

正式名称	内存 \ 已提交字节数计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	已提交的虚拟内存页数。已提交的页必须由 RAM 中的物理页或分页文件上的插槽支持。
使用备注	已提交的字节数报告总共分配了多少虚拟内存进程地址空间。如果已提交字节数与 RAM 的比率 > 1，表示虚拟内存超过 RAM 的大小，需要执行某些内存管理。当已提交的字节数与 RAM 的比率超过 1.5 时，到磁盘的分页通常最大可增加到分页磁盘的带宽设置的限制。
性能	已提交字节数与 RAM 的比率是实际内存不足的次要指示器。
操作	过多的分页可导致响应时间缓慢和不稳定。
阈值	当已提交的字节数与 RAM 的比率超过 1.5 时，无疑指示实际内存瓶颈。
相关度量	➤ 内存 \ 页数 / 秒 ➤ 内存 \ 提交限制 ➤ 内存 \ 正在使用的已提交字节数百分比 ➤ 内存 \ 池分页字节数 ➤ 进程 (n)\ 专用字节进程 (n)\ 虚拟字节数

注：如果已提交的字节数与 RAM 的比率接近或超过 1.5，则增加内存是不可避免的。

I/O —— 最重要计数器

通过 I/O 管理器堆栈，Windows 维护物理和逻辑磁盘操作。**逻辑磁盘**用唯一的驱动器盘符代表单个文件系统。**物理磁盘**是特定存储设备的内部表示 —— 即 SCSI、RAID、SATA 或其他技术。使用阵列 Controller 或 RAID 这样的复杂存储系统时，底层物理磁盘硬件的特性并不直接对操作系统可见。这些特性（即磁盘数、磁盘速度、查找时间、转速和位密度，以及某些优化功能如板载内存缓冲区）都对性能有重大影响。内存缓冲区和命令排队之类的高级功能可以将性能提升 25-50%。

主动采取措施提升磁盘性能很重要，因为它下降趋势很快，特别是发生磁盘分页活动时。

计数器	描述
平均值 磁盘秒数 / 传输计数器	指示物理磁盘的潜在瓶颈
空闲时间百分比计数器	指示物理磁盘利用率
磁盘传输次数 / 秒计数器	指示物理磁盘是否是潜在瓶颈
平均值 磁盘队列长度计数器	与其他计数器结合使用，指示磁盘的潜在瓶颈
拆分 IO/ 秒计数器	指示可能碎片整理
可用 MB 计数器	指示逻辑磁盘空间使用情况

平均值 磁盘秒数 / 传输计数器

正式名称	物理磁盘 (n)\ 平均 磁盘秒数 / 传输计数器
计数器类型	平均值
描述	间隔内物理磁盘请求的总体响应时间平均值。平均磁盘秒数 / 传输包括设备服务时间和队列时间。
使用备注	物理磁盘 I/O 性能的主要指示器。性能依赖于底层磁盘配置，此配置对操作系统是透明的。单个磁盘的性能特征根据查找时间、转速、记录密度和接口速度而各有不同。更昂贵的面向性能的磁盘可以使性能提高 50%。
性能	确定磁盘是否是潜在瓶颈的主要指示器。
操作	磁盘响应时间过久会延长应用程序响应时间。
阈值	取决于基础磁盘硬件，但通常不应该超过 18 毫秒。
相关度量	➤ 物理磁盘 (n)\ 磁盘传输次数 / 秒 ➤ 物理磁盘 (n)\ 空闲时间百分比 ➤ 物理磁盘 (n)\ 当前磁盘队列长度

注：此计数器可能指示大量磁盘碎片、磁盘缓慢或磁盘故障。乘以**物理磁盘 \ 平均磁盘秒数 / 传输**和**内存 \ 页数 / 秒**计数器的值。如果这些计数器的输出超过 0.1，则分页正在耗用超过 10% 的磁盘访问时间，因此需要更多 RAM。

空闲时间百分比计数器

正式名称	物理磁盘 (n)\ 空闲时间百分比计数器
计数器类型	间隔 （繁忙百分比）。
描述	间隔内磁盘空闲的时间百分比。 100% 减去空闲时间百分比得出磁盘利用率。
使用备注	磁盘利用率的计算方法如下： **物理磁盘 (n)\磁盘利用率 = 100% – 物理磁盘 (n)\空闲时间百分比**。 对于磁盘阵列，将磁盘利用率除以阵列中的磁盘数估计出单个磁盘的利用率。当磁盘利用率接近 100% 时，假定独立到达磁盘，队列时间将以指数级增加。
性能	确定物理磁盘是否超载并成为潜在瓶颈的主要指示器。
操作	队列时间增加会导致磁盘响应时间延长，进而延长应用程序响应时间。
阈值	当空闲时间百分比小于 20% 时发出警告。
相关度量	➤ 物理磁盘 (n)\ 平均 ➤ 磁盘秒数 / 传输 ➤ 物理磁盘 (n)\ 磁盘传输次数 / 秒 ➤ 物理磁盘 (n)\ 当前磁盘队列长度

注： 计算磁盘利用率、磁盘服务时间和磁盘队列时间以确定磁盘子系统是否性能不佳和 / 或磁盘超载。

磁盘传输次数 / 秒计数器

正式名称	物理磁盘 (n)\ 磁盘传输次数 / 秒计数器
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	该间隔内完成物理磁盘请求的速率。
使用备注	物理磁盘 I/O 活动的主要指示器。也称为磁盘到达率。还可细分为读取数和写入数： **物理磁盘 (n)\ 磁盘传输次数 / 秒 = 物理磁盘 (n)\ 磁盘读取数 / 秒 + 物理磁盘 (n)\ 磁盘写入数 / 秒** 通过应用利用率法则根据空闲时间百分比计算磁盘服务时间。
性能	确定磁盘是否是潜在瓶颈的主要指示器。
操作	磁盘响应时间过久会延长应用程序响应时间。
阈值	取决于底层磁盘硬件。
相关度量	➤ 物理磁盘 (n)\ 磁盘传输次数 / 秒 ➤ 物理磁盘 (n)\ 空闲时间百分比 ➤ 物理磁盘 (n)\ 当前磁盘队列长度

平均值磁盘队列长度计数器

正式名称	物理磁盘 (n)\ 平均 磁盘队列长度计数器
计数器类型	复合计数器。
描述	估计的正在服务或等待磁盘服务的物理磁盘请求平均数。
使用备注	需要认真解释的物理磁盘 I/O 排队的次要指示器。平均值磁盘队列长度计数器的值应该基于对基础物理磁盘实体的特性的理解进行解释。实际上，对作为单个物理磁盘的主机操作系统显示的可能是作为单个 LUN 的物理磁盘的集合。阵列 Controller 通常用于创建由多个物理磁盘支持的虚拟 LUN。使用阵列 Controller，阵列中的多个磁盘可以执行并发操作。在这些环境下，不应该再将物理磁盘实体视为单个服务器。 磁盘读取时间百分比、磁盘时间百分比和磁盘写入时间百分比使用相同公式推导出来，区别在于它们报告的值以 100% 为上限。
性能	确定磁盘是否是潜在瓶颈的次要指示器。
操作	N/A
阈值	不应该高于心轴数加 2。
相关度量	➤ 物理磁盘 (n)\ 空闲时间百分比 ➤ 物理磁盘 (n)\ 平均磁盘秒数 / 传输 ➤ 物理磁盘 (n)\ 磁盘传输次数 / 秒 ➤ 物理磁盘 (n)\ 当前磁盘队列长度 ➤ 物理磁盘 (n)\ 磁盘时间百分比

拆分 IO/ 秒计数器

正式名称	物理磁盘 (n)\ 拆分 IO/ 秒计数器
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	间隔内物理磁盘请求拆分为多个磁盘请求的速率。请注意，发生拆分 I/O 时， I/O 管理器度量层既计数拆分 I/O 时的原始 I/O 请求，也计数拆分 I/O 请求，因此拆分 I/O 计数准确反映由 I/O 管理器启动的 I/O 操作数。
使用备注	物理磁盘碎片的主要指示器。 当请求的数据的大小太大而无法适合单个 I/O 时也会导致拆分 I/O。拆分 I/O 通常延长磁盘服务时间，因此还请观察与物理磁盘 (n)\ 平均 磁盘秒数 / 传输的相关性。
性能	帮助您确定需要多久运行一次磁盘碎片整理软件的次要指示器。
操作	磁盘响应时间过久会延长应用程序响应时间。
阈值	当拆分 I/O 所用时间超过磁盘传输次数 / 秒的 20% 时发出警告。
相关度量	➤ 物理磁盘 (n)\ 磁盘传输次数 / 秒 ➤ 物理磁盘 (n)\ 平均 磁盘秒数 / 传输 ➤ 物理磁盘 (n)\ 空闲时间百分比

注： 定期整理磁盘碎片或当拆分 I/O 的次数过多时整理磁盘碎片，通常能提高磁盘性能，因为磁盘处理连续操作的速度远远快于处理随机请求的速度。

可用 MB 计数器

正式名称	逻辑磁盘 (n)\ 可用 MB 计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	逻辑磁盘上未分配的空间量，单位是 MB。 因为为非常大的文件系统计算可用 MB 十分耗时，I/O 管理器度量层大约每 5 分钟重新计算一次计数器的值。
使用备注	已用的逻辑盘空间容量的主要指示器。
性能	N/A
操作	用尽文件系统上的空间通常会导致灾难性后果。
阈值	当达到此计数器值或当逻辑磁盘 (n)\可用空间百分比 < 10% 时发出警报。
相关度量	逻辑磁盘 (n)\ 可用空间百分比。

网络 —— 最重要的计数器

Windows 中的网络流量在最低级别的硬件接口上和较高级别的网络协议下（如 TCP/IP）测量。网络接口统计信息由嵌入网络接口驱动程序层中的软件收集。此软件计算发送和接收的数据包数。生成网络接口对象的多个实例，为每个安装的网络接口芯片或卡生成一个网络接口对象实例。每个支持的协议，如 TCP、UDP、NetBEUI、NWLink IPX、NWLink NetBIOS、NWLink SPX 等，都可以使用较高级别的计数器，如协议对象 \ 接收的段数 / 秒和协议对象 \ 发送的段数 / 秒。

计数器	描述
总字节数 / 秒计数器	指示总吞吐量
服务器总字节数 / 秒	指示网络的总体服务器利用率
数据报数 / 秒计数器	指示 IP 协议负载
建立的连接数计数器	指示 TCP 协议连接成功率
接收的段数 / 秒计数器	指示接收的 TCP 数据段数
中断时间百分比计数器	指示处理器在硬件设备中断（如网卡）上所用的时间

总字节数 / 秒计数器

正式名称	网络接口 (n)\ 总字节数 / 秒计数器
计数器类型	间隔差异计数器 （比率 / 秒）
描述	间隔内每秒通过此接口传输和接收的总字节数。这是通过此接口的吞吐量 （字节）。
使用备注	网络接口流量的主要指示器。计算网络接口利用率： **网络接口 (n)\ 繁忙百分比 = 网络接口 (n)\ 总字节数 / 秒 ÷ 网络接口 (n)\ 当前带宽** 切换的链接上最大可获得的带宽应该接近于当前带宽计数器的 90-95%。
性能	确定网络是否是潜在瓶颈的主要指示器。
阈值	当总字节数 / 秒超过线路容量的 80% 时发出警告。
相关度量	➤ 网络接口 (n)\ 接收的字节数 / 秒 ➤ 网络接口 (n)\ 发送的字节数 / 秒 ➤ 网络接口 (n)\ 接收的数据包数 / 秒 ➤ 网络接口 (n)\ 发送的数据包数 / 秒 ➤ 网络接口 (n)\ 当前带宽

注：此计数器帮助您识别特定网络适配器上的流量是否饱和，以及是否需要再添加一个网络适配器。

服务器总字节数 / 秒

正式名称	服务器 \ 总字节数 / 秒计数器
计数器类型	间隔差异计数器 （比率 / 秒）
描述	服务器从网络接收和向网络发送的字节数。此值提供服务器如何繁忙的总体指示。
使用备注	此计数器指示通过网络发送和接收的字节数。较高值表示网络带宽是瓶颈。如果所有服务器的总字节数 / 秒的总和大约等于网络的最大传输率，则需要对网络分段
性能	确定网络是否是潜在瓶颈的主要指示器。
阈值	值不应该超过网络容量的 50%。
相关度量	网络接口 (n)\ 接收的字节数 / 秒

数据报数 / 秒计数器

正式名称	IPvn\ 数据报数 / 秒计数器
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	间隔内每秒传输和接收的 IP 数据报总数。
使用备注	IP 流量的主要指示器。
性能	确定网络是否是潜在瓶颈的次要指示器。
操作	IP 流量突然升高可能指示存在入侵者。
阈值	意外升高超过 10% 可能指示超载或安全漏洞。
相关度量	➤ IPvn\ 接收的数据报数 / 秒 ➤ IPvn\ 发送的数据报数 / 秒 ➤ 网络接口 (n)\ 数据包数 / 秒

建立的连接数计数器

正式名称	TCPvn\ 建立的连接数计数器
计数器类型	瞬时 （每个度量时段采样一次）。
描述	度量间隔结束时处于 “已建立” 状态的 TCP 连接的总数。
使用备注	TCP 会话连接行为的主要指示器。 可建立的 TCP 连接数受未分页池大小的约束。当未分页池耗尽时，无法建立新连接。
性能	确定网络是否是潜在瓶颈的次要指示器。
操作	TCP 连接数突然增加可能指示拒绝服务攻击。
阈值	意外升高超过 10% 可能指示超载或安全漏洞。
相关度量	➤ TCPPvn\ 接收的段数 / 秒 ➤ TCPPvn\ 发送的段数 / 秒 ➤ 网络接口 (n)\ 数据包数 / 秒 ➤ 内存 \ 未分页池字节数

接收的段数 / 秒计数器

正式名称	TCPvn\ 接收的段数 / 秒计数器
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	度量间隔内通过各个建立的连接接收的 TCP 段数的平均值。
使用备注	TCP 网络负载的主要指示器。 计算每个连接接收的平均段数： TCPvn\ 接收的段数 / 秒 ÷ TCPPvn\ 建立的连接数 / 秒 这可用于预测随用户数增长的未来负载。
性能	确定网络是否是潜在瓶颈的次要指示器。
操作	接收到的 TCP 请求数的突然增加可能指示存在入侵者。
阈值	意外升高超过 10% 可能指示超载或安全漏洞。
相关度量	➤ TCPPvn\ 建立的连接数 / 秒 ➤ TCPPvn\ 发送的段数 / 秒 ➤ IPvn\ 接收的数据报数 / 秒 ➤ 网络接口 (n)\ 数据包数 / 秒

中断时间百分比计数器

正式名称	处理器 (_Total)\ 中断时间百分比计数器
计数器类型	间隔 （繁忙百分比）。
描述	该间隔的中断模式下发生的总体处理器平均利用率。只有中断服务例程 (ISR) 作为设备驱动程序的函数在中断模式中运行。
使用备注	处理器对象的 _Total 实例表示所有处理器利用率实例的**平均**总计值。ISR 中断处理是最高优先级的处理。中断处理是系统函数，没有关联的进程。过高的中断时间百分比可能指示设备发生故障但无法指明是哪个设备。使用 Kernrate（内核调试器）可以确定分派最频繁的 ISR。
性能	此计数器指示处理器接收和处理硬件中断所用的时间百分比。此值是生成中断的设备活动的间接指示，如网络适配器。此计数器显著增加表示有潜在硬件问题。
操作	确定故障设备是否正引起潜在处理器瓶颈的辅助指示器。
阈值	取决于处理器。
相关度量	➤ 处理器 (_Total)\ 中断数 / 秒 ➤ 处理器 (_Total)\DPC 时间百分比 ➤ 处理器 (_Total)\ 特权时间百分比

4

监控 Unix

HP Performance Center 提供全面的监控解决方案，可应对各种 Unix 平台上运行的应用程序行为的性能测试。

此章节包括：

概述

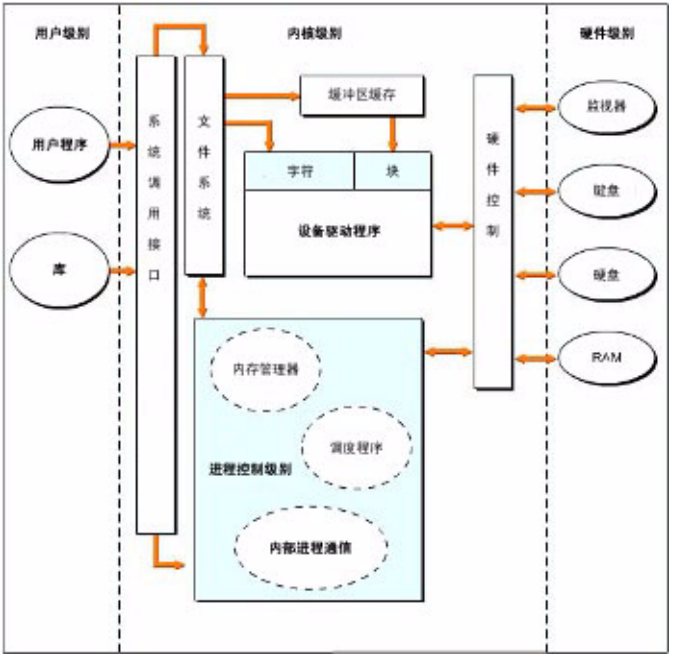
尽管基于 Windows 的系统和应用程序无可争议地占了大多数，但仍有大量过去和当前的应用程序是在 UNIX 平台上生成的。除 HP/UX、 Sun Solaris 和 IBM AIX 这样受青睐并广为人知的 UNIX 风格操作系统外，Linux 的快速扩张也已导致热门应用程序在 Unix 上创建及向其移植，后者以稳定性和可扩展性闻名。UNIX/Linux 还成为基于 J2EE 的系统的主要平台，从 Apache Web 服务器到 WebSphere 应用程序服务器，直至 Oracle 数据库服务器。

因此，HP LoadRunner 和 HP Performance Center 包含访问 UNIX 操作系统性能计数器的工具以用于跟踪测试中的应用程序行为，也就不奇怪了。

尽管 UNIX 风格的操作系统可能在特定命令及其选项上有所不同，但它们都提供各种内置工具来收集、显示和重用与性能相关的信息。这些工具使用多种采样技术生成在性能问题诊断中非常有用的间隔性能监控数据。它们设计得十分高效，您可以连续运行它们而只有极小的影响。

体系结构

UNIX 操作系统的体系结构由三个层次组成：**用户**、**内核**和**硬件**，如下图所示：



内核级别是 UNIX 的核心，并用作**用户**和**硬件**级别之间的接口。**内核**级别由一组各种用途的程序组成。它们包括：

- **系统调用接口。**处理和执行系统调用，后者是程序用来请求操作系统的函数。
- **文件系统。**与进程控制和系统调用接口协调，处理字符和块数据的输入和输出。设备驱动程序负责数据 I/O。
- **进程控制。**协调和控制 UNIX 中的各种进程。进程就是操作系统上当前正在执行的程序。该程序是用户程序或系统程序。
- **硬件控制。**与硬件设备 （如键盘、监控器、硬盘和 RAM）协调。
- **设备驱动程序。**与系统设备（如硬盘、RAM 和打印机）通信，进行 I/O 操作。

内存管理器是 UNIX 体系结构的完整组成部分。它管理分配给 UNIX 上运行的不同进程的内存量。它负责管理内存层次结构。内存层次结构由以下部分组成：

- **缓冲区，或称缓存内存。**快速、昂贵的易失性内存，容量为数千字节 (KB) 或兆字节 (MB)。
- **主要内存，或称随机访问内存 (RAM)。**中速、中等价格的易失性主内存，容量为数兆字节和数吉字节。
- **辅助存储器，或称磁盘存储。**慢速、廉价的非易失性磁盘和磁带存储，容量以吉字节为单位。

用户程序和**系统程序**统称为**进程**。其主要目标是执行任务。系统给每个进程分配名为**进程标识 (PID)** 的唯一数字，并使用这些数字标识和管理进程。系统用这些数字将优先级分配到每个进程。创建进程之后，它们可在前台或后台运行。在后台运行进程可让系统同时处理多个进程。

性能资源

在 UNIX 中有 7 类主要资源需要监控和调整：

- CPU
- 内存
- 磁盘空间和磁盘臂
- 通信线
- I/O 时间
- 网络时间
- 应用程序。

总执行时间

从用户角度看，总执行时间由壁钟时间组成。在进程级别，这是通过运行 **time** 命令度量的。它向您提供实时（壁钟）用户代码 CPU 和系统代码 CPU。如果**用户 + 系统 > 80%**，则很可能系统的 CPU 受限。

总执行时间的组成部分包括：

- **用户状态 CPU。**CPU 花在运行用户状态中的程序的实际时间量。它包括花费在执行库调用的时间，但不包括自身花在内核中的时间。编译时使用优化、编写高效代码，都可明显影响该值。

- **系统状态 CPU。**CPU 代表该程序花在系统状态中的总时间。所有 I/O 例程都需要内核服务。程序员可通过阻止 I/O 传输影响此值。
- **I/O 时间。**服务于 I/O 请求花费的时间量
- **网络时间。**花费在移动数据上的时间。
- **虚拟内存性能。**包括上下文切换和交换。
- **运行其他程序花费的时间。**因为其他程序当前占用了 CPU，系统未能服务于此应用程序时。

工具

多数 UNIX 风格的操作系统都包含内置统计信息，后者由操作系统在进程执行过程中收集。这些统计信息的各个方面都可以用以下 UNIX 功能访问：

- **rstat。**服务器 / 守护程序，返回从内核获取的性能统计信息
- **netstat。**网络统计信息
- **nfsstat。** NFS 统计信息
- **time/timex。**进程 CPU 利用率
- **uptime。**系统平均负载
- **ps。**进程统计信息
- **iostat。** I/O 工具
- **sar。**大容量系统活动
- **vmstat。**虚拟内存工具
- **prof。**进程分析
- **trace。**用于获得更大深度

最有用的命令之一是 **uptime**，它提供系统平均负载，尽管只有当它不考虑优先级计划时才可用作大致指示器。运行 **uptime** 时，它提供三个平均负载：第一个是过去 1 分钟，第二个是过去 5 分钟，第三个是过去 15 分钟。

与 **-q** 选项一起使用时，**sar** 命令能很好地代替 **uptime**。它提供以下统计信息：运行队列的平均长度、占用运行队列的时间百分比、交换队列的平均长度和占用交换队列的时间百分比。运行队列列出在内存中并可运行的作业，但不包括正在等待 I/O 或正在休眠的作业。运行队列大小应小于 **2**。

注：各种 UNIX 风格操作系统都可能包含特定的功能，能简化性能监控。例如，Sun Solaris 使用 **rup** 和 **perfmeter** 命令增强，后者被广泛使用而取代了基础的 BSD 工具。

计数器类型

每个计数器都有计数器类型。知道计数器类型很有用，因为它指示性能统计信息是如何得出的。下面是一些最重要的计数器类别：

- **瞬时计数器。**显示最近度量的简单数字值。
- **间隔计数器。**显示活动率随时间的变化。
- **已用时间计数器。**按间隔收集，不能聚合。
- **平均值计数器。**提供该间隔得出的平均值。

用 HP 工具进行的 UNIX 监控

与 Windows 不同，UNIX 中的性能信息分散在收集各种统计信息的不同进程中。其中某些进程 （守护程序）持续运行，另一些必须调用才能获取数据。

UNIX 环境的 HP LoadRunner 和 HP Performance Center 的内置监控解决方案使用 **rstatd** 守护程序，后者通常已经在大部分版本上配置并运行。要验证是否已经配置 **rstatd** 守护程序，请执行 **rup** 命令，它报告各种计算机统计信息，包括 **rstatd**。使用此守护程序收集的统计信息，可从 UNIX 主机获取最流行的计数器，如 CPU 利用率、上下文切换率、磁盘速率等。如果需要获取性能度量的详细视图，建议使用前面讨论过的 UNIX 工具。

与发出其参数在不同操作系统之间有差异的特定命令相比，部署与 LoadRunner 和 / 或 Performance Center 一起使用的 HP SiteScope 显得更有道理。

HP SiteScope 通过隔离每个变体的细节并按照其用途对计数器分组，提供监控各种 UNIX 风格的操作系统的适应性基础结构。这是通过配置适配器文件，以便在需要监控时支持 UNIX 的特定版本实现的。 SiteScope 使用适配器文件描述从运行 UNIX 操作系统的不同版本的服务器检索多种系统资源信息需要的命令。

这些命令本质上是通用的，但在某些 UNIX 变体的基础功能上有扩展。这些命令覆盖 UNIX 的方方面面，有：

- **disk**。以磁盘为参数，返回该磁盘的总计、可用和已用百分比。
- **disks**。返回系统上的文件系统的列表。
- **memory**。已用和可用的交换空间量。
- **pageFault**。每秒页面故障数。如果发生多个页面错误行，则加起来。
- **cpu**。返回 CPU 的**等待**和**空闲**百分比。
- **process**。具有长进程名称的进程列表。

SiteScope 还会按用途（CPU、内存、I/O）对计数器分组，并自动收集关于该组实例的性能数据。例如，它提供 CPU 利用率总计与每个已安装处理器的样本数据，显示每个已安装的网络接口卡的网络统计信息，并提供总网络吞吐量的总计。该途径简化了性能测试人员的工作负载，因为它逻辑上合并了 Windows 和 UNIX 系统；而通常测试人员需要在环境之间不断切换，有时甚至在一次性能测试中也是如此。

处理器 —— 最重要的计数器

每个应用程序都会在执行时利用处理器 (CPU) 资源。对处理器资源的请求分为**用户状态**和**系统状态**处理。

用户状态处理与 CPU 花费在运行用户状态中的用户程序的时间实际量相关。它包括花费在执行库调用的时间，但不包括自身花在**内核**中的时间。

系统状态处理表示 CPU 代表该程序花在系统状态中的总时间。所有 I/O 例程都需要**内核**服务。

通常很容易识别 CPU 瓶颈：总体 CPU 利用率（所有现有处理器的平均值）达到或接近 **100%**，并始终有进程要等待得到服务时。但是，并不总容易找出 CPU 瓶颈发生的*原因*。因此分析负载时，获取正常时期应用程序行为的基本知识用作基准非常重要。

下面的计数器与系统级别监控相关，其中考虑到一般处理器参数，而不管特定进程的行为如何。

计数器	描述
CPU 利用率	处理器花在执行某任务上的总时间百分比。
用户模式 CPU 利用率	处理器花在用户模式下执行代码的已用时间百分比。
系统模式 CPU 利用率	处理器花在系统模式下执行代码的已用时间百分比
平均负载	在上一分钟内同时处于**就绪**状态的进程平均数
中断率	采样间隔内处理器花在接收和服务硬件中断上的时间。
上下文切换率	计算机上的所有处理器从一个进程或线程切换到另一个的合计速率。

CPU 利用率百分比

正式名称	CPU 利用率计数器
计数器类型	间隔 （繁忙百分比）
描述	该间隔内的总体处理器平均利用率。处理器*未*运行**空闲线程**的每个间隔，假设处理器在忙于处理某种真实工作负载。此计数器是**空闲 + 用户 + 系统利用率**（名称在不同平台上有所变化）的总和。 由于在大多数平台 （参见下面的相关度量）上都有专门的空闲 CPU 计数器，为了了解总体 CPU 占用情况，可使用以下公式： CPU 占用率 = 100 - 空闲 CPU (%)
使用备注	总体处理器使用情况的主要指示器。值落在 0-100% 繁忙程度的范围内。
性能	确定处理器是否是潜在瓶颈的主要指示器。
操作	持续接近 100% 利用率的时段可能意味着有失控进程。通常与重要运行队列（超过 3 个）或由于优先级被阻塞（超过 3 个）的进程相组合。 通过查看用户模式 CPU 利用率计数器以了解它是由用户进程还是由**内核**活动占用，做进一步调查。
阈值	对于面向响应的工作负载，要注意利用率持续高于 80-90% 的时段。
相关度量	➤ CPU 利用率 \ 空闲百分比 ➤ CPU 利用率 \ 用户百分比 ➤ CPU 利用率 \ 系统百分比 (Solaris) ➤ 处理器 \ 空闲 ➤ 处理器 \ 内核 (Linux) ➤ 处理器 \ 空闲百分比 ➤ 处理器 \ 用户百分比 ➤ 处理器 \ 系统百分比 (AIX)

注：计算机上处理器的利用率高并不总表示有需要解决的问题。但是，如果 CPU 空闲时间低于 20%，就需要调查，低于 10% 就可能表示有错误。

用户模式 CPU 利用率

正式名称	用户模式 CPU 利用率
计数器类型	间隔 （繁忙百分比）
描述	该间隔中在用户模式发生的总体处理器平均利用率，即 CPU 忙于处理应用程序请求。
使用备注	如果操作系统花费多数时间在内核外执行，则这通常是好事。但是，其处理能力应花在*正确的*进程上，且不应有重要的应用程序在等待得到服务。
性能	N/A
操作	如果进程仅在用户模式中运行，且没有系统调用和 I/O，则它可能卡在无限循环中了。具有密集型 I/O 操作的用户模式进程通常执行内存映射。 如果某些应用程序显示为占用了全部 CPU 时间，使测试中的应用程序无法使用，则测试中的应用程序会显示为由于优先级被阻塞。
阈值	总是超过 50% 的数字表示有瓶颈。
相关度量	在 CPU 利用率中列出

系统模式 CPU 利用率

正式名称	系统模式 CPU 利用率
计数器类型	间隔 （繁忙百分比）。
描述	该间隔中系统 （**内核**）模式下发生的总体处理器平均利用率。所有操作系统函数都在**内核**模式中运行。系统模式包括在启动设备 I/O 操作时涉及的设备驱动程序代码，以及用于完成中断处理的已延迟的过程调用。
使用备注	多数情况下，系统模式 CPU 利用率过高是由其他原因导致的。 CPU 在系统模式中花费的大多数时间是由于发生上下文切换 —— 实质上就是**内核**运行了太多作业。这种现象的另一个缘由可能是高中断率 （超过 30%），底层问题在于磁盘 I/O 或网络带宽。内存可能也有关系；如果它被完全占用，则交换开始拖慢系统。
性能	确定操作系统是否功能正常 （包括设备驱动程序是否正常）的辅助指示器，负责潜在的处理器瓶颈。
操作	如果没有上下文切换或高 I/O 可指责，则问题与系统调用有关；如果它超过 30%，则用操作系统工具深入发现根源。
阈值	总是超过 50% 的数字表示有瓶颈。
相关度量	在 CPU 利用率中列出

平均负载

正式名称	平均负载或运行队列
计数器类型	瞬时 （每个度量时段采样一次）。
描述	观察到的处理器 “就绪队列”中延迟并等待计划执行的进程数。处理器 “就绪队列”中等待的线程按优先级排序，处理器空闲时，就计划让优先级最高的线程运行。CPU 单元数对 “运行队列”没有影响。
使用备注	许多程序线程在自愿等待状态中休眠。活动线程子集设定了可观察到的处理器队列长度的实际上限。
性能	确定处理器是否是潜在瓶颈的重要辅助指示器。
操作	容量约束可能正导致过多应用程序延迟的指示。
阈值	在非常繁忙的单处理器计算机上，**平均负载 > 2** 是一个警告，说明工作量经常超过处理器能稳定处理的范围。在多处理器上，要用运行队列长度除以物理处理器数。
相关度量	➤ CPU 利用率 ➤ 队列长度 \runq-sz (Solaris) ➤ 队列统计信息 \runq-sz (AIX)

中断率

正式名称	中断率
计数器类型	间隔 （繁忙百分比）
描述	该间隔的中断模式下发生的总体处理器平均利用率。只有中断服务例程 (ISR) 作为设备驱动程序的函数在中断模式中运行。
使用备注	ISR 中断处理是最高优先级的处理。中断处理是系统函数，没有关联的进程。中断率过高可能表示设备故障，但无法查明是哪个设备。
性能	表示处理器花费在接收和服务硬件中断上的时间百分比。此值是生成中断的设备活动的间接指示，如网络适配器。此计数器显著增加表示有潜在硬件问题。
操作	确定故障设备是否正引起潜在处理器瓶颈的辅助指示器。
阈值	此计数器超过 30% 时需要引起注意。
相关度量	系统模式 CPU 利用率

上下文切换率

正式名称	上下文切换率
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	一个运行中的线程由另一个代替时，就发生上下文切换。因为 UNIX 支持多线程的操作，所以上下文切换对于系统是正常行为。用户模式线程调用任何有特权的操作系统函数时，用户模式线程与在系统模式中执行被调用函数的相应内核模式线程之间发生上下文切换。
使用备注	上下文切换是正常的系统功能，上下文切换率是工作负载的副产品。上下文切换率高通常并不表示有问题。它也并不意味着 CPU 容量不足。此外，对于上下文切换发生率，系统管理员通常可以做的事非常少，除非有某些特定的系统配置参数可用于调整，诸如增加每个进程默认可以占用 CPU 的时间量。 相对于历史正常值，每秒上下文切换率大幅增加可能反映出有问题，例如设备故障。
性能	上下文切换率很高通常表示应用程序设计有问题，也可能预示着可扩展性方面的难题。
操作	上下文切换发生在以下情况中：高优先级的线程抢占了正在运行的低优先级线程，或者高优先级线程阻塞。多数情况中，这是由于频繁创建和完成的进程导致的，例如，使用 shell 命令登录。这表示系统上参与处理器竞争的线程太多了。如果未看到很高的处理器利用率，上下文切换水平也很低，则可能表示线程阻塞了。

阈值	N/A
相关度量	N/A

注：度量性能之前，服务器和服务器应用程序必须打开、在运行并可用。

进程监控

UNIX 是功能强大而非常灵活的操作系统。它允许用户根据需要在前台或后台运行进程。在前台运行的程序有完整的读写访问权限，而那些在后台运行的则没有任何读访问权限。

性能计数器可用于度量特定线程和其他可执行的工作单元占用了多少 CPU 处理时间。这些处理器利用率度量使您能确定哪些应用程序应对 CPU 占用率高负责。

尽管没有在所有 UNIX 风格的操作系统上都可用的通用功能，但可使用 HP SiteScope 的“进程”对象提供每个所选进程 / 线程的统计信息，有以下数据可用（并非所有计数器在所有变体上都可用）：

- **CPU。**每个选择的进程的 CPU 利用率，以占总体 CPU 使用量的百分比表示。
- **MEMSIZE。**所选进程占用的内存量。
- **PID。**注册到操作系统的进程 ID。
- **THREADS。**所选进程的分叉线程数。
- **USER。**用户会话数。

如果 HP SiteScope 不能提供进程监控的满意详细信息，总还可以发出内置 UNIX 命令：

- **ps。**显示当前运行的进程的静态列表。此外，**ps** 命令还显示进程的特定详细信息，如 PID、已用内存和用于运行进程的命令行。多数情况下，建议添加 **-aux** 属性，因为它提供用户和非终端进程的数据
- **top。**显示所有当前运行的进程及其占用的内存量的列表。top 命令每隔几秒就自动更新该列表，以显示计算机上的活动进程。
- **proc 工具。**能用于获取有关进程的更详细信息。这些工具应谨慎使用，因为它们执行时会暂停进程的执行。 Proc 工具位于 **/var/proc** 中，包含 **pfiles**（活动进程）、**pflags**（状态信息和进程标记）、**pldd**（附加到每个进程的所有动态库文件）、**pmap**（进程的地址空间映射）、**psig**（为各种信号和线程处理程序执行的操作）、**prun**（运行或开始进程）、**pstack**（堆栈跟踪）、**pstop**（暂停执行特定进程）。

内存 —— 最重要的计数器

UNIX 维护物理 （驻留）和虚拟内存。操作系统会屏蔽应用程序现有的实际内存量，因此有夸大可用性的趋势。 UNIX 使用的术语**虚拟内存**本质上包括程序为其所有数据分配的内存量，包含共享内存、堆空间、程序文本、共享库、内存映射文件。分配给系统上所有进程的虚拟内存总量可大致转换为要保留的*交换*空间量（程序文本除外）。虚拟内存实际上与分配有多少实际物理内存关系不大，因为并非映射到虚拟内存中的所有数据都会在物理内存中活动 （驻留）。程序出现 “内存不足”错误时，通常意味着它已用完可保留的交换空间 （虚拟内存），而非用完物理 （驻留）内存。

RAM 不足常是磁盘性能问题的间接证据，此时过多的磁盘分页占用了太多可用磁盘带宽。因此，磁盘分页率是重要的内存性能指示器。

经常有人说，现在内存很便宜，因此购买更多的内存就可以解决所有问题。但是，拥有大量的物理内存并不能避免虚拟内存的不足，并且如果应用程序未在使用之后释放分配的内存而导致内存泄漏，可能产生致命的崩溃。在某些情况下，如果将底层 UNIX 系统设置为主管数据库或类似的大量事务处理应用程序，则添加很多内存可能会通过允许使用更大的内存中的缓存而大幅改善数据库性能。

观察到可用 RAM 不足时，通常一个很重要的步骤是确定分配的物理内存正如何使用，并计算有问题进程的驻留页面 （称为其驻留内存集）数。

除下面的常见计数器以外，跟踪**缓存**和**缓冲**内存的使用也很重要，随着前者成为后者的一部分而导致可用内存量下降并不一定表示有内存泄漏（在 Solaris 和 **HP/UX** 上参见 **%rcache/%wcache** 和 **bread/s bwrit/s**，在 Linux 上参见 **Cached** 和 **Buffers**）。

提示：建议您在以下情况下开始注意内存使用情况：

- 有一段时期内系统中的总交换空间使用量持续上升
- 内存使用可按以下公式计算：

```
已用内存 = 所有内存 -（缓存 + 缓冲 + 交换）
```

- 特定进程导致可保留交换空间持续上升 —— 多数情况下，这明确表明该进程导致内存泄漏。

计数器	描述
已用百分比	表示对计算机上运行的进程可用的总物理内存使用量。
可用兆字节数	表示对运行中的进程可用的总内存量。
分页率	表示每秒从磁盘读取页面或将页面写入磁盘以解决硬页面故障的发生率。
调入页率	表示每秒读取到物理内存的页数
调出页率	表示每秒写入页文件和从物理内存中删除的页数

已用百分比

正式名称	已用百分比
计数器类型	瞬时 （每个度量时段采样一次）
描述	RAM 中可寻址而不导致发生页面故障的已分配页面数量，以相对于所有已安装内存的百分点表示。
使用备注	内存使用情况的主要指示器。
性能	N/A
操作	N/A
阈值	已安装 RAM 的值持续超过 80% 表示内存不足。它达到 90% 时就要当心了，因为此时运行进程可能会失败。
相关度量	内存\可用内存 —— 以字节为单位 (Solaris)

可用兆字节数

正式名称	可用兆字节数
计数器类型	瞬时 （每个度量时段采样一次）
描述	可用虚拟内存的兆字节总数。
使用备注	显示有多少内存可用于运行进程。
性能	N/A
操作	N/A
阈值	N/A
相关度量	内存 \ 交换空间可用内存 —— 以字节为单位 (Solaris)

分页率

正式名称	分页率或页数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）
描述	该间隔内磁盘的分页操作数。页数 / 秒是调入页数 / 秒和调出页数 / 秒的总和。
使用备注	程序接触到页面上不在物理内存中的虚拟地址时，结果就是 “调入页面”。 UNIX 需要在物理内存中腾出空间或发布内存映射文件时，结果就是 “调出页面” 。调出页面期间，全部驻留内存集都转移到磁盘交换区域。在调出页面时，进程挪出运行队列，这样它不占用 CPU。
性能	确定物理内存是否是潜在瓶颈的主要指示器。通常没必要密切监控页面调入，而需密切监控页面调出，因为后者经常表示内存瓶颈。 高分页率的另一个原因可能是文件系统缓存缓冲区过大。
操作	过多的分页可导致响应时间缓慢和不稳定。
阈值	分页率超过每个交换设备 50 时就要小心了。
相关度量	➤ 调入页率 ➤ 调出页率

注：

- 过高的分页率通常可通过增加 RAM 来降低。磁盘带宽有限。用于分页操作的容量对其他面向应用程序的文件操作不可用。
- 计算交换空间大小时，建议至少有任何应用程序可能请求的“可保留”交换空间大小。

调入页率

正式名称	调入页率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	此计数器表示进程需要访问的部分内存在虚拟内存中，需要读入物理内存中才能执行。它显示读取操作数，而不考虑每次操作中检索的页面数。较高的值表示有内存瓶颈。
使用备注	此计数器和相应的页面调出计数器相比不太重要。除非它意外升高，否则无需随时关注。
性能	确定物理内存是否是潜在瓶颈的次要指示器。
操作	过多的分页可导致响应时间缓慢和不稳定。
阈值	N/A
相关度量	分页率

调出页率

正式名称	调出页率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	此计数器表示该进程的驻留内存集相对于物理内存太大，正将它分页到磁盘。它显示读取操作数，而不考虑每次操作中检索的页面数。较高的值表示有内存瓶颈。
使用备注	如果很低的页面调出操作率与很高的物理磁盘活动值同时出现，则可能有磁盘瓶颈。如果队列长度增加并未伴随页面调出率下降，则存在内存不足。
性能	确定物理内存是否是潜在瓶颈的主要指示器。
操作	过多的分页可导致响应时间缓慢和不稳定。
阈值	分页率超过每个交换设备 50 时就要小心了。
相关度量	分页率

I/O —— 最重要的计数器

UNIX 通过 I/O 管理器堆栈维护物理和逻辑磁盘操作。**逻辑卷**代表有唯一驱动器盘符的一个文件系统。**物理（原始）卷**是特定存储设备的内部表示，无论它是 SCSI、 RAID、 SATA 还是其他技术。

使用阵列 Controller 或 RAID 这样的复杂存储系统时，底层物理磁盘硬件的特性并不直接对操作系统可见。这些特性（即磁盘数、磁盘速度、查找时间、转速和位密度，以及某些优化功能如板载内存缓冲区）都对性能有重大影响。内存缓冲区和命令排队之类的高级功能可以将性能提升 25-50%。

主动采取措施提升磁盘性能很重要，因为它下降趋势很快，特别是发生磁盘分页活动时。

注：

- 通常，使用许多较小的磁盘胜过使用少量大磁盘，因为前者更有灵活性并缓解了磁盘瓶颈。尝试将使用率过高的逻辑卷拆分到若干不同磁盘和 I/O 通道。
- 为应用程序确定目录路径时，尽可能减少从文件系统根目录开始的层数。过深的目录树由于需要更多查询才能访问文件，会影响性能。相反，如果给定目录中有太多文件（几千个）时，文件访问也可以变慢。

有大量 I/O 活动的面向事务的应用程序在使用原始设备而不是文件系统时会性能更好。多数数据库供应商（如 Oracle）通常都会这样建议。但是，对逻辑卷管理的最新改进使文件系统设备达到了原始卷的水平。任何情况下，将独立应用程序分配到唯一的物理磁盘以减少彼此影响都是好的想法。

计数器	描述
已使用百分比	表示每个已装入文件系统上已用空间的相对量。
可用	表示每个已装入文件系统上的可用字节数。
磁盘速率	表示物理磁盘是否是潜在瓶颈。

已使用百分比

正式名称	Filesystsems(n)\ 已使用百分比
计数器类型	间隔 (%)
描述	当前文件系统磁盘利用率，以全部容量的百分点表示。
使用备注	物理磁盘 I/O 性能的主要指示器。性能依赖于底层磁盘配置，此配置对操作系统是透明的。单个磁盘的性能特征根据查找时间、转速、记录密度和接口速度而各有不同。较为昂贵、面向性能的磁盘可提供 50% 的更佳性能。
性能	确定磁盘是否是潜在瓶颈的主要指示器。
操作	磁盘响应时间过久会延长应用程序响应时间。
阈值	如果此度量达到 90%，则为警告的指示，超过 95% 则表示错误。
相关度量	➤ Filesystsems(n)\ 已使用百分比 (Linux) ➤ Filesystsems(n)\ 已使用 —— 以字节为单位 (Solaris)

可用

正式名称	Filesystsems(n)\ 可用
计数器类型	瞬时 （每个度量时段采样一次）
描述	逻辑磁盘上的未分配空间量，以字节数报告。因为计算非常大型文件系统的可用兆字节数很耗时， I/O 管理度量层大约每 5 分钟重新计算一次该计数器的值。 用于计划磁盘使用的主要度量。如果没有磁盘容量计数器可用 （某些 UNIX 风格的操作系统确实提供此计数器），则使用该度量并知道总磁盘容量，即可计算利用率。
使用备注	已用的逻辑盘空间容量的主要指示器。
性能	N/A
操作	用尽文件系统上的空间通常会导致灾难性后果。
阈值	不可用
相关度量	➤ Filesystsems(n)\ 可用 ➤ Filesystsems(n)\ 已使用 (Linux) ➤ Filesystsems(n)\ 可用 ➤ Filesystsems(n)\ 已使用 ➤ Filesystsems(n)\ 容量 (Solaris)

磁盘速率

正式名称	Filesystsems(n)\ 磁盘速率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	该间隔内完成物理磁盘请求的速率。
使用备注	物理磁盘 I/O 活动的主要指示器。也称为磁盘到达率。
性能	确定磁盘是否是潜在瓶颈的主要指示器。
操作	磁盘响应时间过久会延长应用程序响应时间。
阈值	取决于底层磁盘硬件。
相关度量	N/A

提示：改进 I/O 吞吐量的常规提示有：

- 尽可能分散磁盘 I/O —— 10 个磁盘 10% 忙碌胜于一个磁盘 100% 忙碌。
- 避免过多的记录 —— 某些应用程序可以控制日志详细程度级别。
- 调整 SCSI 设备 —— 有时可以调整特定设备的最大队列长度。这通常会提高并行处理率，可能的代价是硬件负担过重。

注：关于磁盘的某些事实：

- I/O 越小，服务时间就越短。 I/O 越长，服务时间就越长。
- 顺序 I/O 比随机 I/O 更快，这是因为减少了磁头的移动。
- 更大的 I/O 可容纳最大的顺序 I/O 吞吐量。
- 跨越各种系统边界，如文件系统块、缓冲区链或文件范围，可能导致将一个 I/O 请求分解为较小的若干个。
- 如果最忙的磁盘是交换设备，则很可能有表现为磁盘问题的内存瓶颈，而您需要首先解决内存问题。

网络 —— 最重要的计数器

随着分布式和云应用程序的增加，网络性能现在变得越来越重要。但是，UNIX 操作系统通常提供各层的有限统计信息：在最低的硬件接口层和较高的网络协议层（如 TCP/IP）。网络接口统计信息由嵌入网络接口驱动程序层中的软件收集。此软件计算发送和接收的数据包数。

网络统计信息是通过 **netstat**、**netperf**、**iozone** 和 **nfsstat**（用于 NFS 监控）这样的 UNIX 功能收集的；安装的每个网络接口芯片或接口卡各一个。网络节点管理器和 SiteScope 这样的 HP 产品可以收集随时间变化的统计信息，使您能洞察性能瓶颈的真实原因。

捕获并分网络瓶颈是很复杂的。数据包率、冲突率和错误率并不总是指向问题的原因：

- 仅冲突率过高可能表示网络瓶颈。如果其水平随时间相对降低，则通常是正常行为。本质上是错误的冲突是双工或速度设置中不匹配的结果。纠正后，冲突率下降，同时性能改善。
- 数据包率的突然提高与很高的网络输出队列一起出现时也可能表示网络瓶颈。但是，为做出明智的决定，有必要观察随时间变化的模式行为。
- 如果频繁使用 NFS，则有必要监控由 **nfsstat** 收集的数据，尤其是在服务器端。如果 NFS 统计信息显示由一个特定客户端导致的很多活动，则建议在该客户端主机上运行该工具以识别进程。
- 一个处理器上系统模式 CPU 利用率或中断率很高，而其他处理器多数空闲时，可能有网络瓶颈。检查设备配置和硬件可能可找到原因。

计数器	描述
传入数据包率	表示每秒传入 NIC 的以太网数据包数。
传出数据包率	表示每秒 NIC 发送的以太网数据包数。
传入数据包错误率	表示每秒传入 NIC 的以太网数据包错误数。
传出数据包错误率	表示每秒 NIC 发送的以太网数据包错误数。
冲突率	表示网络冲突数。

传入数据包率

正式名称	传入数据包率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	该间隔期间此接口每秒接收的总计字节。
使用备注	网络接口流量与传出数据包率的主指示器。
性能	确定网络是否是潜在瓶颈的主要指示器。
阈值	传入数据包率超过线容量的 40% 时发出警告。
相关度量	传出数据包率

传出数据包率

正式名称	传出数据包率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	该间隔期间此接口每秒发出的总计字节。
使用备注	网络接口流量与传入数据包率的主指示器。
性能	确定网络是否是潜在瓶颈的主要指示器。
阈值	传入数据包率超过线容量的 40% 时发出警告。
相关度量	传入数据包率

注：以上这两个计数器显示流经此接口的吞吐量 （字节数），帮助识别特定网络适配器的流量是否饱和，有没有必要增加另一个网络适配器。

传入数据包错误率

正式名称	传入数据包错误率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	该间隔期间此接口每秒接收的错误数。
使用备注	网络接口流量与传出数据包率的重要辅助指示器之一。
性能	确定网络是否是潜在瓶颈 （通常是双工和速度配置不匹配的结果）的次要指示器。
阈值	传入数据包错误率超过每秒 0.025 个错误时发出警告。
相关度量	传出数据包错误率

传出数据包错误率

正式名称	传出数据包错误率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	该间隔期间此接口每秒发出的错误数。
使用备注	网络接口流量与传入数据包率的重要辅助指示器之一。
性能	确定网络是否是潜在瓶颈 （通常是双工和速度配置不匹配的结果）的次要指示器。
阈值	传出数据包错误率超过每秒 0.025 个错误时发出警告。
相关度量	传入数据包错误率

提示：以上两个计数器跟踪网络质量。如果比率超过指定阈值，则可能需要检查网络硬件设备了。

冲突率

正式名称	冲突率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	每秒接口上发生的错误数。
使用备注	此计数器表示通过网络发送或接收数据时发生的错误数。较高值表示网络带宽是瓶颈。通常由硬件压缩问题或物理连接器 / 终端奇故障所导致。
性能	确定网络是否是潜在瓶颈的主要指示器。如果值高于阈值，可能需要重新评估网络拓扑了，因为该段上网络超载。
阈值	值不应超过 10%。
相关度量	N/A

第 III 部分

运行时平台

5

运行时平台监控

此章节提供有关运行时平台监控的概述，并描述所需的 J2EE 和 .NET 应用程序体系结构。

此章节包括：

概述

通常应用程序开发为在特定操作系统上运行，其性能取决于控制操作系统的因素。每个操作系统都有自己的一组性能参数，通过监控和调整它们获得更好的性能。

应用程序的性能还取决于体系结构级别的监控和调整。但是，体系结构的设计基于特定的技术。因此技术级别的监控和调整也必须处理好，才能获得更好的结果。为取得所有这些效果，必须在各阶段强制实现用于监控和调整的正确准则。

尽管如今有许多技术（通用的和专用的），但创建企业应用程序都用 Java 2 Enterprise Edition (J2EE) 或其 Microsoft 的 .NET Framework。开发者现在可以在前所未有的更短时间内构建功能更多和更健壮的业务解决方案。

设计这些解决方案未必简单，有了更多特性和功能，开发者开发出糟糕的解决方案的机会也增加了。应用程序可能在开发和 QA 环境中表现很好，但在生产中无法扩展或出现性能问题。

了解运行应用程序的基础结构的影响和许多应用程序组件在负载下交互时的行为很重要。

由于快速上市的压力增加，许多面向 Web 的 J2EE 和 .NET 应用程序的部署生命周期被压缩了。开发、 QA、部署、生产阶段和 IT 组之间的界限模糊了。集中的 IT 组织可能要管理几百个应用程序，每个的管理深度都很浅。 IT 员工的 J2EE 技能可能发展得不够。

很多应用程序的设计尚未顾及性能和可扩展性、未全面考虑到设计和使用模式，且未充分注意到根据定义明确的服务目标进行计划和测试性能。J2EE 的可扩展性功能尽管广泛，也不能代替这样的努力。.NET 配置设置也一样，缓冲、会话超时、应用程序保护级别和记录配置都可影响 .NET 应用程序在负载下的性能。

体系结构

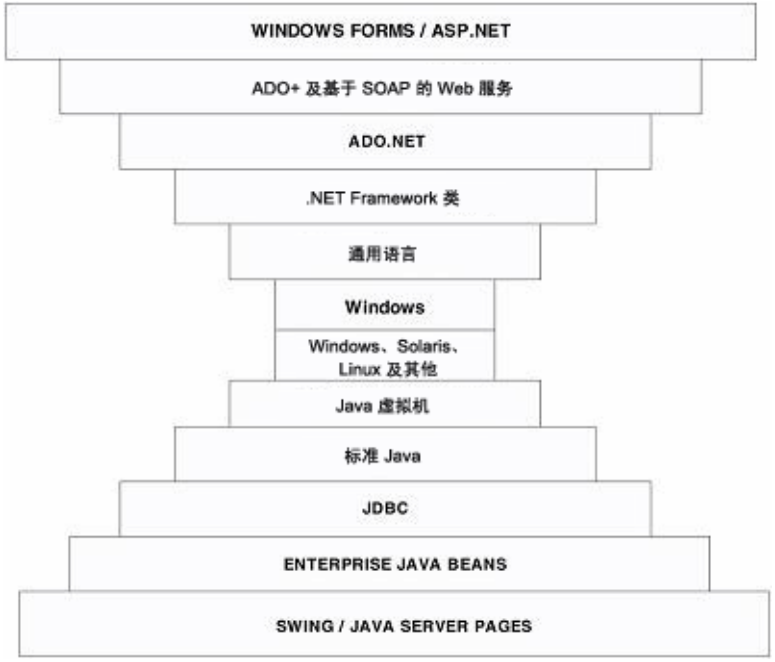
J2EE 或 .NET 应用程序运行时，运行它们的操作系统提供可为达到最佳性能设置成特定值的各种参数。此类参数由各种计数器监控和度量。对所有测试分析师来说，知道有助于从性能角度调整操作系统的计数器都是最重要的。

后面几章讨论与 Windows 和 UNIX 系统相关的重要计数器，因为大多数应用程序都在这两种操作系统上运行。

在 UNIX 中，需要监控和调整的主要资源类型是 CPU、内存、磁盘空间、通信行、I/O 时间、网络时间和应用程序。UNIX 操作系统维护着几个跟踪系统资源及其利用率的计数器。下面列出这些计数器中的一部分：CPU 利用率、缓冲区使用情况、磁盘 I/O 活动、磁带 I/O 活动、终端活动、系统调用活动、上下文切换活动、文件访问利用率、队列活动、进程间通信 (IPC)、分页活动、可用内存和交换空间、内核内存分配 (KMA) 等等。有关详细信息，请参见第 4 章 “监控 Unix”。

Windows 是**自我调整**的操作系统。这意味着在大多数情况下，如果硬件配置正确，Windows 会根据运行它的环境自动适应，以实现性能优化。例如，Windows 部署为 Web 服务器时，同样存在但不用的其他服务被置于一种极少占用 CPU 和内存之类资源的状态中。但是，和许多其他操作系统一样，性能取决于很多外部因素，诸如硬件、设备驱动程序、应用程序、工作负载、网络等等。有关详细信息，请参见第 3 章 “Windows 监控”。

J2EE 和 .NET 都需要在应用程序开发之前定义应用程序体系结构。这些技术支持自己的一套用于定义体系结构的框架。但是，这些技术之间，在定义系统方面有些体系结构上的相似点。这些相似点有助于我们定义用于监控性能计数器和调整应用程序的通用准则。 J2EE 和 Microsoft 的 .NET 技术共享广泛的通用基础标准，它们都采用多层体系结构方式，后者通常在不同逻辑层中实现应用程序，这些层将呈现与内部结构 （业务逻辑和数据管理）分开：



- J2EE 和 .NET 体系结构模型都使用面向对象 (OO) 方式支持主流企业计算，用功能强大的 OO 框架 （类库）支持诸如企业组件管理、对象持久性、事务、 Web 服务、异步通信、松耦合事件服务、消息收发等等服务。
- 虚拟机 (VM) 体系结构的使用对 J2EE 和 .NET 是通用的。应用程序开发工具产生中间级代码，而不是特定于平台的二进制代码。这意味着 VM 实时解释代码或执行实时 (JIT) 编译。
- J2EE 和 .NET 共享实现多层方式的广泛通用基础。

在 QA 周期中，性能测试通常紧跟集成功能和回归测试。在发行软件前，应对整个应用程序进行性能测试，包括与外部系统的所有接口。

目标包括估计在真实代表预期实时使用情况的负载下的可扩展性和容量，以及深入探究应用程序的内部性能行为、收集可用于处理瓶颈问题的数据。这应当包括每个 J2EE/.NET 层和方法所导致延迟的事务细分，以及其他的特定根源诊断信息。

6

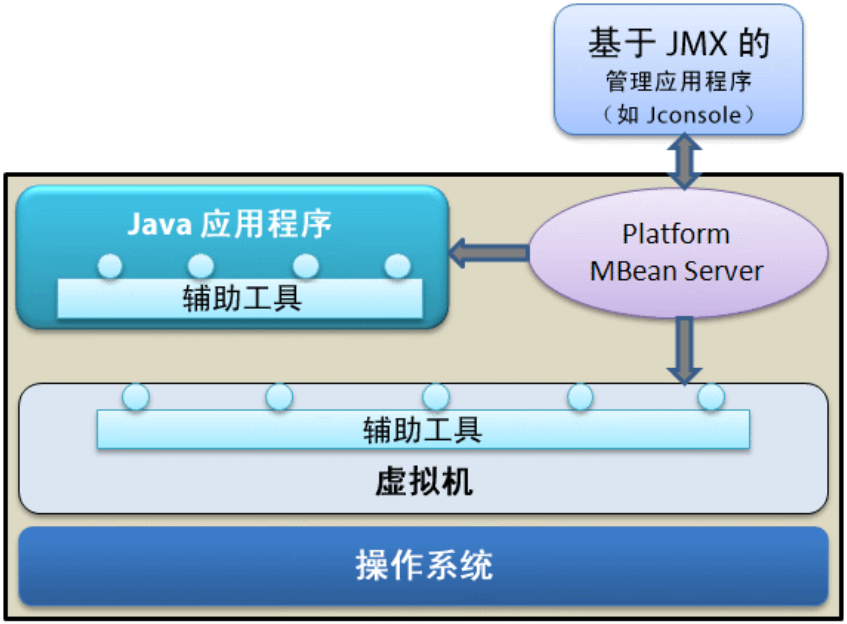
Java 平台监控

此章节描述 Java 平台监控的最佳实践。

此章节包括：

概述

Java 2 平台提供全面的监控和管理支持。它不仅定义 Java 虚拟机 (JVM) 的管理接口，还提供 Java 平台上以及在其上运行的应用程序的现成远程监控和管理功能。



此外， JDK 5.0 还包括 Java 监控和管理控制台工具 **(JConsole**)。 JDK 5.0 使用 JVM 的众多测量功能，用 Java 管理扩展 (JMX) 技术提供有关 Java 平台上运行的应用程序性能和资源消耗的信息。 JMX 提供测量 Java 运行时环境和应用程序的标准方式。该测量可通过 JMX 管理的 Bean (MBean) 界面访问，后者在平台 MBean 服务器中注册。应用程序还可以创建自己的 MBean，并在平台 MBean 服务器中注册它们，该服务器可充当远程访问的单一点。符合 JMX 的客户端 （如 JConsole）可连接到平台 MBean 服务器并用 JMX 技术管理应用程序（以及 Java 平台）。每个平台 MBean 都有一组丰富的属性和操作，例如内存使用、线程 CPU 使用、垃圾回收统计等等。

HP SiteScope 提供对 JMX 的内置支持，使 JConsole 不再必需，并提供操作系统计数器和特定于 Java 的应用程序度量的合并视图。所有可通过 JConsole 访问的计数器也可通过 HP SiteScope 访问。

最重要的 Java 计数器

	计数器	描述
常用	正常运行时间	表示 JVM 运行了多久
	总编译时间	表示在实时 (JIT) 编译中花费的时间
	进程 CPU 时间	表示 JVM 消耗的 CPU 总时间
内存	当前堆大小	表示堆当前占用的千字节数
	最大堆大小	表示堆占用的最大千字节数
	承诺的内存	表示分配给堆使用的内存总量
	GC 时间	表示花费在垃圾回收上的累计时间和调用总数
线程数	活动线程数	表示活动守护程序线程数加非守护程序线程数的当前数
	峰值线程数	表示 JVM 启动以来活动线程的最高数
	守护程序线程	表示当前的活动守护程序线程数
	启动的总线程数	表示自 JVM 启动以来启动的线程总数（包括守护程序、非守护程序和已终止的）
类	加载的当前类	表示当前加载到内存中的类数
	加载的总类数	自 JVM 启动以来加载到内存中的类的总数，包括那些随后卸载的
	卸载的总类数	JVM 启动以来从内存卸载的类数

常用计数器

此部分描述与计算机上运行的 JVM 相关的常用信息的计数器。

正常运行时间

正式名称	正常运行时间
计数器类型	已用时间
描述	在计算机上启动 JVM 以来的已用时间量
使用备注	显示 Java 的总体状态
性能	总体运行状况的重要指示器
操作	如果很少运行垃圾回收，则 JVM 运行越久，就有越多线程保持打开
阈值	N/A

总编译时间

正式名称	总编译时间
计数器类型	已用时间
描述	在实时 (JIT) 编译中花费的时间。JVM 实现决定了 JIT 编译何时发生。
使用备注	由于 JVM 将 Java 解释为字节码，它需要在加载时编译对象。此计数器显示自 JVM 开始运行之前，总共花费了多少时间。 Sun 的 Hotspot VM 使用适应性编译，在这种编译中， VM 使用标准解释程序启动应用程序，但随即在运行时分析代码以检测性能瓶颈或“热点”。
性能	辅助指示器，确定大量新对象是否造成潜在瓶颈
操作	此计数器可以查明系统是否正确部署和启动
阈值	N/A

进程 CPU 时间

正式名称	进程 CPU 时间
计数器类型	已用时间
描述	JVM 消耗的 CPU 总时间
使用备注	查看 JVM 如何影响总体操作系统表现的主要指示器之一
性能	可用于计算处理器在 Java 和所有其他进程上花费的时间的百分比。此计数器显著增加表示有潜在问题。
操作	此计数器可查明按比例扩大 JVM 需要的更改
阈值	取决于处理器

内存计数器

这部分描述 JConsole 的 “内存”选项卡上通常显示的计数器。它们显示有关内存使用、内存池和垃圾回收统计信息的数据。

内存池可用性取决于所用的 JVM。下表显示 Sun Java 的标准安装附带的 HotSpot 虚拟机池。

- **Eden Space （堆）池。**最初从该池为多数对象分配内存。
- **Survivor Space（堆）池。**包含 Eden Space 池的垃圾回收后仍存在的对象。
- **Tenured Generation （堆）池。**包含 Survivor Space 池中已存在一段时间的对象。

- **Permanent Generation （非堆）池。**保留了虚拟机本身的所有反射型数据，如类和方法对象。对于使用类数据共享的 JVM，该池划分成只读和读写区域。
- **Code Cache（非堆）池。**HotSpot JVM 还包括“代码缓存”，后者包含用于本机代码的编译和存储的内存。

每个内存池都可能有两类内存阈值，用于低内存检测支持：**使用阈值**和**回收使用阈值**。特定内存池可能不支持这两个阈值中的任意一个。

- **使用阈值**。内存池的可管理属性。它使您能以很低的开销监控内存使用。将阈值设置为正值时，允许对内存池进行使用情况阈值检查。将使用情况阈值设置为 **0** 会禁用使用情况阈值检查。默认值由 JVM 提供。JVM 在最合适的时间检查内存池的使用情况阈值，通常是在垃圾回收期间，有时会在分配时。如果 JVM 检测到当前内存使用超过使用情况阈值，则它会将 **UsageThresholdExceeded** 属性设置为 **true**。
- **回收使用情况阈值。**某些垃圾回收的内存池的可管理属性。 JVM 对内存池执行了垃圾回收后，池中的部分内存仍被可访问的对象占用。回收使用情况阈值允许您设置只在垃圾回收后检查内存使用情况的值。如果 JVM 检测到内存使用超过了回收使用情况阈值，它将把 **CollectionUsageThresholdExceeded** 属性设置为 **true**。

JVM 管理两类内存，它们都是在 JVM 启动时创建的：

- **堆内存。**运行时数据区， JVM 从中为所有类实例和数组分配内存。堆的大小可以固定，也可以可变。垃圾收集器是一种收回对象的堆内存的自动内存管理系统。
- **非堆内存。**包括在所有线程间共享的方法区和 JVM 的内部处理或优化需要的内存。它存储运行时常量池、字段和方法数据之类的按类划分的结构，以及方法和构造函数的代码。方法区是堆的逻辑组成部分，但根据实现的不同， JVM 可能不会回收垃圾或压缩它。与堆一样，方法区可以采用固定或可变的大小。方法区的内存不必相邻。

  除方法区以外， JVM 实现可能还需要内存用于内部处理或优化，后者也属于非堆内存。例如，JIT 编译器需要内存用于存储从 JVM 代码转换来的本机计算机代码，以实现高性能。

当前堆大小

正式名称	当前堆大小
计数器类型	瞬时 （每个度量时段采样一次）
描述	当前使用的内存量
使用备注	使用的内存包括由所有对象 （含可访问和无法访问的对象） 占用的内存
性能	是确定内存是否是潜在瓶颈的重要指示器
操作	如果此参数以后增加了，则可能表示需要重新配置
阈值	参见如上说明

最大堆大小

正式名称	最大堆大小
计数器类型	瞬时 （每个度量时段采样一次）
描述	可用于内存管理的内存最大量。其值可能更改或未定义
使用备注	如果 JVM 尝试将所用内存增加到大于承诺的内存，则即使使用的数量小于或等于最大值，内存分配也可能失败 （例如在系统上虚拟内存不足时）
性能	显示内存上限 —— 如果接近物理内存边界则发出警告
操作	如果没有隐式定义，则可能导致错误的内存分配
阈值	参见如上说明

承诺的内存

正式名称	承诺的内存
计数器类型	瞬时 （每个度量时段采样一次）
描述	分配给堆使用的内存总量
使用备注	JVM 确保可用的内存量。承诺的内存量以后可能更改。JVM 可能将内存释放到系统，且承诺的内存可小于启动时最初分配的内存量
性能	N/A
操作	N/A
阈值	承诺的量将始终大于或等于使用的量

GC 时间

正式名称	GC（垃圾回收）时间
计数器类型	已用时间
描述	花费在垃圾回收上的累计时间和调用总数。它可能有多行，每行表示 JVM 中使用的一个垃圾回收器算法。
使用备注	垃圾回收 (GC) 就是 JVM 如何释放由不再引用的对象占用的内存。 通常将有活动引用的对象视为**活动的**，将无引用（不可访问）的对象视为**不活动的**。垃圾回收就是释放不活动对象所用内存的过程。
性能	GC 使用的算法和参数都可能对性能产生显著影响。Sun 的 HotSpot VM 垃圾回收器使用**分代**垃圾回收。分代 GC 利用这样一个事实，即实际上多数程序创建的： ➤ 很多对象寿命都很短（例如迭代器和本地变量） ➤ 部分对象有很长的寿命（例如高级的持久对象） 如此，分代 GC 将内存划分成几代，每代都分配到一个内存池。某一代用完分配的内存时，VM 在该内存池上执行部分垃圾回收（也叫**小**回收）以收回由不活动对象使用的内存。这种部分 GC 通常比完整 GC 快得多。
操作	如果 GC 已成为瓶颈，则可能要自定义代的大小。检查详细的 GC 输出，然后探索单个性能度量对 GC 参数的敏感性。
阈值	承诺的将始终大于或等于使用的。

注：困扰内存配置低于理想值的用户的最大问题之一就是 GC 暂停。有很多设置影响 JVM 分配内存的方式和 GC 的行为。监控 GC（进而加以调整）的主要目的是减少重大 GC 事件的频率，而不增加其积累持续时间。

线程计数器

线程与程序中执行的线程相关。 JVM 允许应用程序同时运行多个执行线程。每个线程都有个优先级。高优先级的线程先于低优先级的线程执行。每个线程可以标记或不标记为守护程序。某个线程中运行的代码新建“线程”对象时，新线程的优先级最初设置为创建它的线程的优先级；当且仅当创建它的线程是守护程序时，它才是守护程序线程。

JVM 启动时，通常有单个的非守护程序线程 （通常调用某指定类的名为 **main** 的方法）。 JVM 继续执行线程，直到发生以下两种情况之一：

- 已调用类 Runtime 的 exit 方法，并且安全管理器已允许退出操作发生。
- 非守护程序线程的所有线程皆已停止，要么是通过从调用返回到 run 方法，要么是通过抛出传播范围超过 run 方法的异常

活动线程数

正式名称	活动线程数
计数器类型	瞬时 （每个度量时段采样一次）
描述	显示活动守护程序线程数加非守护程序线程数的当前数
使用备注	N/A
性能	线程过多可能导致垃圾回收操作缓慢
操作	N/A
阈值	观察计数器何时接近 “峰值”线程值

峰值线程数

正式名称	峰值线程数
计数器类型	累积
描述	JVM 启动以来活动线程的最高数
使用备注	可能有助于识别 JVM 行为的趋势模式
性能	N/A
操作	N/A
阈值	N/A

守护程序线程

正式名称	守护程序线程
计数器类型	瞬时 （每个度量时段采样一次）
描述	当前的活动守护程序线程数
使用备注	
性能	线程过多可能导致垃圾回收操作缓慢
操作	N/A
阈值	观察计数器何时接近 “峰值”线程值

启动的总线程数

正式名称	启动的总线程数
计数器类型	累积
描述	自 JVM 启动以来启动的线程总数（包括守护程序、非守护程序和已终止的）
使用备注	可能有助于识别 JVM 行为的趋势模式
性能	N/A
操作	N/A
阈值	N/A

注：通常仅监控一对线程计数器就足够了，如 “启动的总线程数”和 “活动线程数”，因为其他的线程计数可以从中推出。

提示： 要检查应用程序是否遇到死锁（例如应用程序似乎已挂起），可以从 JConsole 的 MBeans 选项卡调用 **findMonitorDeadlockedThreads** 操作。

类计数器

此部分描述最重要的类计数器。

加载的当前类

正式名称	加载的当前类
计数器类型	瞬时 （每个度量时段采样一次）
描述	当前加载到内存中的类数
使用备注	
性能	N/A
操作	N/A
阈值	N/A

加载的总类数

正式名称	加载的总类数
计数器类型	累积
描述	自 JVM 启动以来加载到内存中的类的总数，包括那些随后卸载的
使用备注	可能有助于识别 JVM 行为的趋势模式
性能	N/A
操作	N/A
阈值	N/A

卸载的总类数

正式名称	卸载的总类数
计数器类型	累积
描述	显示 JVM 启动以来从内存卸载的类数
使用备注	可能有助于识别 JVM 行为的趋势模式
性能	此计数器值长期非零，可能表示垃圾回收机制有问题
操作	N/A
阈值	N/A

7

.Net 平台监控

此章节描述 .Net 平台监控的最佳实践。

此章节包括：

概述

用 Microsoft 技术开发的大多数应用程序都在 .NET Framework 下。此框架为应用程序的开发和运行提供了良好平台。它还提供了度量和监控应用程序性能的计数器。

.NET Framework 有两个主要组件：

➤ 公共语言运行时

➤ .NET Framework 类库。

公共语言运行时 (CLR) 是 .NET Framework 的基础。可以将运行时视为在执行时管理代码的代理程序，提供诸如内存管理、线程管理和远程管理这样的核心服务，并且还强制实行严格的类型安全及其他代码准确性机制，从而提升安全性和健壮性。实际上，代码管理的概念是运行时的基本原则。以运行时为目标的代码称为受管代码，否则称为非受管代码。类库作为 .NET Framework 的另一主要组件，是全面的、面向对象的可重用类型集。

运行时是设计用于增强性能的。尽管公共语言运行时提供很多标准的运行时服务，但从不解释受管代码。名为实时 (JIT) 编译的功能允许所有受管代码以执行它的系统的本机机器语言运行。同时，内存管理器消除了内存碎片化的可能性，并提高了内存的“引用局部性”，进一步增强了性能。

性能计数器被组织并分组为不同的类别。总的来说，正如 Windows 操作系统提供很多预定义的性能计数器，可以通过编程检索或使用性能监控器显示一样，在 .Net 中 CLR 提供自己的性能计数器组。它们组织为 9 个重要类别，帮助测试者监控和调试应用程序的性能：

- **异常。**提供有关应用程序所抛出异常的信息。
- **内存。**提供有关垃圾回收器的信息。
- **锁定和线程。**提供有关应用程序所用的受管锁定和线程的信息。
- **协作性。**提供有关应用程序与 COM 组件、COM+ 服务和类型库交互的信息。
- **JIT。**提供有关已由实时编译器编译的代码的信息。
- **加载中**。提供有关已加载的组件、类和 AppDomain 的信息。
- **联网。**提供有关由应用程序通过网络发送和接收的数据的信息。
- **远程。**提供有关由应用程序使用的远程对象的信息。
- **安全性。**描述 CLR 对应用程序执行的安全检查。

最重要的 .Net 计数器

监控 .Net 应用程序时，建议从度量处理器、内存、网络和 I/O 设备的利用率的操作系统计数器开始监控（参见 Windows 一章）。然后可以添加涵盖从异常处理到安全检查的 CLR 操作每个方面的 .Net 性能计数器。

	计数器	描述
异常	每秒抛出的异常数	表示每秒抛出的受管代码异常数
	抛出至捕获深度 / 秒	表示堆栈帧数
内存	大对象堆大小	表示大对象堆的当前大小（以字节为单位）
	所有堆中的字节数	表示当前在 GC 堆上分配的内存字节数。
	固定对象数	表示在最后一个 GC 中遇到的固定对象数。
	GC 中的时间百分比	表示上个 GC 周期以来在执行垃圾回收 (GC) 中已花时间的百分比。
线程数	当前逻辑线程数	表示应用程序中当前 .NET 线程对象数。
	当前物理线程数	表示由 CLR 创建和拥有的本机操作系统线程数。
	当前识别的线程数	表示当前 CLR 识别的线程数。
	识别的总线程数	表示启动以来 CLR 识别的线程总数。
	争用率 / 秒	表示运行时中的线程尝试获取受管锁定的失败率

	计数器	描述
加载中	**当前组件**	表示进程中加载的组件数。
	组件率	表示将组件加载到内存中的每秒速率。
	加载程序堆中的字节数	表示类加载程序承诺的字节数。
安全性	总计运行时检查数	表示在执行运行时代码访问安全中花费的已用时间百分比。
	堆栈走查深度	表示上次运行时代码访问安全检查的堆栈深度。

异常计数器

此部分描述提供 .Net 抛出的异常相关信息的计数器。

每秒抛出的异常数

正式名称	.NET CLR 异常 / 每秒抛出的异常数 (_Global_)
计数器类型	瞬时 （每个度量时段采样一次）。
描述	此计数器显示每秒抛出的异常数。这些异常包括 .NET 异常和非受管的异常。
使用备注	此计数器包括已处理和未处理的异常。异常只应出现在偶然情况下，而非程序的正常控制流中。
性能	由于抛出异常率过大而产生潜在性能问题的指示器
操作	正常环境下必须是 0。
阈值	如果该比率大于 100 则出错

抛出至捕获深度 / 秒

正式名称	.NET CLR 异常 / 抛出至捕获深度 / 秒
计数器类型	瞬时 （每个度量时段采样一次）。
描述	显示从抛出 .NET 异常的帧遍历到处理该异常的帧的每秒堆栈帧数。
使用备注	输入异常处理程序时重置为 0；因此嵌套的异常会显示处理程序到处理程序的堆栈深度。
性能	用于判断能暴露出潜在瓶颈的代码缺点的辅助指示器。
操作	N/A
阈值	N/A

内存计数器

此部分描述属于 .Net CLR 的内存管理的计数器。它们提供有关内存使用、内存池和垃圾回收统计信息的数据。

公共语言运行时的垃圾回收器 (GC) 管理应用程序内存的分配和释放。这一自动内存管理功能可以消除常见问题，如忘记释放对象而导致内存泄漏，或试图访问已释放对象的内存。

初始化 .Net 应用程序时，运行时为该进程保留连续的地址空间区。此保留的地址空间称为**受管堆**。应用程序创建第一个对象时，在受管堆的基址分配内存。应用程序创建下一个对象时，垃圾回收器在地址空间中紧接第一个对象处为其分配内存。只要有地址空间可用，垃圾回收器就会继续以这种方式为新对象分配空间。从受管堆分配内存比**未受管**的内存分配更快。未受管的资源需要明确清理，因为 GC 不能始终跟踪执行的堆栈。

为优化垃圾回收器的性能，受管堆分成三代：**0 代**、**1 代**和 **2 代**。运行时的垃圾回收器将新对象存储在 0 代中。应用程序生存期中较早创建的对象，经历回收而幸存的，就被提升而存储到 1 代和 2 代中。此方案使垃圾回收器能更快释放特定代中的内存，而不是每次执行回收就对整个受管堆释放内存。

大对象堆大小

正式名称	.NET CLR 内存 \ 大对象堆大小
计数器类型	瞬时 （每个度量时段采样一次）
描述	大对象堆当前使用的内存字节数。
使用备注	垃圾回收器将大于 20 KB 的对象视为大对象，直接分配在特殊堆中，这是由高 CPU 利用率度量的。
性能	判断代码低效率的重要指示器，因为释放整个堆所需时间比分代算法正确运行时所需的更多。通常称为 “碎片化大对象”堆瓶颈。
操作	大对象不通过代提升
阈值	N/A

所有堆中的字节数

正式名称	.NET CLR 内存 \ 所有堆中的字节数
计数器类型	瞬时 （每个度量时段采样一次）
描述	显式当前在 GC 堆上分配的内存字节数。
使用备注	此计数器是 *0 代堆大小*、*1 代堆大小*、*2 代堆大小*和*大对象堆大小*计数器的总和。此计数器表示垃圾回收堆上当前分配的内存字节数。
性能	在内存中使用大数据集时，过多的缓存条目、使用 reg-ex 和字符串解析以及过多的状态查看或会话对象都会增加内存需求。
操作	您通常从该选择计数器开始监控。 *专用字节数*增加而*所有堆中的字节数*计数器保持不变，表示未受管内存有消耗。两个计数器都增加表示受管内存有消耗。
阈值	应小于进程 \ 专用字节数计数器

固定对象数

正式名称	.NET CLR 内存 \ 固定对象数
计数器类型	瞬时 （每个度量时段采样一次）
描述	显示在最后一个 GC 中遇到的固定对象数。
使用备注	固定对象是垃圾回收器无法在内存中移动的对象。此计数器只跟踪已回收垃圾的堆中的固定对象，例如，0 代垃圾回收只会引起 0 代堆中固定对象的枚举。
性能	N/A
操作	N/A
阈值	N/A

GC 中的时间百分比

正式名称	.NET CLR 内存 \GC 中的时间百分比
计数器类型	已用时间
描述	上个 GC 周期以来在执行垃圾回收 (GC) 中已花时间的百分比。
使用备注	此计数器是由垃圾回收器代表应用程序回收和压缩内存所完成的工作的指示器。
性能	分配大字符串给缓存、大量字符串操作等会留给 GC 很多内存空间必须清理。
操作	该计数器值只在每个 GC 结束时更新，反映最后一个观察到的值而不是平均值。如果此计数器中有任何峰值，都会被接受。
阈值	应当在 5-10% 的范围中

线程计数器

线程与程序中执行的线程相关。.NET **逻辑线程**对象通过在代码中发出新的 **System.Threading.Thread** 命令隐式创建或在未受管线程进入受管环境时显式创建。还有物理线程，由 CLR 创建和拥有，实质上用作 .NET 线程对象的基础线程的本机操作系统。 .Net 应用程序可利用并非 CLR 创建的可识别线程；它们在 CLR 以外创建，但至少在 CLR 内运行过一次。

当前逻辑线程数

正式名称	.NET CLR LocksAndThreads\ 当前逻辑线程数
计数器类型	瞬时 （每个度量时段采样一次）
描述	显示应用程序中当前的受管线程对象数。
使用备注	此计数器同时维护运行中和已停止线程的计数。
性能	线程过多可能导致垃圾回收操作缓慢
操作	N/A
阈值	

当前物理线程数

正式名称	.NET CLR LocksAndThreads\ 当前物理线程数
计数器类型	瞬时 （每个度量时段采样一次）
描述	表示由公共语言运行时创建和拥有，用作受管线程对象的基础线程的本机操作系统线程数。
使用备注	这是操作系统进程中的线程的子集。
性能	N/A
操作	此计数器不包括其内部操作中 CLR 使用的线程。
阈值	N/A

当前识别的线程数

正式名称	.NET CLR LocksAndThreads\ 当前识别的线程数
计数器类型	瞬时 （每个度量时段采样一次）
描述	显示当前 CLR 识别的线程数。
使用备注	只跟踪唯一线程。
性能	N/A
操作	使用相同线程 ID 重新进入 CLR 或线程退出后重新创建的线程不会计数两次。
阈值	N/A

识别的总线程数

正式名称	.NET CLR LocksAndThreads\ 识别的线程总数
计数器类型	累积
描述	该应用程序启动以来 CLR 识别的线程总数。
使用备注	只跟踪唯一线程。
性能	N/A
操作	使用相同线程 ID 重新进入 CLR 或线程退出后重新创建的线程不会计数两次。
阈值	N/A

争用率 / 秒

正式名称	.NET CLR LocksAndThreads\ 争用率 / 秒
计数器类型	瞬时 （每个度量时段采样一次）
描述	运行时中的线程尝试获取受管锁定的失败率
使用备注	争用率增加或争用总数显著增加表示应用程序肯定正遭遇线程争用问题。为解决这个问题，必须找出访问共享资源的代码或使用同步机制。
性能	和争用总数一起，可以指出线程瓶颈所在。
操作	
阈值	N/A

加载计数器

此部分描述最重要的加载计数器。

当前组件

正式名称	.NET CLR 加载 / 当前组件 (_Global_)
计数器类型	瞬时 （每个度量时段采样一次）
描述	表示并记录进程中加载的组件数。
使用备注	此值在当前运行的应用程序中的所有应用程序域上累计。
性能	N/A
操作	如果从多个应用程序域将组件加载为中立域，则此计数器只递增一次。
阈值	N/A

组件率

正式名称	.NET CLR 加载 / 组件率
计数器类型	瞬时 （每个度量时段采样一次）
描述	将组件加载到内存中的每秒速率。
使用备注	此值在当前运行的应用程序中的所有应用程序域上累计。
性能	N/A
操作	如果从多个应用程序域将组件加载为中立域，则此计数器只递增一次。
阈值	N/A

加载程序堆中的字节数

正式名称	.NET CLR 加载 \ 加载程序堆中的字节数
计数器类型	累积
描述	表示所有应用程序域上类加载程序承诺的字节数。
使用备注	承诺的内存是磁盘分页文件中保留的物理空间。
性能	此计数器必须处于稳定状态，否则此计数器中过大的波动表示每个应用程序域上加载的组件太多。
操作	N/A
阈值	N/A

安全计数器

此部分描述最重要的安全计数器。

总计运行时检查数

正式名称	.NET CLR 安全 / 总计运行时检查数
计数器类型	累积
描述	显示应用程序启动以来运行时代码访问安全性 (CAS) 检查的总数。
使用备注	CAS 允许对代码信任到不同程度，并根据代码 ID 强制实施这些不同的信任级别。运行时代码访问安全检查在调用程序要求特定权限时执行。运行时检查在调用程序每次调用时进行，并检查调用程序的当前线程堆栈。
性能	与*堆栈走查深度*计数器一起使用时，此计数器表示为安全检查付出的性能代价。
操作	此计数器在运行时安全检查结束时更新
阈值	N/A

堆栈走查深度

正式名称	.NET CLR 安全 / 堆栈走查深度
计数器类型	瞬时 （每个度量时段采样一次）
描述	该计数器显示上次运行时代码访问安全检查的堆栈深度。
使用备注	运行时代码访问安全检查是通过在堆栈间走查执行的。
性能	N/A
操作	与*运行时检查总数*计数器一起使用时，此计数器表示为安全检查付出的性能代价。
阈值	N/A

第 IV 部分

Web 服务器监控

8

Apache 监控

此章节描述 Apache 监控的最佳实践。

此章节包括：

概述

Apache HTTP 服务器是开源、可配置、可扩展的多平台 Web 服务器。它最初是于 1995 年在 NCSA httpd（HTTP 守护程序）的基础上开发的。 Apache HTTP 服务器后来成为商业网站和基于 Web 的应用程序的最常用 Web 服务器之一。

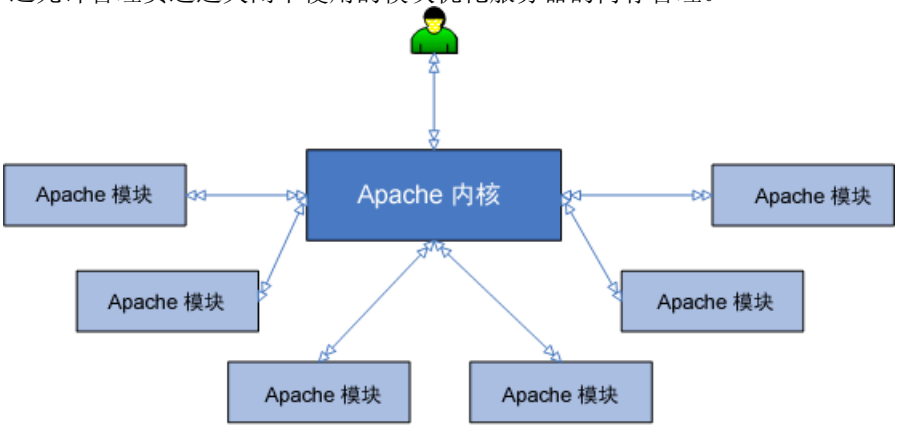
作为主流 Web 服务器之一，重要的是要了解 Apache 的高级体系结构、用于监控的计数器、调优方面以及其他性能相关的最佳实践。本章对这些方面作了总结，同时还使您熟悉用于监控 Apache Web 服务器的 LoadRunner 和 ALM Performance Center 技术。

体系结构

Web 服务器的功能是处理通过 HTTP 协议所作的请求。通常服务器接收请求特定资源的请求，并作为响应将资源返回到客户端。Apache 通过将请求处理责任分到 Apache 核心和 Apache 模块中来达到此目的：

- 核心负责定义和执行处理请求时涉及的步骤。
- 模块实际上实施处理请求时涉及的不同阶段。

此体系结构使 Apache 成为第三方的良好平台，不仅可以覆盖或扩展功能，而且还允许管理员通过关闭不使用的模块优化服务器的内存管理。



高级概念体系结构

尽管 Apache 2.0 和 Apache 1.3 都被视为生产质量版本，但前者在体系结构和功能上要优于后者。以下内容描述 Apache 2.0 体系结构中的性能相关特征：

- **多个处理模块。**不同于 Apache 1.3， Apache 2.0 支持多个处理模块 (MPM)，它基于进程并在启动时派生出数个子进程。 MPM 意味着 Apache 可以配置为完全基于进程的服务器、完全线程化的服务器或这些模型的混合。线程包含在进程中，并且是同时运行的，大多数情况下，线程化的服务器与基于进程的服务器相比，扩展性更好。
- **模块和筛选。**如前所述，Apache 维护模块体系结构。Apache 2.0 还增加另外的扩展机制：筛选。筛选允许模块修改由其他模块生成的内容。它们可以进行加密、扫描病毒或压缩静态文件以及动态生成的内容。

- **Apache 可移植运行时。**由于 Apache 可移植运行时（Apache Portable Runtime，APR）库，Apache 2.0 在 Windows 和 UNIX 平台上的运行同样地好。它将操作系统之间的差异（例如文件或网络访问 API）抽象化。此抽象层还带来了特定于平台的调整和优化。APR 使用内存池的概念，可以大幅简化内存管理代码和减少内存泄漏的可能性。

由 Apache 提供的用于了解和监控服务器状态的计数器遵循 Apache 体系结构，位于 Apache **mod_status** 模块中。 status 模块提供有关服务器活动和性能的信息。它以便于阅读的表单（即 http://your.server.name/server-status）或面向自动执行监控过程的机器可读的简单列表（即 http://your.server.name/server-status?auto）在 HTML 页面中提供服务器统计信息。这两种模式都可以配置为自动刷新状态，方法是在 URL 查询字符串中添加刷新参数（例如，http://your.server.name/server-status?auto&refresh=30 将每隔 30 秒自动刷新一次机器可读状态）。

mod_status 模块可以配置为提供扩展的状态。默认情况下它是禁用的。

注：Apache 监控器连接到 Web 服务器收集统计信息，并且对每次采样都登记一次点击。因此，即使没有客户端连接到 Apache 服务器，Apache 图形也总是显示每秒一次以上点击。

最重要的 Apache 计数器

构建 HP LoadRunner/ALM Performance Center Apache 监控器的目的是跟踪机器可读页面 (server-status?auto) 中提供的计数器。 HP SiteScope 支持这两种模式。在机器可读页面中提供了最重要的计数器。

计数器	描述
CPULoad	由 Apache 服务器消耗的 CPU 的当前百分比
ReqPerSec	每秒请求数 （也叫作每秒点击数）
BytesPerSec	每秒传输的字节数
BytesPerReq	每个请求传输的字节数
BusyWorkers	正在处理请求的活动线程数
IdleWorkers	非活动 / 空闲线程数

提示：在 server-status?auto 页面中提供了所有这些计数器。您可以方便地创建 VuGen 脚本以自行解析这些计数器数据，以及使用 lr_user_data_point 将它联机发送到 LoadRunner/ALM Performance Center

优化和调优

遇到性能问题时，调优和优化对于减轻性能问题是必需的。建议主动采取措施，以将问题扼杀于萌芽状态。此部分列出了面向 Apache Web 服务器的几个可能的调优参数、优化实践和基准方法。在应用任何内容之前，都应先了解根据您的服务器生成的参数和工作负载，来验证配置是否适合您的特定情况。

- HostnameLookups 指令应当关闭（默认情况下关闭）。DNS 查找在打开时会耗用大量时间并使服务器变慢。

- KeepAlive 指令应当开启（默认情况下开启）。KeepAlive 提供更长时间的 HTTP 会话，这样就允许多个请求通过同一 TCP 连接发送。这已经证明对于加快响应时间非常重要。

- KeepAliveTimeout 指令表示 Apache 在关闭连接之前等待后续请求的秒数。默认配置是 15 秒。超时值越高，保留的等待与空闲客户端建立连接的服务器线程就越多。

- 避免使用 .htaccess 文件。可以通过将 AllowOverride 指令设置为 **none** 完全禁用 .htaccess 文件。如果 AllowOverride 设置为允许使用 .htaccess 文件，则 Apache 会在每个目录中查找 .htaccess 文件。无论您实际上是否使用文件，允许 .htaccess 文件都会产生不良性能影响。而且，每次请求文档时都会加载 .htaccess 文件。

- 建议卸载不使用的模块以优化内存利用率。

- MaxKeepAliveRequests 指令。Web 服务器千万不要交换，因为交换会增加每个请求的滞后时间，以至于超出了用户认为“足够快速”的界限。这会导致用户点击“停止”并重新加载，进一步增加负载。您可以（并且应当）控制 MaxClients 设置，以防止服务器衍生出过多的子进程开始进行交换。
  MaxKeepAliveRequests 指令指定将为处理请求而创建的子进程的最大数目，并且限制将处理的并发请求的数目。超出 MaxClients 限制的任何连接尝试通常都要排队等候，直到达到了基于 ListenBacklog 指令的数字。应当将它设置为您的环境可以管理而不会发生吞吐量降级或响应时间的过高增加的客户端最大数目。

9

IIS 监控

此章节描述 IIS 监控的最佳实践。

此章节包括：

概述

Microsoft Internet Information Services (IIS) 是世界流行度排列第二位、仅次于 Apache HTTP 服务器的 Web 服务器。所有 Windows 操作系统版本中都提供 IIS，但具有不同的特点。它不断地发展成熟。 IIS 6.0 较之早期版本有了很大的进步，它使得 IIS 不仅作为 Web 服务器，而且作为应用程序服务器。 IIS 7.0 是最新 IIS 发布，通过增加一些对性能和可靠性也有助益的重要功能，继续这一发展道路。

IIS 包括以下服务器：FTP/FTPS、SMTP、NNTP 和 HTTP/HTTPS。本章着重介绍 HTTP/HTTPS 服务器。其中涵盖 IIS 体系结构、性能监控和某些调优准则。本章的重点主要放在 IIS 6.0 上，尽管必要时也会提到 IIS 7.0。

体系结构

IIS 运行服务器时使用两种不同的请求处理模型（称为应用程序隔离模式）中的一种。应用程序隔离是按进程边界作出的应用程序分离，可防止应用程序或网站彼此影响，并可减少重新启动服务以更正与应用程序相关的问题所花费的时间。

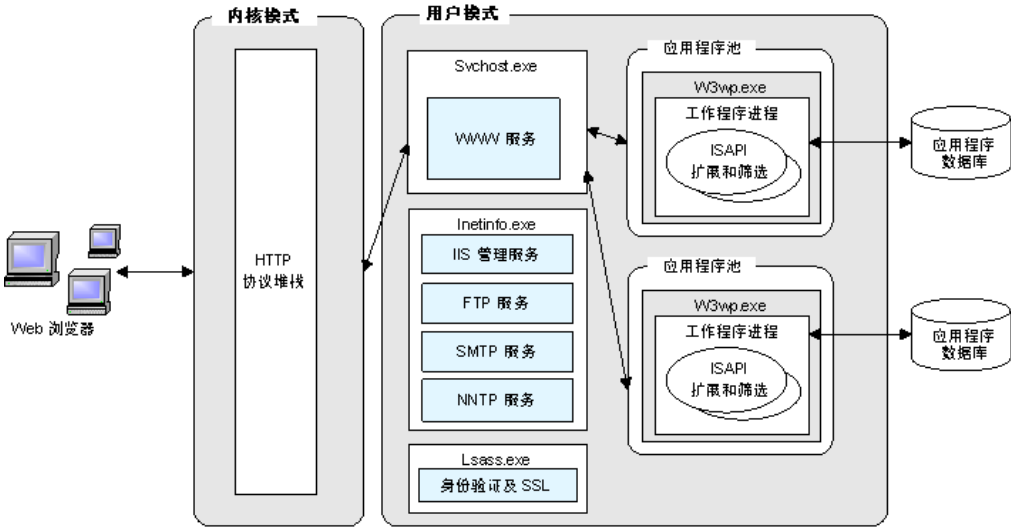
IIS 6.0 支持两种应用程序隔离模式。每种模式有不同配置：

- **工作程序进程隔离模式。**支持将 Web 应用程序分组到应用程序池中，因此使每个应用程序能够在自包含的工作程序进程中实现功能。工作程序进程是用于处理请求 （例如返回静态页或调用 Internet Server API (ISAPI) 扩展或筛选）的用户模式的代码。此模式提供 IIS 6.0 体系结构的所有优点，包括多个应用程序池、运行状况监控和回收、增强的安全性和性能、改进的可扩展性和处理器相似性。
- **IIS 5.0 隔离模式。**为设计在较早版本的 IIS 中运行的应用程序提供兼容性。IIS 6.0 在此模式中运行时，请求处理几乎与 IIS 5.0 中的请求处理完全相同。除非应用程序不能在工作程序进程隔离模式中工作，否则不推荐使用此模式。

这两种模式都依赖于 HTTP 协议堆栈 (HTTP.sys) 接收 HTTP 请求和返回响应。HTTP.sys 侦听 HTTP 请求，对请求排队，并在处理请求之后返回响应。

HTTP.sys 驻留在运行操作系统代码（例如设备驱动程序）的内核模式中。这确保操作系统以高优先级管理 HTTP 请求。请求的实际处理由相关工作程序进程在用户模式中完成。

下图说明了工作程序进程隔离模式。



应用程序池可以受管多个工作程序进程，从而提供负载平衡和故障转移功能。这有助于应用程序的性能、可靠性和可扩展性。包含多个工作程序进程的应用程序池称为 Web 园区 (Web garden)。

如先前所述， IIS 提供四种 Internet 服务：万维网发布服务 （WWW 服务），用于受管 Internet 和内部网内容；文件传输协议 (FTP) 服务，用于受管可供用户上载和下载文件的站点；网络新闻传输协议 (NNTP) 服务，用于受管讨论组；以及简单邮件传输协议 (SMTP) 服务，用于发送和接收电子邮件消息。建议禁用 / 卸载不使用的服务以减少 IIS 内存占用。

IIS 7.0 引入了一些体系结构增强功能：

- 新增了 Windows 进程激活服务 (WAS)。使站点能够使用 HTTP/HTTPS 以外的其他协议。
- 集成了来自 IIS 和 ASP.NET 的请求处理管道。此功能与 IIS 7.0 中支持的应用程序池模式相关。
  - 不再支持 IIS 5.0 隔离模式
  - 继续支持 IIS 6.0 工作程序进程隔离模式
  - 增加了集成应用程序池模式，用以实现 IIS 和 ASP.NET 的集成请求处理
- 可通过添加或删除模块对 Web 服务器引擎进行自定义

监控

IIS 性能计数器是通过 Microsoft Windows 性能数据助手库（pdh.dll，它是 Windows 的常规监控平台）提供的。这意味着每个 IIS 性能计数器都是数字类型，并且按其路径唯一标识，通常使用以下语法：

```
\\ 计算机名称 \Object(Parent/Instance#Index)\Counter
```

路径的**计算机名称**部分是可选的。

SiteScope 和 LoadRunner 都使用 Windows pdh 接口来监控 IIS 和 ASP/ASP.NET 相关计数器。要从远程计算机调用 pdh 接口， Windows 要求通过具备相应权限的用户进行身份验证。

作为最佳实践，建议您好好了解应用程序体系结构和部署。在产品整个生命周期中执行不同性能工程实践时，此信息都很有用。例如，如果没有使用 IIS FTP 服务器或 Frontpage 服务器扩展，则没有理由让它们运行。同样，如果应用程序完全基于 ASP.NET，则没有理由监控 Active Server Pages。

最重要的 IIS 计数器

此部分所列的计数器包括有关性能和工作负载特征的最重要计数器。它们不包括不与 IIS 6.0 兼容的计数器。

注：监控基于 Web 的 .NET 应用程序时，建议也监控 .NET CLR。有关 .NET CLR 监控器的重要计数器的列表，请参见第 7 章 “.Net 平台监控”。

WWW 服务

Web 服务计数器帮助确定万维网发布服务（WWW 服务）处理请求的情况。WWW 服务是用户模式服务。这些计数器还反映在内核模式驱动程序 **HTTP.sys** 中发生的处理。

可以按每个网站配置此类计数器，也可以选择 **_Total** 实例以整个服务器的全局方式进行配置。

计数器	描述
发送的字节数 / 秒	WWW 服务发送数据字节的秒速率
接收的字节数 / 秒	WWW 服务接收数据字节的秒速率
当前连接数	与 WWW 服务的活动连接的数目
“找不到”类型的错误数 / 秒	服务器由于找不到所请求的文档而不能满足请求的秒速率
“已锁定”类型的错误数 / 秒	由于所请求的文档已被锁定而不能满足请求的秒速率

计数器	描述
当前 ISAPI 扩展请求数	WWW 服务正在同时处理的 ISAPI 扩展请求的数目
ISAPI 扩展请求数 / 秒	WWW 服务处理 ISAPI 扩展请求的秒速率

WWW 服务缓存

WWW 服务和 FTP 服务不共享共同缓存。而是将缓存拆分为两个不同的性能对象：一个用于 FTP 服务，一个用于 WWW 服务。WWW 服务缓存计数器仅用于监控服务器性能；因此，无法配置它们监控单个站点。

计数器	描述
当前文件缓存内存使用情况	当前用于用户模式文件缓存的字节数
当前缓存文件数	内容当前在用户模式缓存中的文件数
当前缓存 URI 数	当前存储在用户模式缓存中的 URI 信息块数
当前缓存元数据	在用户模式缓存中的当前元数据信息块数
内核：URI 缓存点击数 / 秒	每秒发生的内核 URI 缓存平均点击次数

ASP.NET

ASP.NET 支持以下 ASP.NET 系统性能计数器，这些计数器聚合一台 Web 服务器计算机上所有 ASP.NET 应用程序的信息，或者，这些计数器通常也适用于运行相同应用程序的一系列 ASP.NET 服务器。

注：并非所有这些计数器在所有 IIS 部署中都可用。

计数器	描述
断开连接的请求数	由于发生通信故障而断开连接的请求数。
排队的请求数	队列中等待处理的请求数。如果此数字随客户端请求数的增加而增加，则说明 Web 服务器已经达到它可以处理的并发请求的限制。此计数器的默认最大值是 5,000 个请求。可以在计算机的 Machine.config 文件中更改此设置。
被拒绝的请求数	由于没有足够的服务器资源处理请求而未执行的请求总数。此计数器表示返回 503 HTTP 状态码的请求数，此代码表示服务器太繁忙。
错误总数 / 秒	HTTP 请求执行期间的每秒平均错误数。包括任何解析器、编译或运行时错误。
输出缓存变动率	输出缓存中每秒发生的添加和删除的平均次数。如果变动率很大，说明缓存未被有效使用。
活动会话数	处于活动状态的会话数。仅对 “在内存中” 会话状态支持此计数器。

计数器	描述
事务数 / 秒	每秒启动的平均事务数。
待定事务数	正在进行的事务数。

Active Server Pages

如果服务器上正在运行 Active Server Pages (ASP)，则 ASP 计数器可以帮助确定服务器或站点响应 ASP 请求的情况。 ASP 计数器用于监控服务器性能，无法监控单个 ASP 应用程序，因为 ASP 计数器跨整个 WWW 服务收集全局数据。

计数器	描述
错误数 / 秒	每秒发生的平均错误数。
请求数 / 秒	每秒执行的平均请求数。
执行的请求数	当前正在执行的 ASP 请求数（例如，活动工作线程的数目）。
排队的请求数	正在等待处理的排队 ASP 请求数。由元数据库属性 AspRequestQueueMax 确定此计数器的最大数。
事务数 / 秒	每秒启动的平均事务数。

优化和调整

遇到性能问题时，调优和优化对于减轻性能问题是必需的。在大多数情况下，应用程序代码优化是必需的，但有时修复未得到良好调整的环境可以大幅改进性能。

此部分列出几个可能的调整做法。一些做法面向 IIS Web 服务器，而另一些则通用于所有 Web 服务器。有许多其他的调整做法可能对您的应用程序更有效。

调整需要漫长的测试和解析的迭代过程。任何配置更改都需要仔细验证。在应用以下任何做法之前，都应先了解根据您的服务器生成的参数和工作负载，来验证配置是否适合您的特定应用程序。

- 调优连接限制。如果连接数很多并且伴有高 CPU 利用率和高处理器队列长度，则表明存在 CPU 瓶颈。应当限制允许的最大连接数，或增大 CPU 处理能力。
- 关闭 ASP 调试。验证将 **AppAllowDebugging** 和 **AppAllowClientDebug** 设置为 **false** 后，服务器端和客户端是否都已关闭。
- 将 **AspBufferingOn** 设置为 **true** 以便在 ASP 输出缓存发送到客户端前对它进行收集。
- **AspProcessorThreadMax Metabase** 属性指定 IIS 可以创建的每个处理器工作线程的最大数目。要找出 IIS 对于每个 ASP 进程允许的工作线程最大数目，请将此值乘以服务器上的处理器数。如果减小此值，请监控性能以确保较低的线程限制不会降低性能。如果性能降低，则再次增大该值。
- **AspRequestQueueMax Metabase** 属性指定队列中允许的 ASP 请求的最大数目。默认值是 3,000，而最佳设置取决于应用程序的行为。如果请求的执行时间非常短并且在队列中的时间也短，则减小此值是合理的。

- 验证每个 TCP 连接的保持活动 (keep-alive) 状态是否已启用 (**connection = keep-alive**)。如果关闭了保持活动连接，则每个文件都需要新的 TCP 连接。在 IIS 中启用 HTTP 保持活动，对于小文件来说实际上使往返数翻倍。
- 启用 HTTP 压缩以增加带宽使用的效率。
- 为所有图像和 HTML 设置 HTTP 到期标头，以使代理服务器和浏览器减少 Web 服务器调用次数。
- 删除不需要的文件内容。删除不需要的空行、制表符、字符等等。较大的文件会影响通过网络传输文件的时间。
- 尽量使用静态文件，以尽可能地降低处理器需求。
- 建立 Web 园区，这些园区是可以运行多个工作程序进程的应用程序池。

第 V 部分

应用程序服务器监控

10

WebLogic 监控

此章节描述 WebLogic 监控的最佳实践。

此章节包括：

概述

Oracle WebLogic 是顶级 J2EE 应用程序服务器之一。WebLogic 体系结构和基础结构面向性能和可扩展性，允许部署很多类型的分布式应用程序（例如，基于 Web 的应用程序和 Web 服务）。此外， WebLogic 对 Sun Microsystems Java EE 5.0 规范的完整实现，为创建可访问各种服务 （例如，数据库、消息传送服务和与外部企业系统的连接）的分布式 Java 应用程序提供了一组标准 API。

这些能力与其他能力一起，使 WebLogic 应用程序服务器成为可以从性能角度来了解的一种重要环境。此章节描述 WebLogic 应用程序服务器高级体系结构、建议的监控计数器以及与调优相关的一些主要方面。

体系结构

WebLogic 有不同产品配置：

- **WebLogic Server。**提供 J2EE 应用程序的核心服务和基础结构。
- **WebLogic Enterprise。**由 WebLogic Server 和 BEA Tuxedo 软件组成。
- **WebLogic Express。**是 WebLogic Server 的非 J2EE 的 “轻量型”版本。

此章节着重介绍 WebLogic Server。

若要了解 WebLogic 体系结构和部署，需要熟悉 WebLogic Server 域。

WebLogic Server 域是 WebLogic Server 资源的逻辑相关组。它包括名为管理服务器的特殊 WebLogic Server 实例，以及名为被管服务器的其他 WebLogic Server 实例。 WebLogic Server 实例可以部署为管理服务器或被管服务器。

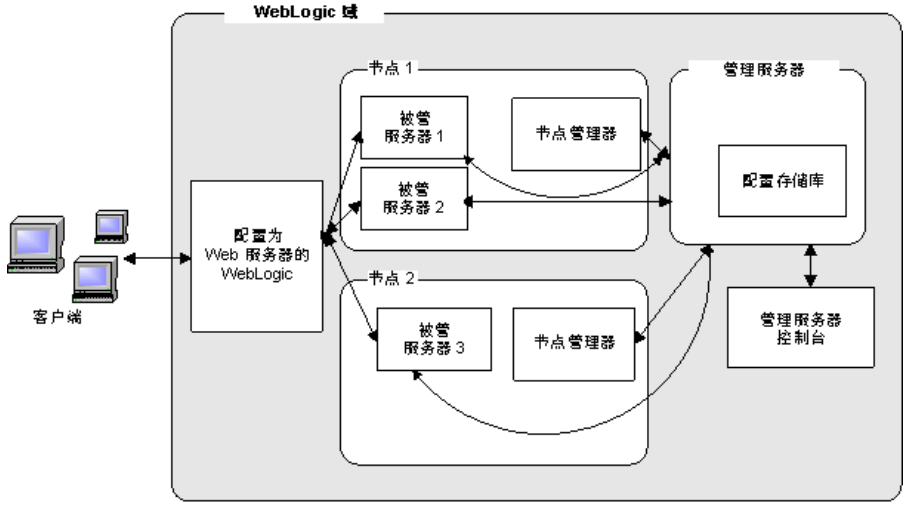
管理服务器仅仅用于管理和监控整个域，而被管服务器则受管和执行部署于其中的应用程序逻辑。每个被管服务器都使用 Oracle JRockit JVM 在自身的 Java 进程下运行。管理服务器也是如此。

除管理服务器和被管服务器以外，域还包含被管服务器和应用程序需要的其他资源和服务。节点管理器就是此类资源之一。节点管理器与计算机 （而不是与逻辑实体）关联，允许域的管理服务器控制计算机上部署的被管服务器。

您可以使用单个 WebLogic Server 安装创建并运行多个域，也可以使用多个安装运行单个域。重要的是要了解一些注意事项，然后才能在环境中进行 WebLogic Server 域配置，因为这一配置通常会影响性能和可扩展性。

每个 WebLogic Server 都可以配置为利用自身的 HTTP 监听器（支持 HTTP 1.1）的 Web 服务器。或者也可以使用 Apache、Microsoft IIS 和 Netscape Web 服务器。 Web 服务器配置使 WebLogic Server 除了能够处理 servlet 或 JSP 生成的动态内容以外，还能处理静态 HTML 内容的请求。

下图说明的是一个由两个计算机/节点上部署的三个被管服务器组成的 WebLogic 域。



监控

WebLogic Server 管理系统基于 Sun 的 Java 管理扩展 (JMX) 标准，通过一组被管 bean (MBean) 提供管理、运行状况和性能数据。这些 MBean 可以使用 JXM 或 SNMP 查询。另外，WebLogic Server 在日志文件中记录关于配置更改和子系统故障的信息。这些日志文件对调查关键故障会很有用，但对于负载则相关性较小。

最新 WebLogic 版本包括名为 WebLogic Diagnostics Framework (WLDF) 的诊断框架。 WLDF 利用前面提到的 MBean，并且增添了其他功能，包括：

- 捕获诊断快照用于故障后分析
- 将来自服务器实例和应用程序的数据事件、日志记录和度量存档
- 测评服务器及其运行的应用程序

检查产品整个生命周期的性能时，重要的是彻底了解应用程序的体系结构和部署。这在监控 WebLogic Server 上部署的 J2EE 应用程序时尤其如此。例如，除非在群集模式中配置 WebLogic Server 部署，否则群集监控器是不相关的。

HP SiteScope WebLogic 解决方案模板是使用 LoadRunner 或 ALM Performance Center 时监控 WebLogic 的建议方法。SiteScope WebLogic 解决方案基于 SiteScope WebLogic 监控器和 JMX 监控器，并带有一组用于监控的预定义计数器。它使用 JMX 接口来监控 WebLogic，因此需要 WebLogic 管理员进行安全访问配置。

或者也可使用 SiteScope WebLogic 监控器来监控 WebLogic 6.x、7.x 和 8.x，使用 SiteScope JMX 监控器来监控 WebLogic 9.x 或 10.x。配置所需计数器时，后者需要更多的手动工作量。详细说明请参见《SiteScope 用户指南》(SiteScope User Guide)。

最重要的 WebLogic 计数器

下面的计数器列表包括有关性能和工作负载特征的最重要计数器。 WebLogic 还提供其他计数器；要监控它们，可以从相关 MBean 进行选择。

计数器按不同实体并根据 WebLogic MBean 分类。

注：计数器可能有所不同，具体取决于应用程序服务器上安装的内容。

服务器

当工作进入 WebLogic Server 时，将放在执行队列中。此工作然后分配到队列中执行工作的线程。

以下计数器帮助评估服务器处理工作负载的能力，并识别是执行队列还是线程池与潜在瓶颈相关。

WebLogic Mbean:

计数器	描述
MBean: weblogic.management.runtime.ServerRuntimeMBean	
OpenSocketsCurrentCount	此服务器上已注册用于套接字混合的当前套接字数。
MBean: weblogic.management.runtime.ExecuteQueueRuntimeMBean	
ExecuteThreadCurrentIdleCount	分配到队列的空闲线程的数目。
ExecuteThreadTotalCount	分配到队列的执行线程的总数。
PendingRequestCurrentCount	队列中的待定请求的数目。

计数器	描述
MBean: weblogic.management.runtime.ThreadPoolRuntimeMBean	
ExecuteThreadIdleCount	池中的空闲线程的数目。此计数不包括备用线程和挂起线程。计数指示到达时已准备好执行新工作的线程。
ExecuteThreadTotalCount	池中的线程总数。
PendingUserRequestCount	优先级队列中的待定用户请求的数目。优先级队列包括来自内部子系统和用户的请求。这是所有用户请求的计数。
QueueLength	优先级队列中的待定请求的数目。这是内部系统请求和用户请求的总计。
Throughput	每秒完成的平均请求数。
StandbyThreadCount	返回备用池中的线程数目。处理当前工作负载时不需要的多余线程被指定为备用线程并添加到备用池。需要更多线程时，将激活这些线程。

EJB

Enterprise JavaBean 是封装业务逻辑的服务器端组件。这使它们很有可能存在性能瓶颈，因此是重要的监控对象。

主要有两种类型的 bean：会话 Bean 和消息驱动 Bean，其中会话 Bean 可以是有状态或无状态。

WebLogic MBean:

计数器	描述
MBean: weblogic.management.runtime.EJBCacheRuntimeMBean 监控实体 Bean 和有状态 Bean 的缓存计数器。	
ActivationCount	提供来自此 EJB Home 的已激活的 bean 总数。
CacheAccessCount	提供尝试从缓存访问 bean 的总次数。 **注：**缓存点击计数与缓存错失计数的总和，可能不会加总到正在运行服务器的 CacheAccessCount，因为这些度量是使用多个调用检索的，这些计数在调用之间可能发生变化。
CachedBeansCurrentCount	提供来自此 EJB Home 的当前在 EJB 缓存中的 bean 总数。
CacheMissCount	提供未能从缓存访问 bean 的尝试总次数。 **注：**缓存点击计数与缓存错失计数的总和，可能不会加总到正在运行服务器的 CacheAccessCount，因为这些度量是使用多个调用检索的，这些计数在调用之间可能发生变化。

计数器	描述
PassivationCount	提供来自此 EJB Home 的已挂起的 bean 总数。
MBean: weblogic.management.runtime.EJBLockingRuntimeMBean	
LockEntriesCurrentCount	提供当前锁定的 bean 计数。
LockManagerAccessCount	提供尝试获取 bean 锁定的总次数。这包括获取已代表客户端锁定的 bean 锁定的尝试次数。
TimeoutTotalCount	提供等待 bean 锁定已超时的当前线程数。
WaiterCurrentCount	提供已等待 bean 锁定的当前线程数。
MBean: weblogic.management.runtime.EJBPoolRuntimeMBean 监控 EJB 实例的实体 Bean、消息驱动 Bean 和无状态 Bean	
AccessTotalCount	提供从可用池检索实例的尝试总次数。
BeansInUseCurrentCount	提供可用池中当前被使用的 bean 实例计数。
DestroyedTotalCount	提供此池中的 bean 实例由于引发非应用程序异常而被销毁的总次数。
MissTotalCount	提供未能从可用池获取实例的尝试总次数。如果池中没有可用实例，从池中获取 bean 的尝试失败。

计数器	描述
PooledBeansCurrentCount	提供可用池中可用 bean 实例的当前计数。
TimeoutTotalCount	提供等待可用池中可用 bean 实例时已超时的线程总计数。
WaiterCurrentCount	提供当前正在等待可用池中可用 bean 实例的线程计数。
MBean: weblogic.management.runtime.EJBTransactionRuntimeMBean 监控实体 bean、消息驱动 bean、无状态 bean 和有状态 bean 的事务计数器	
TransactionsCommittedTotalCount	提供此 EJB 已提交的事务总计数。
TransactionsRolledBackTotalCount	提供此 EJB 已回滚的事务总计数。
TransactionsTimedOutTotalCount	提供此 EJB 已超时的事务总计数。

Servlet

WebLogic MBean:

计数器	描述
MBean: weblogic.management.runtime.ServletRuntimeMBean	
ExecutionTimeAverage	提供 servlet 自创建以来已执行的所有调用的平均时间。

JRockit

这些计数器仅当将服务器与 JRockit 虚拟机一起运行时才可用，对反映应用程序性能的特征以及进行调优都很重要。

计数器	描述
MBean: weblogic.management.runtime.JRockitRuntimeMBean	
UsedHeap	指示虚拟机当前正在使用的 Java 堆内存的数量（字节）。
UsedPhysicalMemory	指示主机计算机上当前正在使用的物理内存的数量（字节）。此值报告计算机上的所有进程（而不只是 VM）使用的内存。
TotalNurserySize	指示当前分配给收容所 (nursery) 的内存数量（字节）。 收容所是 VM 分配给大多数对象的 Java 堆所在区域。分代垃圾收集器重点处理收容所，而不是收集整个堆的垃圾。因为大多数对象的生存期很短，很多时候仅对收容所（而不是整个堆）进行垃圾收集就已足够。 如果没有使用分代垃圾收集器，则收容所大小是 0。
AllProcessorsAvgLoad	显示主机计算机中所有处理器的平均负载的快照。如果计算机只有一个处理器，此值与 JVMProcessorLoad 相同。 值以双精度数返回，1.0 表示 100% 负载（没有空闲时间），0.0 表示 0% 负载（纯空闲时间）。

计数器	描述
JVMProcessorLoad	显示 VM 放置在主机计算机中所有处理器上的负载的快照。如果主机包括多个处理器，值表示平均负载的快照。 值以双精度数返回，1.0 表示 100% 负载（没有空闲时间），0.0 表示 0% 负载（纯空闲时间）。
TotalNumberOfThreads	指示所有处理器上当前正在 VM 中运行的 Java 线程（守护程序和非守护程序）数目。
NumberOfDaemonThreads	指示所有处理器上当前正在 VM 中运行的守护程序 Java 线程数目。

JDBC 连接池

Java 数据库连接 (JDBC) 是与数据库对接和执行 SQL 语句的标准 Java API。

数据库通常是性能瓶颈，应该从所有角度进行仔细监控。以下计数器与 JDBC 连接池相关。它们有助于了解负载下的数据库行为。

计数器	描述
MBean: weblogic.management.runtime.JDBCDataSourceRuntimeMBean	
ActiveConnectionsAverageCount	数据源的此实例中活动连接的平均数。活动连接是应用程序正在使用的连接。
ActiveConnectionsCurrentCount	应用程序当前正在使用的连接数。

计数器	描述
ConnectionDelayTime	创建与数据库的物理连接所耗用的平均时间（毫秒）。该值是将用于连接的所有时间的总和除以连接总数而计算出来的。
CurrCapacity	数据源中已在连接池中的 JDBC 连接的当前计数。
LeakedConnectionCount	泄漏连接的数目。泄漏连接是已在数据源上保留但是调用 close() 后未返回到数据源的连接。
NumAvailable	此数据源中当前可用（不在使用中）的数据库连接数目。
NumUnavailable	数据源的此实例中当前不可用（正在使用或正由系统测试）的数据库连接数目。
PrepStmtCacheHitCount	使用了缓存中的语句的累计运行次数。
PrepStmtCacheMissCount	缓存中的语句不能满足语句请求的次数。
WaitingForConnectionCurrentCount	正在等待数据库连接的连接请求数。

JMS

WebLogic JMS 是紧密集成到 WebLogic Server 平台的企业级消息传送系统。

仅当应用程序使用 WebLogic JMS 时，以下计数器相关。在这种情况下，这些计数器对确定 JMS 服务器是否为性能瓶颈非常有用。

WebLogic MBean:

计数器	描述
MBean: weblogic.management.runtime.JMSRuntimeMBean	
ConnectionsCurrentCount	与 WebLogic Server 的当前连接数。
MBean: weblogic.management.runtime.JMSServerRuntimeMBean	
BytesCurrentCount	此 JMS 服务器上存储的当前字节数。此数字不包括待定字节数。
BytesPageableCurrentCount	所有消息中当前可供进行页调出但是尚未进行页调出的字节总数。 JMS 服务器尝试使此数字小于 “MessageBufferSize” 参数。
BytesPendingCount	此 JMS 服务器上存储的当前待定（未确认或未提交）字节数。待定字节数是当前字节数以外的字节数目。
BytesReceivedCount	自上次重置以来此 JMS 服务器上接收的字节数。
DestinationsCurrentCount	此 JMS 服务器的当前目标数。
MessagesCurrentCount	此 JMS 服务器上存储的当前消息数。此数字不包括待定消息。

计数器	描述
MessagesPageableCurrentCount	此 JMS 服务器中当前可供进行调页但是尚未进行页调出的消息数。
MessagesPendingCount	此 JMS 服务器上存储的当前待定（未确认或未提交）消息数。待定消息数是当前消息数以外的消息数目。
MessagesReceivedCount	自上次重置以来此目标上接收的消息数。
SessionPoolsCurrentCount	此 JMS 服务器上实例化的当前会话池数。

JTA

WebLogic 的一个基本功能是事务管理，它可保证数据库更改已高度完整地准确完成。

以下计数器对尝试评估服务器和应用程序可以维持的工作负载很有用。

提示：评估回滚事务率。如果比率高于预期，应该进行调查，方法是了解回滚的原因，然后将它与操作系统、应用程序服务器、数据库服务器和 LoadRunner 事务中测得的其他计数器进行关联。

计数器	描述
MBean: weblogic.management.runtime.JTARuntimeMBean	
TransactionTotalCount	已处理事务的总数。此总计包括所有已提交、已回滚和启发式完成的事务。
TransactionCommittedTotalCount	已提交的事务数。
TransactionRolledBackTotalCount	已回滚的事务数。
TransactionRolledBackTimeoutTotalCount	由于超时时间已到而回滚的事务数。
TransactionRolledBackResourceTotalCount	由于资源错误而回滚的事务数。
TransactionRolledBackAppTotalCount	由于应用程序错误而回滚的事务数。

计数器	描述
TransactionRolledBackSystemTotalCount	由于内部系统错误而回滚的事务数。
TransactionHeuristicsTotalCount	完成时具有启发式状态的事务数。
TransactionAbandonedTotalCount	已放弃的事务数。
AverageCommitTime	服务器提交事务所用的平均时间（毫秒）。
ActiveTransactionsTotalCount	服务器上活动事务的总数。

优化和调优

要解决性能问题，优化和调优非常重要。在大多数情况下，应用程序代码优化是必需的，但有时修复未得到良好调整的环境可以大幅改进性能。

此部分列出几个可能的调整做法。一些做法面向 WebLogic 应用程序服务器，而另一些则通用于任何应用程序服务器。其他许多调优做法也可以改进应用程序的性能。

调整需要漫长的测试和解析的迭代过程。任何配置更改都需要仔细验证。在应用以下任何做法之前，都请先了解根据您的服务器生成的参数和工作负载，来验证配置是否适合您的特定应用程序。

调优池大小

将 EJB、JDBC 和线程相关池调整到合适大小，以增加服务器的容量和提高其性能。要调整这些池，请监控前面提到的相关计数器，并查找等待时间以及 LoadRunner 事务响应时间。记下最佳响应时间。

使用准备语句缓存

准备语句缓存将已编译的 SQL 语句保存到内存中，因此避免了以后使用同一语句时的数据库往返。

JVM 调优

- 检查哪个收集算法更适合应用程序：并发或并行。
- 确定最佳堆大小。
  - 监控峰值负载下的应用程序。
  - 分析收集的发生频率。如果收集的发生频率过高并且可用内存大小降低，则可能需要进行应用程序代码优化。
  - 分析完整 GC 的耗用时间。如果超过 5 秒，则降低堆大小。
  - 分析平均内存占用。如果堆在完整 GC 之后有 85% 可用，则可以降低其大小。

执行队列

如果队列长度和 CPU 的利用率不足，则增加线程计数。这样可以更好地利用 CPU。

常规

- 始终使用 Web 服务器处理 HTML 页面、图像、CSS 文件 和 JavaScript 文件之类的静态内容。这可以减少在应用程序服务器计算机上花费的 CPU 时间，留下更多时间处理其他作业。
- 使用 WebLogic 群集实现可扩展性和高可用性。

11

WebSphere 监控

此章节描述 WebSphere 平台监控的最佳实践。

此章节包括：

概述

IBM WebSphere 应用程序服务器是 IBM WebSphere 平台中的旗舰产品。它是顶级 J2EE 应用程序服务器之一。 WebSphere 体系结构和基础结构面向性能和可扩展性，允许部署很多类型的分布式应用程序（例如，基于 Web 的应用程序和 Web 服务）。此外， WebSphere 对 Sun Microsystems Java EE 5.0 规范的完整实现，为创建可访问各种服务（例如，数据库、消息传送服务和与外部企业系统的连接）的分布式 Java 应用程序提供了一组标准 API。

这些能力与其他能力一起，使 WebSphere 应用程序服务器 (WAS) 成为可以从性能角度来了解的一种重要环境。此章节有助于您了解 WebLogic 应用程序服务器高级体系结构、建议的监控计数器以及与调优相关的一些主要方面。

体系结构

WebSphere 应用程序服务器有五个不同版本：

- **WebSphere Application Server Network Deployment。**通过高级的性能和管理能力，为任务关键的应用程序提供接近连续的可用性。
- **WebSphere Application Server for z/OS。**提供的能力与 Network Deployment 版本相似，但它面向 z/OS，并使用对其有利的 z/OS 工作负载管理器。
- **WebSphere Application Server。**提供 Java EE 5 配置，为可扩展单服务器环境中的简易管理进行了优化。
- **WebSphere Application Server Express**。提供 WebSphere Application Server 这个版本的简略版本。
- **WebSphere Application Server Community Edition。**提供基于开源 Apache Geronimo 的轻量级 Java EE 5 应用程序服务器。

WebSphere 应用程序服务器系列的每个成员使用相同的体系结构，但是在能力、平台兼容性和许可方面存在某些差异。

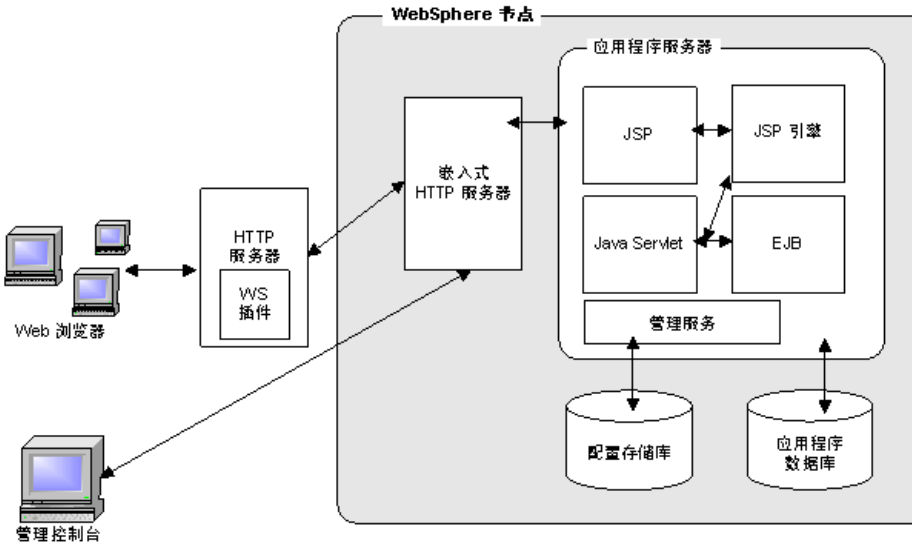
WebSphere 应用程序服务器是基于单元、节点和服务器的概念而进行组织的。单元和节点在您达到 Network Deployment 配置时会起到重要作用。

- **服务器。**服务器执行实际代码执行。服务器有若干种类型，具体取决于配置：应用程序服务器和 JMS 服务器。每个服务器在其自己的 JVM 上运行。
- **节点。**节点是由 WebSphere 管理的共享公共配置和操作控制的服务器进程的逻辑分组。节点通常与 WebSphere 应用程序服务器的一个物理安装关联。
- **单元。**单元是单个管理域中节点的一种分组。

典型 WebSphere 单元包含一些软件组件，这些软件组件可安装在一个节点上，或者出于可扩展性和可靠性目的而分布在多个节点上。这些软件组件包括：

- 提供 HTTP 服务的 Web 服务器
- 用于存储应用程序数据的数据库服务器
- WebSphere 应用程序服务器 (WAS)

下图说明的是单个 WebSphere 节点体系结构。



监控

WebSphere 应用程序服务器提供性能监控基础结构 (PMI)，PMI 是提供客户端 API 的服务器端监控基础结构。使用 PMI 可以监控应用程序服务器的总体运行状况和性能。性能数据是通过 JMX 提供的。

注：PMI 从 WebSphere 管理控制台启用。

检查产品整个生命周期的性能时，重要的是彻底了解应用程序的体系结构和部署。这在监控 WebSphere 服务器上部署的 J2EE 应用程序时尤其如此。例如，仅当应用程序有 Web 服务计数器时， Web 服务计数器才是相关的。

HP SiteScope WebSphere 解决方案模板是使用 LoadRunner 或 ALM Performance Center 时监控 WebSphere 的建议方法。解决方案模板附有一组用于监控的预定义计数器。

或者也可以使用 SiteScope WebSphere 监控器。使用此监控器需要手动配置所需计数器。详细说明请参见 《SiteScope 用户指南》 (SiteScope User Guide)。

最重要的计数器

以下计数器列表包括关于性能和工作负载特征的最重要计数器。WebSphere 还提供其他计数器；要监控它们，可以在配置 SiteScope 监控器时进行选择。

以下计数器根据 IBM WebSphere 的分类而进行分类。

注：计数器可能有所不同，具体取决于应用程序服务器上安装的内容。

Enterprise Java Bean

计数器	键	描述
ReadyCount	beanModule.readyCount	并发就绪 bean（实体和会话）的数目。此计数器在版本 3.5.5+ 和 4.0 中称为 **concurrent active**。
LiveCount	beanModule.concurrentLives	并发存活 bean 的数目。
MethodResponseTime	beanModule.avgMethodRt	bean 方法（home、remote、local）的平均响应时间（ 毫秒）。
ActiveMethodCount	beanModule.activeMethods	并发活动方法的数目 —— 同时调用的方法数目。
MessageCount	beanModule.messageCount	传递到 bean onMessage 方法的消息数目（消息驱动 bean）。
MessageBackoutCount	beanModule.messageBackoutCount	未能传递到 bean onMessage 方法的消息数目（ 消息驱动 bean）。
PooledCount	beanModule.poolSize	池中对象（实体和无状态）数目。
WaitTime	beanModule.avgSrvSessionWaitTime	从池获取 ServerSession 而耗用的平均时间（消息驱动 bean）。

JDBC 连接池

计数器	键	描述
Concurrent waiters	connectionPoolModule.concurrentWaiters	当前正在等待连接的线程数目。
Faults	connectionPoolModule.faults	连接池中的错误（例如超时）总数。
Percent used	connectionPoolModule.percentUsed	池的平均已用百分比。

Java 虚拟机 (JVM)

计数器	键	描述
FreeMemory	jvmRuntimeModule.freeMemory	JVM 运行时中的可用内存。
ProcessCpuUsage	jvmRuntimeModule.cpuUsage	Java 虚拟机的 CPU 使用率 （百分比）。
UsedMemory	jvmRuntimeModule.usedMemory	JVM 运行时中的已用内存。

Servlet 会话

计数器	键	描述
ActiveCount	servletSessionsModule.activeSessions	并发活动会话的数目。如果 WebSphere 应用程序服务器当前正在处理请求，则会话处于活动状态。
LiveCount	servletSessionsModule.liveSessions	当前缓存在内存中的本地会话的数目。

事务

计数器	键	描述
ActiveCount	transactionModule.activeGlobalTrans	并发活动全局事务的数目。
LocalActiveCount	transactionModule.activeLocalTrans	并发活动本地事务的数目。
RolledbackCount	transactionModule.globalTransRolledBack	回滚的全局事务的总数。
LocalRolledbackCount	transactionModule.localTransRolledBack	回滚的本地事务的数目。
GlobalTimeoutCount	transactionModule.globalTransTimeout	超时的全局事务的数目。
LocalTimeoutCount	transactionModule.localTransTimeout	超时的本地事务的数目。

线程池

计数器	键	描述
ActiveCount	threadPoolModule.activeThreads	并发活动线程的数目。
PoolSize	threadPoolModule.poolSize	池中线程的平均数。
PercentMaxed	threadPoolModule.percentMaxed	所有线程都在使用中的时间的平均百分比。
DeclaredthreadHungCount	threadPoolModule.declaredThreadHung	声明挂起的线程数目。

Web 应用程序

计数器	键	描述
ConcurrentRequests	webAppModule.servlets.concurrentRequests	并发处理的请求数。
ServiceTime	webAppModule.servlets.responseTime	servlet 请求的响应时间（毫秒）。
ConcurrentRequests	webAppModule.url.concurrentRequests	与 servlet 关联的 URI 的并发处理请求数。
ServiceTime	webAppModule.url.responseTime	与 servlet 关联的 URI 的平均服务响应时间（毫秒）。

系统

计数器	键	描述
CPUUsageSinceLast Measurement	systemModule.cpuUtilization	自上次读数以来的时间间隔内的平均系统 CPU 利用率。 **注：** ➤ 第一个调用需要执行初始化，因此将返回 0 之类的无效值。所有后续调用将返回预期的值。 ➤ 在 SMP 计算机上，返回的值是所有 CPU 的平均利用率。
FreeMemory	systemModule.freeMemory	系统上可用的实际可用内存的数量。 **注：** ➤ 未分配的实际内存只是可用实际内存的下限，因为很多操作系统会取用一些原来未分配的内存并用于其他的 I/O 缓冲。 ➤ 可以释放的缓冲区内存的确切数量取决于其上运行的平台和应用程序两者。

优化和调优

要解决性能问题，优化和调优非常重要。在大多数情况下，应用程序代码优化是必需的，但有时修复未得到良好调整的环境可以大幅改进性能。

此部分列出几个可能的调整做法。一些做法面向 WebSphere 应用程序服务器，而另一些则通用于任何应用程序服务器。其他许多调优做法也可以改进应用程序的性能。

调整需要漫长的测试和解析的迭代过程。任何配置更改都需要仔细验证。在应用以下任何做法之前，都请先了解根据您的服务器生成的参数和工作负载，来验证配置是否适合您的特定应用程序。

调优池大小

将 EJB、JDBC 和线程相关池调整到合适大小，以增加服务器的容量和提高其性能。要调优这些池，需要监控相关计数器。（请参见“最重要的计数器”（第 203 页））具体而言，请查找并发请求、等待和 LoadRunner 事务响应时间的数量。需要考虑应用程序设计，以避免错误配置。

使用准备语句缓存

准备语句缓存将已编译的 SQL 语句保存到内存中，因此避免了以后使用同一语句时的数据库往返。需要根据要处理的并发请求数以及应用程序的设计，调整准备语句缓存的大小。

JVM 调优

- 检查哪个收集算法更适合应用程序：并发或并行。
- 确定最佳堆大小。
  - 监控峰值负载下的应用程序。
  - 分析收集的发生频率。如果收集的发生频率过高并且可用内存大小降低，则可能需要进行应用程序代码优化。
  - 分析完整 GC 的耗用时间。如果超过 5 秒，则降低堆大小。
  - 分析平均内存占用。如果堆在完整 GC 之后有 85% 可用，则可以降低其大小。

常规

- 始终使用 Web 服务器处理 HTML 页面、图像、CSS 文件 和 JavaScript 文件之类的静态内容。这可以减少在应用程序服务器计算机上花费的 CPU 时间，留下更多时间处理其他作业。
- 禁用非必需的功能。例如，如果应用程序不使用 Web 服务寻址（WS 寻址）支持，禁用此功能可以改进性能。
- 确保将事务日志分配到快速磁盘。

第 VI 部分

数据库资源监控

12

数据库资源监控 —— 简介

大多数现代应用程序都设计为在多层体系结构中运行，在这样的体系结构中，应用程序的功能分布在多个层上，每层通常都在自己的服务器上执行。这些层通常包括 （但不限于）以下各层：

- **用户界面。**充当用户和应用程序之间通信的桥梁。
- **业务层。**与运行应用程序所需的所有业务规则关联。
- **数据层。**处理用于管理业务事务所需的数据。

此结构提供某些重大好处（例如：相对小的客户端内存占用、仅在服务器端部署、功能分离、不直接访问数据库），从而降低开发和拥有应用程序的总成本。

有了这种分布式复杂性，每层都可能导致性能问题。但是，有时候，性能工程师发现最终用户对性能不满意的根本原因在于数据库层响应缓慢。

数据库始终处于变化过程中 —— 无论是数据、查询还是某种逻辑。因此，必须确保数据库的最佳性能，因为这对于当前任何数据驱动的应用程序都是最根本的。

有很多影响总体应用程序性能的因素是源于数据库端的，例如：

- 应用程序开发期间的不良数据库设计
- 数据库表设计中遵循的不良标准
- 数据库使用的不良索引
- 跨数据库表的不良分区数据
- 查询中使用的不良逻辑
- 不正确的存储过程
- 配置不佳的存储硬件
- 专用于多个应用程序的数据库服务器计算机

13

Oracle 监控

此章节描述 Oracle 监控的最佳实践。

此章节包括：

概述

Oracle 数据库是 Oracle 公司制作的关系数据库管理系统 (RDBMS)。Oracle 数据库丰富的功能可实现其高可用性、可扩展性、性能、可管理性和安全性。这些功能使 Oracle 成为企业级的 RDBMS 和该领域中的顶尖领航者之一。

Oracle 数据库由于其不同的功能和特性，具有全面支持应用程序开发的能力。Oracle 还同时提供 Java 和 .NET 的数据访问方法。

Oracle 数据库以若干版本提供，每个版本都针对不同的使用规模：

- **Standard Edition (SE)。**包含基本数据库功能。通常面向运行 1-4 个 CPU 的服务器。如果 CPU 数超过 4，则用户必须转换成企业许可证。 SE 没有内存限制，并可对 Oracle RAC 使用群集功能。
- **Enterprise Edition (EE)。**扩展了 Standard Edition，尤其是在性能和安全领域。面向运行 4 个或更多 CPU 的服务器。EE 没有内存限制，并可使用 Oracle RAC 软件利用群集。
- **Standard Edition One。**在 Oracle 10g 中引入，和 Standard Edition 比起来有一些功能上的限制。面向有 1 个或 2 个 CPU 的系统。它没有内存限制。
- **Express Edition (Oracle Database XE)。**在 2005 中引入，在 Windows 和 Linux 平台上免费分配。它只占用 150 MB，并限制为使用单个 CPU 和最多 4 GB 用户数据。尽管它可以安装在任意内存大小的服务器上，但它限于最多使用 1 GB。
- **Oracle Database Lite。**设计为运行于移动设备上。数据库部分位于移动设备上，可以和基于服务器的安装同步。

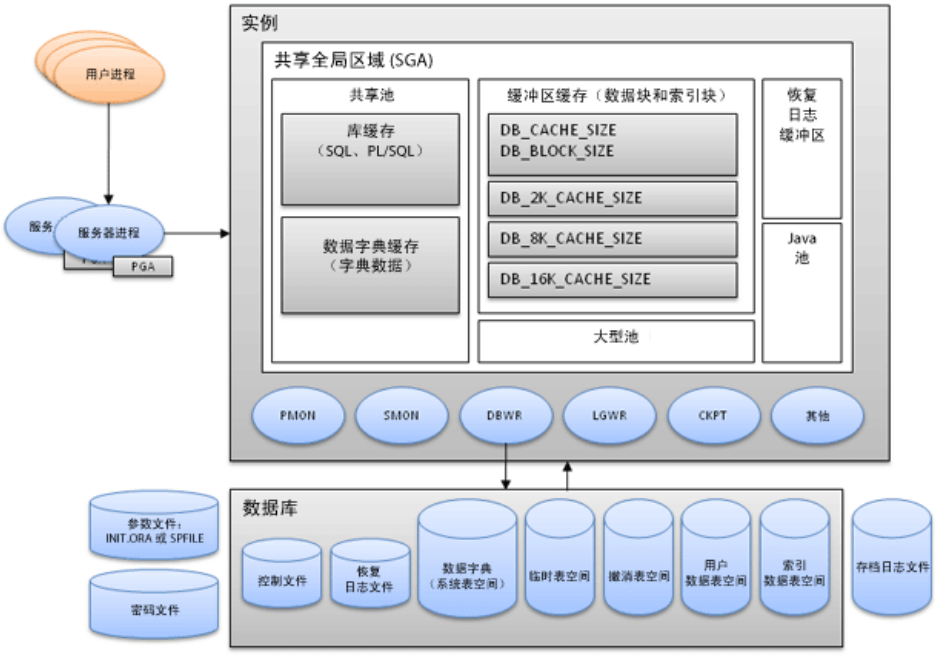
众所周知，数据库层级对应用程序性能有重要影响。从性能角度，Oracle 作为数据库领域的翘楚，是需要熟悉的重要环境。此章节帮助您了解 Oracle 数据库的高级别体系结构，了解其监控能力。它还列出用于监控的最重要计数器，并描述了某些与调整相关的做法。

体系结构

Oracle 数据库由实例和数据存储组成。实例是与存储交互的一组操作系统进程和内存结构。内存结构称为系统全局区域 (SGA)，并逻辑上存储为表空间、物理上存储为数据文件。表空间可包含各类内存段。段又由一个或多个范围组成。范围由连续的数据块组组成，数据块则构成数据存储的基本单位。在物理层面，数据文件包含一个或多个数据块，块的大小可随数据文件而不同。

Oracle 数据库管理借助 SYSTEM 表空间中存储的信息跟踪其计算机数据存储。SYSTEM 表空间包含数据字典，通常 （默认情况下）包含索引和群集。数据字典由包含有关数据库中所有用户对象的信息的一组特殊的表组成。

下图演示 Oracle 数据库的体系结构。它显示实例层面上的不同内存结构和存储层面上的数据文件。



每个 Oracle 实例都使用**系统全局区域 (SGA)**，它是共享的内存区域，用于存储其数据和控制信息。实例在启动时分配给自身一个 SGA，并在关闭时释放它。SGA 中的信息由以下元素组成，每个都有固定大小，并在实例启动时建立：

- **缓冲区缓存。**存储最近用过的数据块。这有助于 Oracle 减少 I/O 并提高性能，因为同样数据的新请求从缓冲区缓存获得，而非从磁盘。
- **重做日志缓冲区。**存储重做条目，即对数据库所做更改的日志。这有助于 Oracle 在系统故障时恢复实例。

- **共享池。**在库缓存中存储共享 SQL 区之类的共享内存结构，在数据字典中存储内部信息。分配给共享池的内存不足可导致性能降低。
  - **库缓存。**存储共享 SQL，为每一 SQL 语句缓存解析树和执行计划。这减少了需要的内存量，减少了用于解析和执行计划的处理时间。
  - **数据字典缓存。**存储类似用户信息、特权、表名称、数据类型等等的信息。数据字典帮助 Oracle 解析 SQL 语句。数据字典中的性能瓶颈会影响所有 Oracle 用户。

程序全局区域 (PGA) 是服务于在客户端计算机上运行的用户进程的服务器端进程。PGA 内存区域包含 Oracle 服务器进程的数据和控制信息。PGA 保存关于用户会话、会话变量、排序、绑定变量等等的信息。

Oracle 通常依赖一组进程，它们同时在后台运行并交互，以监控数据库、增强其性能。以下进程是在实例层面上所运行进程的更长列表的一部分：

- **数据库写入程序进程 (DBWR)。**负责将数据写入磁盘。
- **日志写入程序进程 (LGWR)。**负责将数据写入日志。
- **系统监控进程 (SMON)。**负责实例恢复、释放临时段及合并可用空间区域。
- **进程监控 (PMON)。**负责进程失败之后的清理。
- **检查点进程 (CKPT)。**负责发出有关检查点的信号，更新发生检查点的相关文件。

仅当 Java 代码在实例层面上运行，且“大池”可选时，“Java 池”才相关。在使用“大池”的事件中，它通过存储“共享池”默认存储的部分信息，来减少“共享池”上的开销。

Oracle 体系结构通过将 I/O 操作减少到最低程度，提供可能的最优性能。性能监控和调整应验证部署上的配置是否确实利用了这些能力。

监控

Oracle 提供了若干工具和实用程序，可用于性能监控和调整。

- **自动数据库诊断监控 (ADDM)。**允许 Oracle 数据库诊断自身并确定可能如何解决潜在问题。 ADDM 在每次自动工作负载库 (AWR) 统计信息捕获之后自动运行，使性能诊断数据随时可用。由于 AWR 捕获定期发生，这就确保了数据库会诊断其性能并检测根源。 ADDM 将以下问题视为需要解决的问题：
  - **CPU 瓶颈。**系统 CPU 被 Oracle 或其他某些应用程序完全占用了吗？
  - **内存结构过小。** Oracle 内存结构（如 SGA、PGA 和缓冲区缓存）足够大吗？
  - **I/O 容量问题。** I/O 子系统的表现符合期望吗？
  - **高负载 SQL 语句。**有任何 SQL 语句占用了过多系统资源吗？
  - **高负载 PL/SQL 执行和编译**以及高负载 Java 使用。
  - **RAC 的特定问题。**全局缓存热块和对象是什么；是否有任何互连延迟问题？

- **应用程序对 Oracle 的使用未优化。**是否有连接管理不善、解析过多或应用程序层的锁定争用问题？
- **数据库配置问题。**是否有日志文件大小错误、存档问题、检查点过多或参数设置未优化的证据？
- **并发问题。**是否有缓冲区忙碌问题？
- 各种问题领域的**热门对象和排名考前的 SQL**。

  这使 ADDM 和 AWR 报告成为识别性能问题的非常有意义的工具和调整的起点。

- **Oracle Enterprise Manager。**提供一组系统管理工具，用于管理 Oracle 环境。它有监控 Oracle 环境并使任务自动化的工具。
- **SQL 跟踪。**提供单个 SQL 语句的性能信息。它为每个语句生成以下统计信息：
  - 解析、执行和获取计数
  - CPU 和已用时间
  - 物理读取和逻辑读取
  - 已处理行数
  - 库缓存未点击数
  - 发生每次解析的用户名
  - 每次提交和回滚
- **TKProf。**用于将 SQL 跟踪的输出格式化为可供人阅读格式的实用工具。在调整 SQL 语句时，它很有帮助。它还可以用于确定 SQL 语句的执行计划，创建存储数据库中统计信息的 SQL 脚本。

Oracle 将监控的相关信息存储在不同的统计信息表中。Oracle SQL 语句优化器也使用这些表。例如：

- 会话统计信息， V$SESSTAT
- 系统统计信息， V$SYSSTAT
- V$LATCH， V$BUFFER_POOL_STATISTICS

HP 监控解决方案利用这些表中的数据，允许您在运行性能测试时访问这些数据。建议使用已建议内置计数器的 HP SiteScope Oracle 数据库解决方案。

最重要的 Oracle 计数器

计数器	描述
sorts (disk) (V$SYSSTAT 1/sid) (absolute)	至少需要一次磁盘写入的排序操作数。 需要对磁盘执行 I/O 操作的排序很占用资源。您可能希望增大初始化参数 SORT_AREA_SIZE。
sorts (memory) (V$SYSSTAT 1/sid) (absolute)	完全在内存中执行，不需要任何磁盘写入的排序操作数。除非不排序，否则内存排序是最好的。排序通常由表连接 SQL 操作内的选择条件规范所导致。
db block gets (V$SYSSTAT 1/sid) (absolute)	缓冲区缓存中 INSERT、UPDATE、DELETE 和 SELECT FOR UPDATE 访问的块数。表示块的逻辑读取 （从缓存）。逻辑读取总是包括物理读取。物理读取数最好少一些。

计数器	描述
consistent gets (V$SYSSTAT 1/sid) (absolute)	缓冲区缓存中普通查询（SELECT，无 update 子句）访问的块数。表示块的逻辑读取（从缓存）。逻辑读取总是包括物理读取。物理读取数最好少一些。
physical reads (V$SYSSTAT 1/sid) (absolute)	从磁盘读取的数据块总数。该数字等于**直接物理读取数**加上所有读入缓冲区缓存数。物理读取数最好少一些。此数字必须与逻辑读取数相比，才能计算出缓存点击率。逻辑读取数是数据库块获取 (database block gets) 和一致获取 (consistent gets) 的总和。
physical writes (V$SYSSTAT 1/sid) (absolute)	写入磁盘的数据块总数。该数字等于**直接物理写入数**加上所有从缓冲区缓存写入数。
redo writes (V$SYSSTAT 1/sid) (absolute)	由 LGWR 写入重做日志文件的总数。该统计数字除以**写入的重做块数**等于每次写入的块数。
redo entries (V$SYSSTAT 1/sid) (absolute)	重做条目包含重新构造或重做需要的信息以及由 INSERT、UPDATE、DELETE、CREATE、ALTER 或 DROP 操作对数据库进行的更改。如有必要，重做条目可用于数据库恢复。 重做条目 -> 成功的重做写入。**重做缓冲区分配重试次数 / 重做条目数**的比率应当小于 1%。
redo buffer allocation retries (V$SYSSTAT 1/sid) (absolute)	在重做缓冲区中分配空间需要的重试总数。需要重试，是因为重做写入程序落在后面，或者因为发生了日志切换之类的事件。 重做缓冲区分配重试次数 -> 失败的重做写入。**重做缓冲区分配重试次数 / 重做条目数**的比率应当小于 1%。

计数器	描述
redo log space requests (V$SYSSTAT 1/sid) (absolute)	活动日志文件满， Oracle 必须等待为重做日志条目分配磁盘空间的次数。此类空间是通过执行日志切换创建的。 与 SGA 大小或工作负载提交率相比过小的日志文件可能导致出现问题。发生日志切换时，在切换到新日志文件之前， Oracle 必须确保将所有提交的脏缓冲区写入到磁盘。如果有充满脏缓冲区和小型重做日志文件的大 SGA，则日志切换必须等待 DBWR 将脏缓冲区写入到磁盘才能继续。 还会检查日志文件空间和 V$SESSION_WAIT 中的日志文件空间切换等待事件
parse count (hard) (V$SYSSTAT 1/sid) (absolute)	解析调用的总数 （真实解析）。硬解析在内存使用方面是非常昂贵的操作，因为它需要 Oracle 分配工作堆及其他内存结构，然后生成解析树。 应当尽量少用。硬解析与总数的比率应小于 20%。
parse count (total) (V$SYSSTAT 1/sid) (absolute)	解析调用的总数 （硬解析和软解析）。软解析是对共享池中已有对象的检查，是为了验证底层对象的权限尚未改变。 硬解析与总数的比率应小于 20%。
parse time cpu (V$SYSSTAT 1/sid) (absolute)	用于解析 （硬解析和软解析）的总 CPU 时间，单位为 1/100 秒。
parse time elapsed (V$SYSSTAT 1/sid) (absolute)	用于解析的总已用时间，单位为 1/100 秒。该统计数字减去 **CPU 解析时间**可确定用于解析资源的总计等待时间。

计数器	描述
CPU used by this session (V$SYSSTAT 1/sid) (absolute)	会话使用的 CPU 总时间 （单位为 1/100 秒），从用户调用开始时到结束时。如果用户调用在 10 毫秒内完成，对于该统计信息的用途而言，用户调用的开始和结束时间视为相同，添加 0 毫秒。
bytes sent via SQL*Net to client (V$SYSSTAT 1/sid) (absolute)	从前台进程发送到客户端的字节总数。提供通过网络传输的数据量的一般指示。
bytes received via SQL*Net from client (V$SYSSTAT 1/sid) (absolute)	通过 Oracle Net Services 从客户端接收的字节总数。提供通过网络传输的数据量的一般指示。
logons current (V$SYSSTAT 1/sid) (absolute)	当前登录的总数。仅在 V$SYSSTAT 中有用。

除上面提到的计数器以外，建议监控相关的表空间使用情况。其中任何一个有小于 2% 的情况时，应增加表空间的大小。

优化和调整

遇到性能问题时，优化和调整是减轻问题所必需的。多数情况下，应用程序代码优化是必需的，但有时修复调整不佳的环境会大大改善性能。

此部分列出几个可能的调整做法。有些面向 Oracle 数据库，而其他一些是所有数据库服务器通用的，剩下的适用于任何服务器。有许多其他的调整做法可能对您的应用程序更有效。

调整需要漫长的测试和解析的迭代过程。任何配置更改都需要仔细验证。应用下面提到的任何做法之前，应通过了解您的服务器生成的参数和工作负载，首先验证配置与特定应用程序的关联。

- 确保 Oracle Cost Based Optimizer 正在运行。
- 定期收集优化器统计信息。
- 调整 SQL 语句：
  - 标识有问题的 SQL 语句 （即执行漫长的 SQL 语句）
  - 检查 Oracle 优化器统计信息 （确保基于成本的优化器正在运行，统计信息是最新的）
  - 检查执行计划
  - 重构 SQL 语句 （如有必要）
  - 重构 SQL 索引 （如有必要）
  - 以后持续维护执行计划
- 在 SQL 语句中使用绑定变量。这将减少共享池中存储的光标数量。
- 小心使用索引。不是每个列都应当编制索引，仅那些经常用查询访问的列需要。
- 一旦需要，就通过使用优化提示来辅助 SQL 优化器。这应当在解析 SQL 语句性能之后完成。

- 调整内存结构大小。共享池、缓冲区缓存及其他内存结构的大小对数据库的性能来说很关键。
  - 根据应用程序运行典型工作负载
  - 监控等待数、缓冲区点击率、系统切换和分页等等
  - 下表包括了所有参数中应调整的最重要参数：

参数	描述
db_cache_size	确定 SGA 中缓冲区缓存的大小。
db_keep_cache_size	这是加载对象时它们始终出现的位置。此缓存适用于以下对象：非常频繁地被访问且必须保留在内存中，例如很小的频繁使用的查询表。此缓存是参数 DB_CACHE_SIZE 定义的默认缓存的子集。对于任何数据库都必须设置 DB_CACHE_SIZE。
shared_pool_size	确定共享池的大小。
pga_aggregate_target	指定对附加到实例的所有服务器进程可用的目标聚合 PGA 内存。
log_buffer	确定重做日志缓冲区的大小。
query_rewrite_enabled	确定执行 SQL 语句之前 Oracle 是否改写它。
cursor_sharing	确定哪类 SQL 语句可以共享相同光标。

参数	描述
db_file_multiblock_read_count	用于在完整表扫描期间最小化 I/O 的参数之一。指定在顺序扫描期间一个 I/O 操作中读取的最大块数。
hash_multiblock_io_count	指定一个散列连接在一次 I/O 中读写多少顺序块。

- 要避免 I/O 操作，应把目标定为较高的缓冲区缓存点击率。在 OLTP 环境中应高于 80。 99 是最佳值。
- 字典缓存点击率应在 90% 左右。**dc_table_grants**、**d_user_grants** 和 **dc_users** 的条目在 **MISS RATE %** 列中都应低于 5%。
- 监控排序是指内存中的排序，相对磁盘中的排序而言。磁盘和内存之间的比率应当小于 10。
- 尽可能减少数据库争用。检查锁定和锁存数，尽量消除。
- 对于复杂的大规模数据存储工作负载，请使用 HP Oracle Database 计算机。

14

MS SQL Server 监控

此章节描述 Microsoft SQL Server 监控的最佳实践。

此章节包括：

概述

Microsoft SQL Server 是使用最广的数据库系统之一。它已从处理小部门任务成长为服务于地球上最大的数据库。 Microsoft SQL Server 现在不再仅仅是 “数据库”，而是完整的数据体系结构解决方案，能满足任何组织的数据存储和操纵需要。组织可以使用此解决方案存储和管理很多类型的数据，包括 XML、电子邮件、时间 / 日历、文件、文档、地理空间等，并提供与数据交互的丰富服务集：搜索、查询、数据分析、报告、数据集成和稳健的同步。开发者可以用各种基于 Microsoft 或第三方供应商的技术将从服务器访问 SQL Server 的应用程序写入台式机或移动设备。

SQL Server 有很多版本，有助于满足任何组织的需要。从 Express 和 Compact 到 Workgroup，再到 Standard 和 Enterprise，每个版本都提供一组针对特定需要的功能，同时为开发者和最终用户保持相同的功能级别。

过去人们说， SQL Server 开箱即装即用，运行良好，性能从来不是问题。但是，更廉价硬件的出现和数据的骤增正使得更多用户不满于 SQL Server 的开箱即装即用性能的限制。性能工程师的职责就是通过使用各种监控技术找出这些问题。在 SQL Server 的世界中，对 Enterprise 和 Standard 版本感兴趣的最多。

体系结构

安装 SQL Server 后，它的几乎每个组件的性能行为都有特定计数器指示，后者已添加到常规的 Windows 对象和计数器。但是，您通常从监控 CPU 利用率、磁盘活动、内存管理和网络带宽（参见第 3 章“Windows 监控”）之类的 Windows 系统资源开始。

监控这些资源的原因是它们代表了服务器的主要硬件组件，且每个组件都涉及用户请求的满足。这些组件的实时性能与总体察觉到的应用程序性能直接相关。因此，这四个方面中有一个或多个出问题时，很可能导致用户抱怨。SQL Server 严重依赖 CPU 性能、可用内存和磁盘吞吐量，而客户端性能则严重依赖网络性能。任何处理器如果 90% 或更多的时间持续忙碌，都会导致工作请求排队，性能很可能受影响。

此外， SQL Server 可能很耗内存，如果物理内存耗尽，性能可能真的会受很大影响，此时 Windows 通常被迫使用页面文件。磁盘由于其机械性质，几乎肯定是最慢的组件。 SQL Server 经常需要从磁盘检索数据，这意味着磁盘 I/O 的任何延迟都将影响总体性能。最后，您的数据库可能表现很完美，但如果网络有延迟或者数据包丢失太多而导致重发，您服务器高速的性能在最终用户看来也就不存在了。

相关 Windows 计数器

监控安装了 SQL Server 的计算机的系统资源时，有一些最重要的计数器要跟踪，包括额外的建议：

- **CPU。**添加新的物理处理器不是一件容易完成的任务，因此，确保所有 CPU 单元都同等分担负载很重要。监视以下计数器：

  注：有关这些计数器的完整细节，请参见 “处理器 —— 最重要的计数器”（第 48 页）。

  - **处理器时间百分比。**度量各处理器时间，确保 CPU 之间负载平衡。
  - **处理器队列长度。**如果此计数器经常超过建议的最大值，但 CPU 利用率并没有相应地变为很高 （这经常发生），则考虑降低 SQL Server **最大工作程序线程数**的配置设置。这样做可强制启动线程入池或更充分地利用它。

- **上下文切换数 / 秒。**有两种途径可以降低此值：
  - **关联掩码。**在较重的负载下，指定哪个处理器运行哪个线程可通过减少处理器缓存需要重新加载的次数来改进性能。有时从 SQL Server 排除某些处理器有助于改进操作系统请求的处理。
  - **轻型入池。**使用此 SQL Server 选项时，数据库转到基于光纤的模型，而不是默认的基于线程的模型。光纤是由数据库服务器而不是操作系统安排的，因此 CPU 负载较低。
- **内存。**SQL Server 动态管理其内存，从操作系统请求或释放它。确保选择了合适的动态选项，且数据库可用的最大内存接近物理内存的最高水平。

  监视以下计数器：

  注：有关这些计数器的完整细节，请参见 “内存 —— 最重要的计数器”（第 55 页）。

  - **可用字节数**
  - **页数 / 秒。**表示访问超出 SQL Server 分配范围的磁盘 I/O 和 / 或内存的次数。此值理想情况下应接近 0，可能有备份和还原形成的峰值。
  - **页故障数 / 秒**

- **磁盘。**数据库可能拥有所有应用程序层的多数 I/O 密集型操作，因此监控磁盘活动很关键。

  监视以下计数器：

  注：有关这些计数器的完整细节，请参见 “I/O —— 最重要计数器”（第 66 页）。

  - **磁盘时间百分比。**花在读 / 写功能上的时间百分比。对单个磁盘容量，您监控物理磁盘计数器；对跨多个磁盘的容量，您监控逻辑磁盘计数器。如果此值超过 55%，就表明有 I/O 瓶颈。在这种情况下，还可能要深入查看**磁盘读取时间百分比**和**磁盘写入时间百分比**计数器。它们的和就是该计数器的值。可能的调整可包括添加更多更快的磁盘、为磁盘 Controller 获取更大缓存、进行碎片整理以及重新配置 RAID 设备。
  - **平均值 磁盘队列。**显示特定磁盘的实际队列长度，尽管此计数器在存储区域网络 (SAN) 时代有点随意。对单个磁盘容量，您监控物理磁盘计数器；对跨多个磁盘的容量，您监控逻辑磁盘计数器。不要添加 **_Total** 计数器，因为这可以导致将结果汇总而掩盖了问题，使您对磁盘性能做出错误假定。

  提示：将不同磁盘上的 SQL Server 数据和日志文件分开是很好的做法，因为它们的 I/O 模式不同。还建议您将系统和用户数据库分别放到不同磁盘上。

- **网络。**某些应用程序设计为在有大量数据通过网络发送时将非常“随意”。监控以下计数器：

  注：有关完整详细信息，请参见 “网络 —— 最重要的计数器”（第 73 页）。

  - **字节总数/秒。**与更具体的接收字节数/秒和发送字节数/秒计数器一起，显示实际的网卡吞吐量。可能的调整可包括添加更多更快的网卡、使用该卡的全双工选项。重新配置数据库设置以删除所有不需要的协议，在服务器和客户端上保留 TCP/IP 为主要协议。

最重要的 SQL Server 计数器

SQL Server 性能体系结构遵循在 Windows 操作系统和 .NET 框架中实现的 Microsoft 途径。同样，它围绕着对象、实例和计数器组织（参见（第 46 页）上有关 Windows 体系结构的详细信息）。对象是任何 SQL Server 资源，如 SQL Server 锁或 Windows XP 进程。每个对象都包含确定要监控的对象的各方面的一个或多个计数器。如果计算机上存在给定类型的多个资源，则某些对象就有若干实例。默认实例的计数器以以下格式显示：**SQLServer:< 对象名称 >**。已命名实例的计数器以以下格式显示：**MSSQL$< 实例名称 >:< 计数器名称 >** 或 **SQLAgent$< 实例名称 >:< 计数器名称 >**。

有不少 SQL 性能对象，包括：

- SQLServer 引擎自身的 20 个对象
- Service Broker 的三个对象
- SQLAgent 的四个对象
- SQL Replication 的五个对象

下表列出了数据库引擎计数器：

	计数器	描述
CPU	SQL 编译数 / 秒	表示每秒发生的编译次数
	SQL 重新编译数 / 秒	表示每秒发生的重新编译次数
	批处理请求数 / 秒	表示每秒接收的 Transact-SQL 命令批处理数
内存	总页数	表示缓冲区池中的页数
	目标页数	表示缓冲池中的理想页数
	总计服务器内存 (KB)	表示 SQL Server 当前使用的内存量 (KB)
	目标服务器内存 (KB)	表示 SQL Server 有效运行所需要的内存量 (KB)。
	缓冲区缓存点击率	表示在内存中发现的页面的百分比
	页面寿命预期	表示未被引用的页面在缓冲区池中停留的秒数
	被占用页面	表示用于其他服务器用途 （包括过程缓存）的页数
	缓存点击率	表示缓存点击数与查询数之间的比率
	内存授予待定	表示等待工作区内存授予的进程总数
	检查点页数 / 秒	表示每秒由检查点或其他需要刷新所有 “脏页面” 的操作刷新的页数
	懒写入数 / 秒	表示由缓冲区管理器的懒写入程序每秒写入的缓冲区数

	计数器	描述
搜索	**完整扫描数 / 秒**	表示每秒未受限制的完整扫描数
	页拆分数 / 秒	表示由于溢出索引页面而发生的每秒页面拆分数
	临时表创建速率	表示每秒创建的临时表 / 表变量数
锁	**平均等待时间 (ms)**	表示每个导致等待的锁定请求的平均等待时间 （毫秒）

CPU 相关的计数器

如果 **sqlserver.exe** 利用了大部分 CPU 功能，这可能表明 SQL Server 内部有问题。除在“相关 Windows 计数器”（第 232 页）中说明的 Windows 计数器外，这些问题还可用以下计数器揭示：

SQL 编译数 / 秒

正式名称	SQLServer:SQL Statistics\SQL 编译数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）
描述	每秒发生 SQL Server 编译的次数。
使用备注	CPU 利用率过高的常见原因，这可能由架构问题、内存过低的状况以及查询执行计划的编译和重新编译引起。编译后计划应当保留在内存中 —— 除非有可能内存压力过大，导致从缓存丢弃计划。
性能	在稳定的条件下，您可以期望看到至少 90% 的计划被重用。
阈值	超过 10% 时警告
相关度量	➤ SQL 重新编译数 / 秒

SQL 重新编译数 / 秒

正式名称	SQLServer:SQL Statistics\SQL 重新编译数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）
描述	每秒发生 SQL Server 重编译的次数。
使用备注	如果只使用临时 T-SQL 或者查询未正确参数化，则 SQL Server 可能不会重用任何计划，或导致每个查询都发生计划编译。
性能	重新编译可能导致死锁和与任何锁定类型都不兼容的编译锁定。
阈值	超过 SQL 编译数 / 秒的 10% 时发出警告
相关度量	➤ SQL 编译数 / 秒

批处理请求数 / 秒

正式名称	SQLServer:SQL Statistics\ 批处理请求数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）
描述	每秒接收的 Transact-SQL 命令批处理数。
使用备注	显示 SQL Server 的 CPU 繁忙程度。
性能	如果此计数器超过阈值，这可能意味着您可能很快将遇到 CPU 瓶颈，即使尚未遇到。当然，这是个相对数字，取决于硬件功能。
阈值	超过 1000 时警告
相关度量	N/A

注：有些性能工程师监控 *SQLServer:Databases\ 事务 / 秒: _Total* 计数器，它只度量事务内所采取的活动 —— 而非像*批处理请求数 / 秒*计数器那样度量所有活动。

与内存相关的计数器

SQL Server 的性能和稳定性完全依赖于有足够的可用内存。内存不足通常导致 Windows 从分页文件提供虚拟地址空间，这通常对性能有立即的、非常显著的影响。

除“相关 Windows 计数器”（第 232 页）中说明的 Windows 计数器以外，使用以下计数器监控与内存相关的问题：

总页数

正式名称	SQLServer:Buffer Manager\ 总页数
计数器类型	瞬时 （每个度量时段采样一次）。
描述	缓冲区池中的页数 （包括数据库、可用和被占用页数）。
使用备注	显示 SQL Server 从 Windows 操作系统获取的总页数。
性能	表示计算机上运行并从 SQL Server 取得物理内存的其他过程。
阈值	参见第 240 页上的提示
相关度量	➤ 目标页数

目标页数

正式名称	SQLServer:Buffer Manager\ 目标页数
计数器类型	瞬时 （每个度量时段采样一次）。
描述	缓冲区池中的理想页数
使用备注	显示 SQL Server 获取并用来处理请求的总页数。
性能	N/A
阈值	参见如下提示
相关度量	➤ 总页数

提示：如果目标页数和总页数的值相同，则 SQL Server 有足够内存。如果目标数大于总数，则通常是由于另一个 Windows 进程阻止了 SQL Server 获取运行所需的任意数量内存。

总计服务器内存 (KB)

正式名称	SQLServer:Memory Manager\ 总计服务器内存 (KB)
计数器类型	瞬时 （每个度量时段采样一次）。
描述	SQL Server 当前使用的内存量，以千字节表示。
使用备注	显示 SQL Server 从 Windows 操作系统获取的物理内存总数。
性能	应当小于计算机上的内存总数。
阈值	参见第 241 页上的提示
相关度量	➤ 目标服务器内存 (KB)

目标服务器内存 (KB)

正式名称	SQLServer:Memory Manager\ 目标服务器内存 (KB)
计数器类型	瞬时 （每个度量时段采样一次）。
描述	SQL Server 有效运行需要多少内存
使用备注	显示 SQL Server 获取并用来处理请求的内存总量。
性能	
阈值	参见如下提示
相关度量	➤ 总计服务器内存 (KB)

提示：如果总计服务器内存值 (KB) 小于目标服务器内存值 (KB)，则 SQL Server 没有有效运行需要的足够内存。考虑添加更多物理内存。

缓冲区缓存点击率

正式名称	SQLServer:Buffer Manager\ 缓冲区缓存点击率
计数器类型	间隔 （繁忙百分比）
描述	在内存中发现的页面的百分比。
使用备注	该比率是最近数千次页面访问过程中的缓存查询总数除以缓存点击总数。
性能	如果缓冲区中未发现数据页，则 SQL Server 必须将它们从磁盘读取到缓冲区中。由于磁盘延迟和搜寻时间，这过程通常很慢。 因此，如果将缓冲区池配置为此计数器值的至少 98%，而性能仍然不足，可考虑添加物理内存。
阈值	该值越高越好。最好在 90% 标记附近。
相关度量	N/A

页面寿命预期

正式名称	SQLServer:Buffer Manager\ 页面寿命预期
计数器类型	平均值
描述	未被引用的页面在缓冲区池中停留的秒数。
使用备注	页面寿命预期越长，从内存的角度看服务器就状态越好。
性能	表明服务器上内存不足。
阈值	如果小于 300 秒，就有问题了。
相关度量	N/A

被占用页面

正式名称	SQLServer:Buffer Manager\ 被占用页面
计数器类型	瞬时 （每个度量时段采样一次）。
描述	用于其他服务器用途 （包括过程缓存）的页数
使用备注	被占用页面就是内存中被 SQL Server 计算机上的其他进程占用的那些页面。
性能	被占用页面数高表示服务器内存不足。
阈值	N/A
相关度量	➤ 总页数

缓存点击率

正式名称	SQLServer:Plan Cache\ 缓存点击率
计数器类型	间隔 （繁忙百分比）
描述	缓存点击数与查询数之间的比率。
使用备注	缓存中发现记录的时间百分比。 **注：**在 SQL Server 2000 中，此计数器在缓存管理器对象下。
性能	此计数器是 SQL Server 中缓存机制的良好指示器。
阈值	应在 99% 左右。值为 90% 时应生成警告。
相关度量	N/A

内存授予待定

正式名称	SQLServer:Memory Manager\ 内存授予待定
计数器类型	瞬时 （每个度量时段采样一次）。
描述	表示等待工作区内存授予的进程总数。
使用备注	这实际上是等待授予内存的进程的队列。
性能	如果有任何进程排队等待内存，则应预期到性能降低。运行良好的服务器的理想状况是没有未决的内存授予。
阈值	如果不是 0，就有问题了。
相关度量	N/A

检查点页数 / 秒

正式名称	SQLServer:Buffer Manager\ 检查点页数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）
描述	表示每秒由检查点或其他需要刷新所有 “脏页面” 的操作刷新的页数
使用备注	检查点操作由 SQL Server 执行，并需要将所有脏页面写入磁盘。
性能	检查点进程在磁盘 I/O 方面开销很大。服务器内存不足时，检查点进程会比平时发生更频繁，因为 SQL Server 尝试在缓冲区池中创建空间。 表明服务器上内存不足。
阈值	如果一段时间内观察到值连续偏高，就有问题了。
相关度量	N/A

懒写入数 / 秒

正式名称	SQLServer:Buffer Manager\ 懒写入数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）
描述	表示由缓冲区管理器的懒写入程序每秒写入的缓冲区数。
使用备注	懒写入程序是一个系统进程，它从“脏”的旧缓冲区“冲出”批处理，使之对用户进程可用。此计数器记录每秒 SQL Server 将脏页面从（内存中的）缓冲区池重新安置到磁盘的次数。
性能	磁盘 I/O 昂贵，您应尝试为 SQL Server 提供足够的缓冲区池空间，使懒写入数尽量接近 0。 表明服务器上内存不足。
阈值	如果不是 0，就有问题了。如果超过 20，则需要增加缓冲区池。
相关度量	N/A

与磁盘相关的计数器

将数据移入、移出磁盘，几乎始终是 SQL Server 需要承担的最耗时最昂贵的操作。 SQL Server 使用内置机制避免用户在内存和磁盘之间传输数据时必须等待，因为此过程中的任何轻微延迟都很可能影响察觉到的服务器性能。在 SQL Server 中，本质上有两套机制：具有预加载数据的缓冲区缓存，和加载有详细说明检索数据最有效方式的优化方案的方案缓存。如果有磁盘性能问题，它会引导您检查存储子系统的设计和实现。

除 “相关 Windows 计数器”（第 232 页）中说明的 Windows 计数器以外，使用以下计数器监控与磁盘相关的问题：

完整扫描数 / 秒

正式名称	SQLServer:Access Methods\ 完整扫描数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）
描述	表示每秒未受限制的完整扫描数。
使用备注	尽管表扫描事实上是存在的，并且有时比索引搜索更快，但通常表扫描还是少点好。此计数器是针对整个服务器而言，不只是单个数据库。
性能	定期执行的表扫描可能是由于 SQL Server 的内部作业。但是，此计数器值中的随机峰值表示索引不足或丢失。
阈值	为 1 时警告。为 2 或更大值表示错误。
相关度量	N/A

页拆分数 / 秒

正式名称	SQLServer:Access Methods\ 页拆分数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）
描述	表示由于溢出索引页面而发生的每秒页面拆分数。
使用备注	页面拆分是 I/O 密集型操作，发生在 8 KB 数据页面中的空间不足以使插入或更新操作完成时。在此环境中，将添加新页面，并在插入或更新发生之前在两个页面之间共享原始数据。
性能	偶尔出现页面拆分是正常的，但过多的页面拆分可导致过多的磁盘 I/O 活动，并导致性能下降。这些可以通过正确的索引维护和良好的填充因素选择来避免。
阈值	超过 100 时警告。
相关度量	N/A

提示：SQL Server 默认情况下允许自动增长并在必要时执行数据和文件的增加。虽然这可能很方便，仍建议在企业系统上手动调整设置。

临时表创建速率

正式名称	SQLServer:General Statistics\ 临时表创建速率
计数器类型	间隔差异计数器 （比率 / 秒）
描述	表示每秒创建的临时表 / 表变量数。
使用备注	SQL Server 用 **tempdb** 作为加入、排序和计算操作及版本存储的保存区。在大量使用 **tempdb** 的工作负载下，其响应可直接影响用户体验。
性能	**Tempdb** 是共享的全局资源。这意味着如果一个数据库或应用程序严重依赖于它，相同实例中的其他数据库可能遇到无法控制的性能问题。
阈值	N/A
相关度量	N/A

提示： 充分调整 **tempdb** 的大小，确保不需要自动增长。

与锁定相关的计数器

锁定对于并发而言是必需的。 SQL Server 会自动处理锁定。尽管锁定代表特定数据库或整个 SQL Server 的内部行为，与操作系统资源无关，但它们对响应时间仍有重要影响。锁定是长时间运行的事务导致最终用户申诉的一个主要原因。

在大多数情况中， SQL Server 自动解决锁定问题。但是，有两类容易出问题的锁定如果经常出现而需要处理，则是**阻止锁定**和**死锁**：

- **阻止锁定。**一个进程被阻止而不能锁定某资源，因为另一个进程已锁定它。
- **死锁。**两个进程各自保持对方继续运行所需的锁定时。如果不予理睬，它们会无限期地互相等待。

平均等待时间 (ms)

正式名称	SQL Server:Locks\ 平均等待时间 (ms)
计数器类型	平均值
描述	表示每个导致等待的锁定请求的平均等待时间 （毫秒）。
使用备注	如果对象锁定是响应速度慢的原因，则显示。可以使用此计数器度量多种锁定的平均等待时间，包括数据库、范围、键、页面、 RID 和表。
性能	对每类锁定观察该计数器随时间的表现，找到每类锁定的平均值。然后用这些平均值作为参考点。
阈值	N/A
相关度量	N/A

提示：如果可以识别导致事务延迟的一类或多类锁定，那就应当进一步调查，看看是否可以识别出是哪个特定事务导致锁定。用 HP Diagnostics 软件可捕获有问题的语句。

第 VII 部分

虚拟化技术

15

Microsoft 虚拟化监控

此章节描述 Microsoft 虚拟化服务器 Hyper-V 监控的最佳实践。

此章节包括：

概述

Microsoft 是一个很知名的平台，提供从数据中心到桌面的全面虚拟化产品集，允许从一个平台方便地管理物理和虚拟资产。

Microsoft 的虚拟化愿景和战略中心是 Microsoft Hyper-V，一项包含在 Microsoft Windows Server 2008 x64 版中的硬件辅助虚拟化新技术。

Hyper-V 是基于管理程序的虚拟化平台，它是在硬件上直接运行的软件的瘦层。它使多个操作系统能够在分区内并发运行，并通过实施关键系统资源的访问策略（如内存和处理器）确保严格隔离分区。

Microsoft 借助于 Hyper-V 技术提供的基于管理程序的虚拟化平台，可减少成本、提高硬件利用率、优化基础结构和提高服务器可用性，从而实现了灵活性。

Hyper-V 使虚拟机能够利用增强的安全，包括硬件级别安全功能。

Hyper-V 是一项可靠而高度可扩展的技术，支持以前需要在物理硬盘上运行才能达到企业需要的性能级别的虚拟化工作负载。

Hyper-V 功能包括：

- **通过卷影复制服务的实时备份**。任何运行卷影复制服务 (VSS) 的虚拟机 —— 能够感知来宾 Windows 操作系统（Windows Server 2003 及更高版本）—— 都可以在最短的宕机时间内以实时状态进行备份。
- **使用故障切换群集的高可用性**。 Hyper-V 支持使用 Windows 故障切换群集实现可以管理未计划和已计划的宕机时间的高可用性战略
- **快速迁移**。Hyper-V 支持跨群集节点移动虚拟机而不会丢失数据且服务中断时间极短的快速迁移。要完成此任务，需要将虚拟机置于已保存状态，将活动内存和处理器状态捕获到磁盘，并且将存储器资源所有权转交给群集中的另一个节点。在新节点上，会重新加载虚拟机的活动内存和处理器状态并恢复处理。
- **集成服务**。Hyper-V 集成服务 (IS) 为父和子分区之间需要安全接口的五个唯一组件提供支持。这些功能包括：
  - 时间同步
  - 心跳信号
  - 关机
  - 键 / 值对交换
  - 卷影复制服务 (VSS)

- **虚拟机导入和导出**。 Hyper-V 中的导入和导出功能应该在 Hyper-V 服务器之间移动和复制虚拟机。
- **虚拟硬盘管理**。Hyper-V 提供若干选项（压缩、转换、扩展、合并和重新连接）来管理虚拟硬盘 (VHD)，这些选项可通过 Hyper-V 管理器控制台进行访问。
- **虚拟机快照**。Hyper-V 允许您在任何特定时间点捕获虚拟机的配置和状态的快照，并允许您在几秒内重新加载任何现有快照。
- **虚拟机连接**。虚拟机连接 (VMC) 是 Hyper-V 附带的远程管理工具。VMC 使用 Windows 远程桌面协议允许远程访问虚拟机上运行的来宾操作系统。

体系结构

Microsoft Hyper-V 分两种形式：作为 Windows Server 2008 中的角色或作为名为 Microsoft Hyper-V Server 2008 的独立产品。这两个类型的体系结构非常相似。

Hyper-V 由一组组件构成，包括 Windows 管理程序（环 -1）、内核模式组件（环 0）和用户模式组件（环 3）。

有 3 个定义指令权限级别的处理器环，环 0 的权限最高，环 3 的权限最低。

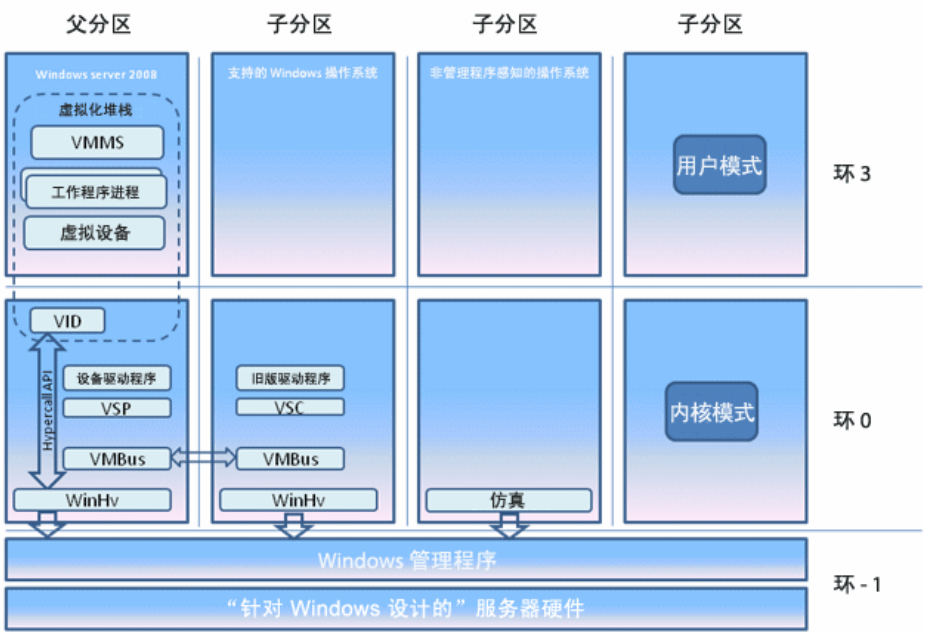
- 环 0：运行操作系统的内核。
- 环 3：运行用户应用程序。
- 环 -1：运行硬件虚拟化扩展。

  此环允许 Windows 管理程序在其自己的环境中并以高于 Windows 内核的权限级别运行，同时允许任何来宾操作系统内核继续在处理器环 0 运行，用户应用程序继续在处理器环 3 运行。

当在 Windows 管理程序之上运行时，有一个父分区，一个或多个子分区。

- 父分区是运行虚拟化堆栈的控制分区。父分区还是拥有硬件设备和管理子分区资源的分区。
- 子分区是由父分区创建的所有分区。来宾操作系统及其应用程序在子分区中运行。

分区使用 hypercall 与管理程序层通信，hypercall 是分区操作系统用于利用系统管理程序提供的优化的 API。



下面是与每个分区和层相关的 Hyper-V 体系结构的主要组件：管理程序（环 -1）、用户模式 （环 3）、内核模式 （环 0）：

管理程序

Windows 管理程序是位于物理硬件和一个或多个操作系统之间的软件接口。Windows 管理程序控制对核心硬件的访问，并定义名为分区的隔离执行环境。

Windows 管理程序的主要任务是保证分区之间隔离，对硬件访问实施策略限制并监控分区。

父分区

父分区是管理程序启动时在系统上创建的第一个分区。父分区为 Windows Server 2008 操作系统而创建，其用途为：

- 父分区用于创建和管理子分区，包括提供远程管理接口的 WMI 提供程序。
- 父分区管理和分配硬件设备，但处理器计划和物理内存分配除外，这些任务由管理程序处理。
- 父分区的硬件资源供子分区共享或分配给子分区使用。
- 父分区处理任何硬件故障事件的电源管理、即插即用操作和日志记录。

父分区中受管的虚拟化组件统称为虚拟化堆栈。虚拟化堆栈在父分区中运行，并可直接访问基础主机计算机的硬件。除虚拟化堆栈组件以外，下面还对一些其他组件进行详细说明。

虚拟化堆栈由以下组件构成：

- 虚拟机管理服务 —— VMMS
- 虚拟机工作程序进程
- 虚拟设备

- 虚拟化基础结构驱动程序 (VID)
- Windows 管理程序接口库

父分区的其他组件包括：

- 虚拟化服务提供程序 (VSP)
- 虚拟机总线 (VMBus)

下面是父分区主要组件的详细描述：

虚拟机管理服务 —— VMMS

虚拟机管理服务 (VMMS) 是共同管理虚拟机的一组组件。

VMMS 在用户模式和内核模式中作为系统服务 (VMMS.exe) 实现，负责管理子分区中虚拟机的状态。

包括管理已停止或脱机的虚拟机，处理快照的创建和管理设备的添加或删除。当子分区中的虚拟机启动时，VMMS 衍生出新的虚拟机工作程序进程，用于执行该虚拟机的管理任务。

VMMS 还控制哪些操作可以在给定状态的虚拟机上执行。 VMMS 管理以下虚拟机状态：正在启动、活动、非活动、正在拍摄 / 应用 / 删除快照、正在合并磁盘。

联机虚拟机操作（如暂停、保存和关闭）不由 VMMS 管理。它们由 VMMS 为被管虚拟机衍生出来的虚拟机工作程序进程管理。

虚拟机工作程序进程

虚拟机工作程序进程 (vmwp.exe) 是用户模式进程，提供从父分区中的 Windows Server 2008 实例到子分区中的来宾操作系统的虚拟机管理服务。

VMMS 为每个正在运行的虚拟机衍生出单独的 VM 工作程序进程，以隔离虚拟机。这样，如果一个 VM 工作程序进程失败，仅影响与该 VM 工作程序进程关联的虚拟机。

VM 工作程序进程管理其关联虚拟机的以下方面：

- 创建、配置和运行虚拟机
- 暂停和恢复虚拟机
- 保存和还原虚拟机
- 拍摄虚拟机的快照

此外，VM 工作程序进程包含虚拟主板 (VMB)，用于提供来宾内存、IRQ 生成、作为单独设备对虚拟机的内存映射和端口映射的 I/O。 VMB 还负责虚拟设备的管理。

虚拟设备

虚拟设备 (VDev) 是为子分区提供设备配置和控制的软件模块。

VDev 分为两种类型：

- 核心 VDev：

这些虚拟设备对现有硬件设备建模并可用于每个虚拟机。它们通常用于这样的情形中：兼容性很重要，以便 BIOS 或设备驱动程序等现有软件可以正确工作而不必修改。核心 VDev 可以是以下任何一种：

  - 模拟设备 —— 这些虚拟设备模拟特定的硬件设备。例如：BIOS、 DMA、PCI 总线、键盘 / 鼠标控制器等。

- 合成设备 —— 这些虚拟设备不对特定硬件设备建模。它们仅对支持集成服务的来宾操作系统可用。

- 插件 Vdev：

这些虚拟设备不对现有硬件设备建模，它们用于实例化、配置和管理在控制硬件的父分区中运行的虚拟化服务提供程序。

插件 VDev 通过 VMBus 支持父子分区之间直接通信。

虚拟化基础结构驱动程序 (VID)

虚拟化基础结构驱动程序 (Vid.sys) 是虚拟化堆栈的内核模式组件，提供针对所有子分区的分区管理服务、虚拟处理器管理服务和内存管理服务。Vid.sys 还使虚拟化堆栈的用户模式组件能够与管理程序进行通信。

Windows 管理程序接口库

Windows 管理程序接口库 (WinHv.sys) 是一个内核模式的动态链接库，在父分区中运行的 Windows Server 2008 实例中和在子分区中感知 Hyper-V 的来宾操作系统中进行加载。

WinHv.sys 抽象化 hypercall 实现详细信息，使操作系统的驱动程序能够使用标准 Windows 调用约定调用管理程序。

虚拟化服务提供程序 (VSP)

虚拟化服务提供程序 (VSP) 受管在父分区中，通过向子分区中运行的虚拟化服务客户端 (VSC) 提供 I/O 相关资源，提供了一种向子分区发布设备服务的方式。对于设备功能的客户端 / 服务器通信，VSP 是服务器端点，VSC 是客户端端点。VSP 和 VSC 之间的所有通信都通过 VMBus 来执行。

虚拟机总线 (VMBus)

虚拟机总线 (VMBus) 是父分区和子分区之间基于通道的逻辑通信机制。 VMBus 的用途是在虚拟化分区之间提供高速、高度优化的通信机制，而不是其他的由于模拟造成的较高开销而降低速度的技术。

子分区

子分区是由父分区创建的所有分区。

子分区是基于软件的物理硬件表示，也称为虚拟机。来宾操作系统及其应用程序在子分区中运行。

子分区不能直接访问服务器的实际物理硬件。它们只能看到向它们显示的虚拟硬件和虚拟设备。

Hyper-V 支持三种类型的子分区：

- 受管 Hyper-V 感知的 Windows 操作系统的子分区。
- 受管支持 Hyper-V 感知的 Windows 操作系统的子分区。
- 受管非 Hyper-V 感知的操作系统的子分区。

受管 Hyper-V 感知的 Windows 操作系统的子分区

运行 Hyper-V 感知的 Windows 操作系统的子分区包括以下内核模式虚拟化组件：

虚拟化服务客户端：

VSC 是驻留在子分区中的合成设备，通过 VMBus 通信使用父分区中的 VSP 提供的硬件资源。

当子分区中安装了使子分区能够使用合成设备的集成服务时， VSC 会自动对安装可用。

启蒙

对来宾操作系统代码的修改，以便在检测到正在作为管理程序环境中的来宾运行时更高效地运行。

Hyper-V 支持以下资源的启蒙：存储器、网络、图形和输入子系统。

受管支持 Hyper-V 感知的 / 非 Windows 操作系统的子分区

运行 Hyper-V 感知的非 Windows 操作系统的子分区使用第三方 VSC 通过 VMBus 与父分区中的 VSP 通信来访问硬件。向子分区提供这些 VSC 需要在分区中安装集成服务。

集成服务主要用于解决由于虚拟机的隔离环境造成的可用性问题。集成服务还提供允许子分区与其他分区和管理程序通信的组件。

集成服务还提供以下针对子分区的功能：

- **心跳信号**：用于验证子分区是否正在响应父分区的请求。
- **键 / 值对交换**：在子和父分区之间交换注册表键对。
- **时间同步**：与父分区同步子分区时间。
- **关机**：允许子分区响应父分区的关机请求。

Hyper-V 包括 Windows 操作系统的 x86 和 x64 版本的集成服务：

Win XP (SP3)、Win Vista (SP1)、Win Server 03 (SP2)、Win Server 08、Linux Enterprise Server 10。

受管非 Hyper-V 感知的操作系统的子分区。

运行非 Hyper-V 感知的操作系统的子分区无法安装集成服务。

这意味着这些来宾操作系统必须使用模拟的设备而不是合成设备，并会因使用此类模拟设备而导致性能下降。

监控工具

以下部分介绍 Microsoft 和 HP 提供的监控工具，用于监控基于 Hyper-V 服务器的虚拟化环境。

Microsoft 监控解决方案

Microsoft 主要提供两种监控解决方案：

- 可靠性和性能监控器
- System Center Operations Manager 2007

使用这些工具可以调查并获取有关物理主机、 Hyper-V 父分区和子分区的潜在运行状况问题的警告。

监控物理主机着重监控环境问题，如温度、电源和运行时间。

监控 Hyper-V 父分区着重监控逻辑处理器、系统内存使用情况、系统存储器性能、系统网络性能、 Windows 管理程序和父分区服务。

监控子分区涉及监控子分区中分配的虚拟硬件 （虚拟处理器、内存、存储器和网络适配器）以及运行的服务和应用程序。

可靠性和性能监控器

可靠性和性能监控器附在标准的 Windows Server 2008 完全安装中。它是基于 MMC 的应用程序，可以监控本地系统或远程系统。可靠性和性能监控器是两个工具合而为一。可靠性监控器提供有关系统稳定性的信息和影响可靠性的事件的信息，性能监控器提供系统组件、服务和应用程序的详细实时性能信息。

可靠性监控器跟踪事件的历史记录，比如，软件安装、应用程序故障、硬件故障、操作系统故障和许多其他杂项故障。以两种形式提供数据，即系统稳定性图表和系统稳定性报告。

系统稳定性图表显示过去 30 天的事件，并提供从 1（最不稳定）到 10（最稳定）的索引值。稳定性索引是派生自在滚动历史期间看到的指定故障次数的加权度量。

系统稳定性报告提供有关实际事件或故障、发生的活动、状态和发生日期的详细信息。

性能监控器可以通过两种模式运行：实时数据捕获和记录的数据捕获。

实时数据捕获允许您查看所选性能计数器的实时性能信息。性能计数器按操作系统、应用程序或服务定义。

计数器组织成组，一个计数器可以提供系统性能数字，也可以由多个实例组成。例如，如果查看名为 LogicalDisk 的性能计数器组，您会看到 23 个定义的计数器的列表。

性能监控器的数据记录功能允许您捕获可用性能计数器的历史视图而不是实时视图。在实时收集模式中，数据图会根据您收集数据的频率覆盖上一组数据。若要保留所收集数据的历史记录，必须使用数据记录模式。

System Center Operations Manager 2007

System Center Operations Manager 2007 是 Microsoft 企业级的硬件、操作系统、服务和应用程序监控解决方案。Operations Manager 2007 使用基于代理程序的数据收集机制从远程系统收集信息，并将数据存储在 SQL 数据库中以供分析。数据收集配置基于管理包的概念。管理包包含特定应用程序、操作系统或硬件的规则、监控器和任务。

规则定义如何从各种源（如 Perfmon、EventLog、SNMP 和日志文件）收集数据。然后将数据存储在 Operations Manager 数据库中以用于报告用途。监控器是定义监控对象的运行状况的状态计算机。监控器的状态可以是两者之一（绿色或红色），或三者之一（绿色、黄色或红色）。监控器的状态根据监控信息而变化。监控器可以定义阈值以监视规则收集的数据，然后基于阈值违反情况执行操作。

例如，监控器可以定义为查看虚拟机的网络吞吐量，如果它超过特定吞吐量值，则触发黄色状态（警告）并向 Operations Manager 2007 操作控制台发送警报。任务是用户从 Operations Manager 2007 操作控制台上启动的操作，该控制台通过 Operations Manager 代理程序运行于远程服务器上。预置任务在管理包中定义，您能够定义其他自定义任务。

监控安装有 System Center Operations Manager 2007 SP1 的 Hyper-V 基础结构时，是否能够维持健康的系统将取决于导入和利用的管理包。至少需要包括支持 Windows Server 2008 的最新 Windows Server Base Operating System 管理包。这将允许您监控操作系统、服务、存储器、网络、处理器和内存的可用性和性能。

监控 Hyper-V 服务器、虚拟机和 SCVMM 2008 服务器所需的工具已组合为一个管理包，即 System Center Virtual Machine 2008 管理包。该管理包允许您监控 Virtual Server 2005 R2、 Hyper-V 和 VMware ESX 服务器，并提供报告。

注：若要监控 VMWare ESX 服务器，它们必须是 SCVMM 2008 管理的主机。

SCVMM 2008 管理包监控存储器、内存、处理器、物理网络、虚拟网络以及虚拟机数等的 Hyper-V 服务器性能。 SCVMM 2008 管理包还监控虚拟处理器、虚拟硬盘和直通磁盘、虚拟机内存利用率和虚拟网络的虚拟机性能。这可以通过加载到 Hyper-V 服务器上的 Operations Manager 2007 代理程序来完成。如果虚拟机内还安装了 Operations Manager 2007 代理程序，且应用程序的关联管理包已导入到 Operations Manager 2007 中，则可以获取应用程序性能信息。

SCVMM 2008 管理包提供更新的监控、规则和报告：

- **VM 虚拟化报告**：提供有关虚拟机的利用率信息。此报告显示所选时间段内虚拟机处理器、内存和磁盘空间的平均、总计或最大使用率。
- **主机虚拟化报告**：显示每个主机上运行的虚拟机数。此报告显示所选时间段和主机组内主机处理器、内存和磁盘空间的平均、总计和最大利用率。
- **虚拟化候选对象报告**：有助于确定转换为虚拟机的合适候选物理计算机。此报告显示 CPU、内存和磁盘使用率及硬件配置 （包括处理器速度、处理器数和 RAM 总计）的一组已定义性能计数器的平均值。
- **主机利用率增长报告**：显示所选时间段内运行的主机资源和虚拟机数的增长百分比。
- **VM 分配**：提供可用于计算拒绝向成本中心支付的虚拟机费用的信息。

通过集成 System Center Operations Manager 2008 和 System Center Operations Manager 2007 SP1 而启用的高级功能是性能和资源优化 (PRO)。性能和资源优化是虚拟机管理器的功能，它利用来自 Operations Manager 2007 的性能信息帮助客户确保虚拟机基础结构正在以理想、高效的方式运行。通过扩展 System Center Operations Manager 2007 的监控能力， PRO 使管理员能够应对因虚拟化硬件、操作系统或应用程序故障而造成的低劣性能。

PRO 提供两个响应选项：

- 第一个选项涉及问题存在时发出警报并提供建议的缓解解决方案。管理员能够通过单击按钮来实施建议的缓解办法。缓解可能涉及将虚拟机从超出定义的处理器利用率阈值的 Hyper-V 服务器迁移到另一个 Hyper-V 服务器的内置操作。 PRO 支持用自定义操作进行扩展，例如，使用 “唤醒”唤醒预置备用 Hyper-V 服务器，这样您就能够动态扩大池以满足新需要。
- 第二个响应选项是系统在没有管理员干预的情况下自动实施建议的操作。

用于监控 Hyper-V 的 HP SiteScope

HP SiteScope 提供用于监控 Hyper-V 基础结构的综合工具。

SiteScope V11.0 提供的 Hyper-V 性能监控器可以监控基于 Hyper-V 的服务器的根和子分区。

在初始监控器创建期间，新监控器使用配置的连接 URL，访问软件并动态发现对象层次结构和可用的性能计数器。您可以选择这些性能计数器以确定 SiteScope 应该检索哪个度量来报告服务器状态。

相关的 Windows 计数器

监控安装了 Hyper-V 的计算机的系统资源时，需要跟踪 Windows 的最重要计数器：

- **CPU。**添加新的物理处理器不是一件容易完成的任务，因此，确保所有 CPU 单元都同等分担负载很重要。监视以下计数器：
  - **处理器时间百分比 /_Total。**
- **内存。** Hyper-V 服务器动态管理其内存，从操作系统请求或释放内存。确保选择了合适的动态选项，且数据库可用的最大内存接近物理内存的最高水平。

  监视以下计数器：
  - **可用字节。**
  - **页数 / 秒**

- **磁盘。**数据库可能拥有所有应用程序层的多数 I/O 密集型操作，因此监控磁盘活动很关键。

  监视以下计数器：

  - **当前磁盘队列长度。**
  - **磁盘字节数 / 秒。**
  - **磁盘传输次数 / 秒。**

- **网络。**一些应用程序设计为当有大量数据通过网络发送时非常 “健谈”。监视以下计数器：

  - **字节总数 / 秒。**
  - **卸载的连接数。**
  - **数据包数 / 秒。**
  - **出站错误数据包数。**
  - **接收错误数据包数。**

最重要的计数器

以下计数器表划分成 5 个类别：CPU、内存、网络、存储器和类属。

每个类别包括用于监控 Hyper-V 服务器及其虚拟机性能的最重要计数器的列表。

	计数器	描述
CPU	**来宾运行时间百分比**	指示处理器用于来宾代码的时间的百分比。
	管理程序运行时间百分比	指示处理器用于管理程序代码的时间的百分比。
	空闲时间百分比	指示处理器用于空闲状态的时间的百分比。
	运行时间百分比总计	指示处理器用于来宾代码和管理程序代码的时间的百分比。
	来宾运行时间百分比 (VPGRT)	指示虚拟处理器用于来宾代码的时间的百分比。
	管理程序运行时间百分比	指示虚拟处理器用于管理程序代码的时间的百分比。
	运行时间百分比总计 (VPTR)	指示虚拟处理器用于来宾代码和管理程序代码的时间的百分比。
	截取总计 / 秒	指示管理程序截取消息的速率
	处理器时间百分比 /_Total	指示处理器执行非空闲线程所用的已用时间百分比。

	计数器	描述
内存	**1G GPA 页数**	指示分区的 GPA 空间内存在的 1G 页数。
	2M GPA 页数	指示分区的 GPA 空间内存在的 2M 页数。
	存储的页数	指示存储到分区中的页数。
	虚拟处理器数	指示分区中存在的虚拟处理器数。
	分配的物理页数	指示分配的物理页数。
	远程物理页数	指示未从首选 NUMA 节点分配的物理页数。
	可用字节数	指示立即可分配给进程或供系统使用的物理内存量（字节）。
	页数 / 秒	指示从磁盘读取页或向磁盘写入页以解决硬页故障的速率。

	计数器	描述
I/O	**当前磁盘队列长度**	指示收集性能数据时磁盘上未解决的请求数。它还包括收集时正在处理的请求数。
	磁盘字节数 / 秒	指示写入或读取操作期间传输到磁盘或从磁盘读取字节的速率。
	磁盘传输次数 / 秒	指示磁盘上的读取和写入操作的速率。
	读取的字节数 / 秒	指示每秒从连接到 IDE Controller 的磁盘读取的字节数。
	写入的字节数 / 秒	指示每秒向连接到 IDE Controller 的磁盘写入的字节数。
	读取的扇区数 / 秒	指示每秒从连接到 IDE Controller 的磁盘读取的扇区数。
	写入的扇区数 / 秒	指示每秒向连接到 IDE Controller 的磁盘写入的扇区数。
	错误计数	指示此虚拟设备上已发生的错误总数。
	清除计数	指示此虚拟设备上已发生的清除操作的总数。
	读取的字节数 / 秒	指示此虚拟设备上每秒读取的字节总数。
	写入的字节数 / 秒	指示此虚拟设备上每秒写入的字节总数。
	读取计数	指示此虚拟设备上已发生的读取操作的总数。
	写入计数	指示此虚拟设备上已发生的写入操作的总数。

	计数器	描述
网络接口	**字节总数 / 秒**	指示通过网络适配器发送和接收字节（包括成帧字符）的速率。
	卸载的连接数	指示当前由具有 TCP 烟囱卸载功能的网络适配器处理的 TCP 连接数 （通过 IPv4 和 IPv6）。
	数据包数 / 秒	指示网络接口上发送和接收数据包的速率。
	出站错误数据包数	指示由于错误而无法传输的出站数据包数。
	接收错误数据包数	指示包含阻止入站数据包传递到更高层协议的错误的入站数据包数。
	字节数 / 秒	指示每秒遍历虚拟交换机的字节总数。
	数据包数 / 秒	指示每秒遍历虚拟交换机的数据包总数。
	丢弃的字节数	指示网络适配器上丢弃的字节数。
	发送的字节数 / 秒	指示每秒通过网络适配器发送的字节数。
	接收的字节数 / 秒	指示每秒通过网络适配器接收的字节数。
	字节数 / 秒	指示已遍历网络适配器的字节总数。
	数据包数 / 秒	指示网络适配器每秒接收的字节数。

	计数器	描述
概述	**运行状况正常**	指示系统运行状况稳定的虚拟机数。
	运行状况严重	指示系统运行状况处于严重状态的虚拟机数。
	逻辑处理器数	指示系统中存在的逻辑处理器数。
	分区数	指示系统中存在的分区 （虚拟机）数。
	总页数	指示管理程序中的引导程序数和存储的页数。
	虚拟处理器数	指示系统中存在的虚拟处理器数。
	监控的通知数	指示向管理程序注册的监控的通知数。

CPU 计数器

此部分描述提供处理器利用率相关信息的计数器：

来宾运行时间百分比

正式名称	Hyper-V 管理程序逻辑处理器 \ 来宾运行时间百分比
计数器类型	间隔 （繁忙百分比）
描述	处理器用于来宾代码的时间的百分比。
使用备注	这是来宾代码在 LP 上运行的时间的百分比，_Total 是在所有 LP 上的平均时间百分比。例如，如果有两个 LP，一个 VM 正在运行 CPU 测试，则可能会看到 LP(0) 的值是 95%，LP(1) 的值是 0%，_Total 的值是 47.5%。因此，可以看到 VM 正在 LP(0) 上运行。
性能	N/A
操作	N/A
阈值	N/A

管理程序运行时间百分比

正式名称	Hyper-V 管理程序逻辑处理器 \ 管理程序运行时间百分比
计数器类型	间隔 （繁忙百分比）
描述	处理器用于管理程序代码的时间的百分比。
使用备注	这是管理程序在 LP 上运行的时间的百分比，_Total 是在所有 LP 上的平均时间百分比。这类似于处理器计数器集中的内核运行时间百分比。
性能	N/A
操作	N/A
阈值	N/A

空闲时间百分比

正式名称	Hyper-V 管理程序逻辑处理器 \ 空闲时间百分比
计数器类型	间隔 （繁忙百分比）
描述	处理器用于空闲状态的时间的百分比。
使用备注	这是 LP 在等待工作的时间百分比，_Total 是在所有 LP 上的平均时间百分比。这类似于处理器计数器集中的内核运行时间百分比。
性能	N/A
操作	N/A
阈值	N/A

运行时间百分比总计

正式名称	Hyper-V 管理程序逻辑处理器 \ 运行时间百分比总计 (LPTR)
计数器类型	间隔 （繁忙百分比）
描述	处理器用于来宾代码和管理程序代码的时间的百分比。此计数器有时称为 LPTR。
使用备注	这正好是来宾运行时间百分比加上管理程序运行时间百分比的总和。此计数器可以稍稍超过 100% (<0.5%)。此问题与性能计数器的计算方法有关。如果采用当前时间，然后 value1，再然后是结束时间和 value2，这意味着 value2 可能会在读取结束时间和读取 value2 之间增加。您应该改为采用开始时间，然后是 value1，再然后是 value2 和结束时间。在这种情况下，数字始终会稍低于 100。
性能	LPTR 指示主机中逻辑处理器的繁忙程度。
操作	N/A
阈值	N/A

来宾运行时间百分比 (VPGRT)

正式名称	Hyper-V 管理程序根虚拟处理器 \ 来宾运行时间百分比 (VPGRT) **或** Hyper-V 管理程序虚拟处理器 \ 来宾运行时间百分比 (VPGRT)
计数器类型	间隔 （繁忙百分比）
描述	虚拟处理器用于来宾代码的时间的百分比。
使用备注	对于来宾 VM，这是来宾 VP 在 LP 上的非管理程序代码中运行的时间的百分比，_Total 是所有来宾 VP 的总计。对于根，这是根 VP 在 LP 上的非管理程序代码中运行的时间百分比， _Total 是所有根 VP 的总计。如果将来宾 VP 和根 VP 的 _Total 相加，值将等于逻辑处理器计数器集的来宾运行时间 _Total 百分比。
性能	VPGRT 指示来宾中的虚拟处理器的繁忙程度。 这些计数器具有很小的时钟偏差。
操作	N/A
阈值	N/A

管理程序运行时间百分比

正式名称	Hyper-V 管理程序根虚拟处理器 \ 管理程序运行时间百分比 **或** Hyper-V 管理程序虚拟处理器 \ 管理程序运行时间百分比
计数器类型	间隔 （繁忙百分比）
描述	虚拟处理器用于管理程序代码的时间的百分比。
使用备注	对于来宾 VM，这是来宾 VP 在 LP 上的管理程序代码中运行的时间的百分比， _Total 是所有来宾 VP 的总计。对于根，这是根 VP 在 LP 上的管理程序代码中运行的时间百分比， _Total 是所有根 VP 的总计。如果将来宾 VP 和根 VP 的 _Total 相加，值将等于逻辑处理器计数器集的管理程序运行时间 _Total 百分比。
性能	N/A
操作	N/A
阈值	“管理程序时间百分比”应该低于 25%。如果高于此值，可能指示未通过安装的集成服务运行。

运行时间百分比总计 (VPTR)

正式名称	Hyper-V 管理程序根虚拟处理器 \ 运行时间百分比总计 (VPTR) **或** Hyper-V 管理程序虚拟处理器 \ 运行时间百分比总计 (VPTR)
计数器类型	间隔 （繁忙百分比）
描述	虚拟处理器用于来宾代码和管理程序代码的时间的百分比。
使用备注	这正好是每个 VP 上来宾运行时间百分比加上管理程序运行时间百分比的总和。如果将根虚拟处理器和虚拟处理器计数器集中的运行时间百分比总计相加，值将等于所有逻辑处理器计数器的运行时间总计的总和。
性能	VPTR 显示主机中的虚拟处理器的繁忙程度。
操作	N/A
阈值	N/A

截取总计 / 秒

正式名称	Hyper-V 管理程序根虚拟处理器 \ 截取百分比总计 / 秒 **或** Hyper-V 管理程序虚拟处理器 \ 截取百分比总计 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	管理程序截取消息的速率。
使用备注	只要来宾 VP 需要退出时是在管理程序中为提供服务而正在运行的当前模式，就称为截取。一些常见的截取原因是解决来宾物理地址 (GPA) 到服务器物理地址 (SPA) 的转换、特权指令（如 hlt / cupid / in / out）以及 VP 计划时间片段的结束。
性能	N/A
操作	N/A
阈值	N/A

处理器时间百分比 /_Total

注：此计数器不特定于 Hyper-V。这是常规 Windows 资源计数器。请参见 “相关的 Windows 计数器”（第 268 页）。

内存计数器

此部分描述属于 Hyper-V 系统内存管理的计数器。它们提供有关内存消耗和内存池等的数据。

1G GPA 页数

正式名称	Hyper-V 管理程序分区 \1G GPA 页数 **或** Hyper-V 管理程序根分区 \1G GPA 页数
计数器类型	瞬时 （每个度量时段采样一次）。
描述	分区的 GPA 空间内存在的 1G 页数。
使用备注	VM 是否正在使用可提高总体 VM 性能的大页。
性能	N/A
操作	N/A
阈值	N/A

2M GPA 页数

正式名称	Hyper-V 管理程序分区 \2M GPA 页数 **或** Hyper-V 管理程序根分区 \2M GPA 页数
计数器类型	瞬时 （每个度量时段采样一次）。
描述	分区的 GPA 空间内存在的 2M 页数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

存储的页数

正式名称	Hyper-V 管理程序分区 \ 存储的页数 **或** Hyper-V 管理程序根分区 \ 存储的页数
计数器类型	间隔 （整个周期中采样）
描述	存储到分区中的页数。
使用备注	管理程序正在使用多少内存来管理 VM。
性能	N/A
操作	N/A
阈值	N/A

虚拟处理器数

正式名称	Hyper-V 管理程序分区 \ 虚拟处理器数 **或** Hyper-V 管理程序根分区 \ 虚拟处理器数
计数器类型	瞬时 （每个度量时段采样一次）。
描述	分区中存在的虚拟处理器数。
使用备注	使您能够知道 VM 配置为使用多少虚拟处理器。
性能	N/A
操作	N/A
阈值	N/A

分配的物理页数

正式名称	Hyper-V VM Vid 分区 \ 分配的物理页数
计数器类型	间隔 （整个周期中采样）
描述	分配的物理页数。
使用备注	管理 VM 所需的来宾页和 VID 页的总数
性能	N/A
操作	N/A
阈值	N/A

远程物理页数

正式名称	Hyper-V VM Vid 分区 \ 远程物理页数
计数器类型	间隔 （整个周期中采样）
描述	未从首选 NUMA 节点分配的物理页数。
使用备注	使您能够知道，在基于 NUMA 的系统上，VM 是否跨多个节点。确实要尽可能避免这种情况。
性能	N/A
操作	N/A
阈值	N/A

可用字节数

注：此计数器不特定于 Hyper-V。这是常规 Windows 资源计数器。请参见 “相关的 Windows 计数器”（第 268 页）。

页数 / 秒

注：此计数器不特定于 Hyper-V。这是常规 Windows 资源计数器。请参见 “相关的 Windows 计数器”（第 268 页）。

I/O 计数器

此部分描述属于 Hyper-V 存储器容量的计数器。

以下 I/O 计数器不特定于 Hyper-V。这些是 Windows 相关的常规计数器。有关详细信息，请参见 “相关的 Windows 计数器”（第 268 页）。

- 当前磁盘队列长度
- 磁盘字节数 / 秒
- 磁盘传输次数 / 秒

读取的字节数 / 秒

正式名称	Hyper-V 虚拟 IDE Controller\ 读取的字节数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	读取的字节数 / 秒是每秒从连接到 IDE Controller 的磁盘读取的字节数。
使用备注	虚拟 IDE 计数器显示在“Hyper-V 虚拟 IDE Controller”计数器集中。
性能	N/A
操作	N/A
阈值	N/A

写入的字节数 / 秒

正式名称	Hyper-V 虚拟 IDE Controller\ 写入的字节数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	写入的字节数 / 秒是每秒写入到连接到 IDE Controller 的磁盘的字节数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

读取的扇区数 / 秒

正式名称	Hyper-V 虚拟 IDE Controller\ 读取的扇区数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	读取的扇区数 / 秒是每秒从连接到 IDE Controller 的磁盘读取的扇区数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

写入的扇区数 / 秒

正式名称	Hyper-V 虚拟 IDE Controller\ 写入的扇区数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	写入的扇区数/秒是每秒写入到连接到 IDE Controller 的磁盘的扇区数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

错误计数

正式名称	Hyper-V 虚拟存储设备 \ 错误计数
计数器类型	间隔 （整个周期中采样）
描述	此计数器表示此虚拟设备上已发生的错误总数。
使用备注	如果已加载集成服务，则将在“Hyper-V 虚拟存储设备”计数器集中看到虚拟 IDE 和 SCSI 的活动。
性能	N/A
操作	N/A
阈值	N/A

清除计数

正式名称	Hyper-V 虚拟存储设备 \ 清除计数
计数器类型	间隔 （整个周期中采样）
描述	此计数器表示此虚拟设备上已发生的清除操作的总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

读取的字节数 / 秒

正式名称	Hyper-V 虚拟存储设备 \ 读取的字节数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	此计数器表示此虚拟设备上每秒读取的字节总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

写入的字节数 / 秒

正式名称	Hyper-V 虚拟存储设备 \ 写入的字节数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	此计数器表示此虚拟设备上每秒写入的字节总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

读取计数

正式名称	Hyper-V 虚拟存储设备 \ 读取计数
计数器类型	间隔 （整个周期中采样）
描述	此计数器表示此虚拟设备上已发生的读取操作的总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

写入计数

正式名称	Hyper-V 虚拟存储设备 \ 写入计数
计数器类型	间隔 （整个周期中采样）
描述	此计数器表示此虚拟设备上已发生的写入操作的总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

网络计数器

此部分描述属于 Windows 资源网络组件的计数器。

以下网络计数器不特定于 Hyper-V。这些是 Windows 相关的常规计数器。有关详细信息，请参见 “相关的 Windows 计数器”（第 268 页）。

- 字节总数 / 秒
- 卸载的连接数
- 数据包数 / 秒
- 出站错误数据包数
- 接收错误数据包数

字节数 / 秒

正式名称	Hyper-V 虚拟交换机 \ 字节数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	此计数器表示每秒遍历虚拟交换机的字节总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

数据包数 / 秒

正式名称	Hyper-V 虚拟交换机 \ 数据包数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	此计数器表示每秒遍历虚拟交换机的数据包的总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

丢弃的字节数

正式名称	Hyper-V 旧版网络适配器 \ 丢弃的字节数
计数器类型	间隔 （整个周期中采样）
描述	丢弃的字节数是网络适配器上丢弃的字节数。
使用备注	在安装集成服务之前，需要旧版网络适配器以使 VM 工作。一旦 VM 通过集成服务工作，您就应该使用网络适配器。
性能	N/A
操作	N/A
阈值	N/A

发送的字节数 / 秒

正式名称	Hyper-V 旧版网络适配器 \ 发送的字节数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	发送的字节数 / 秒是每秒通过网络适配器发送的字节数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

接收的字节数 / 秒

正式名称	Hyper-V 旧版网络适配器 \ 接收的字节数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	接收的字节数 / 秒是每秒在网络适配器上接收的字节数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

字节数 / 秒

正式名称	Hyper-V 虚拟网络适配器 \ 字节数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	此计数器表示已遍历网络适配器的字节总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

数据包数 / 秒

正式名称	Hyper-V 虚拟网络适配器 \ 数据包数 / 秒
计数器类型	间隔差异计数器 （比率 / 秒）。
描述	此计数器表示网络适配器每秒接收的字节总数。
使用备注	N/A
性能	N/A
操作	N/A
阈值	N/A

类属计数器

此部分描述属于 Hyper-V 系统的计数器。

运行状况正常

正式名称	Hyper-V 虚拟机运行状况概要 \ 运行状况正常
计数器类型	瞬时 （每个度量时段采样一次）。
描述	此计数器表示运行状况正常的虚拟机数。
使用备注	如果有状态为严重，则表示某个资源（最可能是磁盘）已被耗尽，或者发生了其他不可恢复的错误。
性能	N/A
操作	N/A
阈值	N/A

运行状况严重

正式名称	Hyper-V 虚拟机运行状况概要 \ 运行状况严重
计数器类型	瞬时 （每个度量时段采样一次）。
描述	此计数器表示运行状况严重的虚拟机数。
使用备注	如果服务器看到“运行状况严重”，您应该执行操作来解决发生的问题。
性能	N/A
操作	N/A
阈值	N/A

逻辑处理器数

正式名称	Hyper-V 管理程序 \ 逻辑处理器数
计数器类型	瞬时 （每个度量时段采样一次）。
描述	系统中存在的逻辑处理器数。
使用备注	这些是管理程序正在管理的核数 /HT 数。如果有双处理器四核而没有 HT，则会看到此数字设置为 8。如果还有 HT，则会设置为 16。如今，该值在启动后是固定的，不会更改。在未来，我们可能支持处理器的热添加和移除，在这种情况下，值将是动态变化的。
性能	N/A
操作	N/A
阈值	N/A

分区数

正式名称	Hyper-V 管理程序 \ 分区数
计数器类型	瞬时 （每个度量时段采样一次）。
描述	系统中存在的分区 （虚拟机）数。
使用备注	系统上的每个虚拟机在名为分区的容器中运行。如果没有任何 VM 在运行，此值将设置为 1，因为 Hyper-V 中名为“根”的“主机操作系统”会同时在分区中运行。因此，如果有 2 个来宾 VM 正在运行，此值将是 3。每个来宾 +1，根 +1。
性能	N/A
操作	N/A
阈值	N/A

总页数

正式名称	Hyper-V 管理程序 \ 总页数
计数器类型	间隔 （整个周期中采样）
描述	管理程序中的引导程序数和存储的页数。
使用备注	管理程序需要内存才能跟踪虚拟处理器、虚拟 TLB 中来宾虚拟地址到系统物理地址的转换条目，等等。因此，总页数跟踪管理程序为管理或分区而使用的内存总量。一页是 4KB。这不是用于支持来宾的总量。您还需要查看工作程序进程 (vmwp.exe) 的大小并计算 vid 中的内存，才能得出此值。由于我们不在非 RTM 发布上发布数字，我们必须等待开销值（未来发布）。总页数可能会根据正在运行的来宾 VM 而变化。
性能	N/A
操作	N/A
阈值	N/A

虚拟处理器数

正式名称	Hyper-V 管理程序 \ 虚拟处理器数
计数器类型	瞬时 （每个度量时段采样一次）。
描述	系统中存在的虚拟处理器数。
使用备注	根和子分区（运行来宾 VM 的分区）中的所有执行都在虚拟处理器 （简称 VP）上进行。至少会看到每个逻辑处理器 (LP) 的 VP。这些是根 VP 数。然后还会看到为来宾配置的 VP，每个来宾 +1。因此，如果系统有 8 个 LP， 1 个来宾正在运行 2 个 VP，则这里的计数将是 10。
性能	N/A
操作	N/A
阈值	N/A

监控的通知数

正式名称	Hyper-V 管理程序 \ 监控的通知数
计数器类型	间隔 （整个周期中采样）
描述	向管理程序注册的监控的通知数。
使用备注	监控的通知数是 Hyper-V 用于减少虚拟化开销的中断合并技术的一部分。例如，当来宾有数据要通过网络传输时，它可以为每个数据包发送一个中断到根 VP，这将实际执行 I/O，或者可以发送一个让根知道数据正在开始流动的中断。此计数器指示正在设置为根和来宾的中断的“流”数。
性能	N/A
操作	N/A
阈值	N/A

优化和调优

以下部分包括优化 Hyper-V 主机 （服务器）和虚拟机的性能的最佳实践和建议。

服务器优化和调优

为获得最佳性能而配置 Hyper-V 服务器需要关注四个主要区域：

处理器、内存、 I/O 和网络。

CPU 性能最佳实践

- 要获得 Hyper-V 的最佳处理器性能，建议运行 Windows Server 2008 和 Hyper-V 角色或 Microsoft Hyper-V Server 2008 的服务器核心安装。只有将 Hyper-V 角色加载到父分区上，Windows Server 2008 才会最小化父分区所需的处理能力，为子分区提供更多的处理能力。
- 建议为服务器提供多核处理器。最新的多核处理器会提供最佳性能。
- 选择缓存 (L2/L3) 较大的处理器会增加性能。

内存性能最佳实践

- 为在 Hyper-V 服务器上获得最快的内存性能，请使用最快的可用内存。如果目标是获得可能的最佳性能，但在服务器上运行可能的最大数量虚拟机，请使用可提供最大容量并对其他组件具有补偿能力的内存，如更快的处理器或速度更快的磁盘子系统。

- 购买可能的具有最高密度模块的服务器以便能够不浪费插槽而进一步进行扩展。例如，可以安装具有 16 ª 1-GB 模块、8 ª 2-GB 模块、4 ª 4-GB 模块或 2 ª 8-GB 模块的 16 GB RAM。使用 8-GB 模块，可以用最少的插槽获得最多的内存量。这允许您通过添加更多内存模块来扩展服务器，而不必卸下较低密度的模块。
- 当计算为服务器配置多大的 RAM 时，应该在物理服务器中为父分区最少分配 1 GB 的 RAM。
- 为获得最佳性能，确定要分配给 NUMA（非统一内存访问）系统上的虚拟机内存的最大内存量，然后按每个处理器至少购买那么多内存。应该将内存均匀分布到每个处理器上，以最大化内存的本地节点利用率，并减少对其他节点的内存调用数。因为 Hyper-V 中的虚拟机最多可以有 64 GB 的 RAM，一般不可能分配最大的内存量让它全部驻留在一个处理器节点上。
- 在 Hyper-V 服务器的完全安装选项之上，使用 Windows Server 2008 服务器核心安装选项和 Hyper-V 角色。这会为您节省下大约 80 MB 的 RAM 可供子分区使用。

I/O 性能最佳实践

- 配置防病毒应用程序排除文件扩展名或进程。建议使用进程排除方法而不是文件排除方法，因为前者提供更好的保护。配置防病毒软件排除 Hyper-V 管理进程时，应该排除 Hyper-V 虚拟机管理服务 (VMMS.exe) 和虚拟机工作程序进程服务 (Vmwp.exe)。如果防病毒应用程序不支持排除进程，应该在防病毒文件排除列表中添加 .vhd、.avhd、.vfd、.vsv 和 .xml 文件扩展名以不扫描这些扩展名的文件。

- 在 Hyper-V 服务器中使用 10,000-RPM（每分钟转数）或更快的驱动器，以最小化虚拟机的数据读取 / 写入时间。使用 10,000-RPM 驱动器而非 7200-RPM 驱动器，可显著增加每分钟执行读取和写入操作的次数。
- 在 Hyper-V 服务器中使用 SATA 或 SAS 驱动器类型，以通过允许单个硬盘一次排队多个 I/O 请求和动态修改执行操作的顺序来提高性能。
- 对于使用内部硬盘驱动器启动的 Hyper-V 服务器，利用 RAID 1（镜像）以提供父分区和 Hyper-V 配置设置的容错性。
- 对于虚拟机存储器，利用 SAN，它可提供 RAID 0（带区）+1（镜像）配置冗余、iSCSI 目标功能和使用高 RPM 命令排队的 I/O 硬盘驱动器的能力。在同一个机柜中选择一个支持 SATA 和 SAS 的硬盘驱动器将会提供最大的灵活性。创建 RAID 0+1 磁盘阵列时，应该使用尽可能多的心轴来分布 I/O 负载。

网络性能最佳实践

- 至少将一个物理网络适配器专用于 Hyper-V 管理和备份。将一个网络适配器专用于 Hyper-V 管理意味着 Hyper-V 服务器的管理和备份网络流量不会影响虚拟机流量。
- 将一个网络适配器专用于 iSCSI 通信，并使用在硬件中提供 iSCSI 处理支持的适配器。
- 将一个网络适配器专用于群集通信。
- 不要在 Hyper-V 主机群集上启用 TCP 烟囱卸载。Windows Server 2008 中的故障转移群集不使用 TCP 烟囱卸载功能。
- 在非群集 Hyper-V 服务器上启用 TCP 烟囱卸载。即使绑定到外部虚拟网络的物理网络适配器不使用 TCP 卸载引擎，其他适配器也会使用。

- 启用 TCP 烟囱之前和之后测试应用程序性能。并非所有应用程序都可以利用 TCP 烟囱卸载，某些网络适配器无法处理来自 TCP 烟囱卸载的额外负载。在任何一种情况下，启用 TCP 烟囱卸载都可能会对某些应用程序网络性能造成负面影响。
- 启用巨大帧允许一次发送更多数据，因此减少需要发送的数据包数，这可通过允许对帧标头作较少处理来减少处理器开销并增加吞吐量。
- 巨大帧可以显著提高尝试在网络上传输大量数据的应用程序或协议的性能。
- 在所有成组网络适配器上禁用接收端负载平衡 （通过不同 MAC 地址响应来自不同客户端的 ARP 请求）。如果不这样做，则连接到与成组网络适配器绑定的外部虚拟网络的虚拟机无法与 Hyper-V 服务器进行外部通信。

虚拟机优化和调优

为获得最佳性能而配置虚拟机需要关注四个主要区域：处理器、内存、I/O 和网络。

CPU 性能最佳实践

- 使用 Windows Server 2008 或更高版本的服务器操作系统，以获得最佳的 Hyper-V 子分区性能。将任何 Windows 2000 Server 或 Windows Server 2003 虚拟机迁移到 Windows Server 2008 以提高虚拟机的性能，并减少 Hyper-V 服务器上的负载。
- 对于任何支持的来宾操作系统，应该首先安装集成服务，以提高模拟设备的性能和吞吐量。

- 迁移到虚拟机时，评估现有物理服务器的处理器使用率。可能已使用最小或标准硬件方法购买了物理服务器。如果处理器使用率未显示正在利用多个处理器，请将迁移的虚拟机配置为单个处理器并监控处理器利用率。
- 从虚拟机上卸下不需要的虚拟 CD/DVD 驱动器。即使您不使用驱动器，系统也必须定期检查 CD/DVD 驱动器中是否插入了使用 CPU 周期的介质。
- 如果 PXE 启动或操作系统对合成网络适配器不支持集成服务，请使用旧版网络适配器。旧版网络适配器通过虚拟机工作程序进程处理数据包需要较长的主机处理器时间。由于通过虚拟化堆栈的路径，旧版网络适配器的吞吐能力低于合成的适配器。
- 需要旧版网络适配器进行每日通信的虚拟机应该与独立的 Hyper-V 服务器隔离。这将使旧版适配器的更多处理器开销不会影响使用合成网络适配器运行虚拟机的 Hyper-V 服务器的性能或可扩展性。
- 在将控制台访问限制为一组可信个人的计算机上，禁用屏幕保护程序可以节省空闲处理器周期。对于需要屏幕保护程序锁定控制台以阻止未经授权的访问的计算机，不显示图像的空白屏幕保护程序至少要检查是否有关键序列。
- 对虚拟机的处理器、网络和磁盘 I/O 的工作负载配置文件进行分析，以了解添加该虚拟机对现有工作负载配置文件有何影响。建议不要组合一系列性能和 I/O 重叠的 VM。
- 如果您有虚拟机需要保证在需要时的处理能力，请使用预留的容量设置（保证逻辑处理器的百分比）。

- 如果 Hyper-V 服务器上的虚拟机的处理器活动突然增加并影响主机上的其他虚拟机，请使用容量限制来限制这些虚拟机以保持预定性能。
- 如果主机上的 VPTR 高，但 LPTR 低，则存在未分配足够处理能力的虚拟机。在每个虚拟机中使用 VPGRT 计数器以确定哪个虚拟机正在以高处理器利用率运行，然后将更多虚拟处理器添加到该虚拟机。如果来宾操作系统不支持更多虚拟处理器，请通过添加更多虚拟机并在虚拟机之间平衡工作负载来扩展应用程序。
- 如果 LPTR 高而 VPTR 低，则存在很多运行轻负载的虚拟机。在虚拟机之间切换上下文会导致主机处理器性能瓶颈。如果主机上正在运行的虚拟机的处理器利用率突然升高，则有两种可能的结果。虚拟机将以其他正在运行的虚拟机的开销获取更多处理能力，或者 VM 不会获取更多处理能力而影响其性能。如果经常发生这种情况，则两者都不是所希望的结果。考虑再添加一个 Hyper-V 服务器，将虚拟机移到该主机上。
- 如果 VPTR 和 LPTR 都高，则预订了太多 Hyper-V 服务器处理器。应该再添加一个 Hyper-V 服务器，并在服务器之间平衡现有虚拟机。

内存性能最佳实践

- 务必为父分区预留最小 1 GB 的内存。应该为 VM 分配足够的内存，以最小化正常操作期间对磁盘的分页，但不消除分页。
- 如果“\ 内存 \ 可用兆字节数”计数器持续显示可用内存小于 10%，并且“\ 内存 \ 页数 / 秒”计数器显示大于 1000，则应该向虚拟机分配更多 RAM。如果“\ 内存 \ 可用兆字节数”计数器持续显示可用内存高于 50%，并且“\ 内存 \ 页数 / 秒”计数器持续显示小于 250，应该考虑减少分配给虚拟机的 RAM。

I/O 性能最佳实践

- 对平衡方法使用固定的虚拟硬盘（封装了单个文件的虚拟硬盘）。它们提供性能、可移植性和快照支持的最快组合。
- 虚拟机数据驱动器应该使用连接到 SCSI Controller 的虚拟硬盘以实现最佳性能和最低处理器开销。
- 将虚拟硬盘文件置于单独的物理磁盘上以获得最佳性能。

网络性能最佳实践

- 加载集成服务并使用合成网络适配器以最大化网络性能 —— 合成网络适配器通过虚拟机总线 (VMBus) 上的专用通道在子和父分区之间通信。
- 使用旧版网络适配器来通过预执行环境加载虚拟机，然后切换到合成网络适配器（假定来宾操作系统有集成服务的支持版本）。
- 购买能为 Hyper-V 服务器提供大型发送卸载和 IPv4 TCP 校验和卸载功能的物理网络适配器。务必正确启用并配置父分区驱动程序设置中的选项。
- 通过使用为 Hyper-V 服务器提供大型发送卸载和 IPv4 TCP 校验和卸载功能的物理网络适配器，减少在父分区中处理数据包所需的处理器周期，并增加虚拟机的吞吐量。务必正确启用并配置父分区驱动程序设置中的选项。
- 为了使 VLAN 标记能够提供可能的最高性能，物理网络适配器应该支持大型发送卸载和 TCP 校验和卸载。
- 如果虚拟机网络适配器输出队列经常大于 2，则虚拟机需要另外的网络适配器来处理网络负载。该附加的网络适配器可以绑定到相同虚拟网络或另一个虚拟网络。

- 要确定现有虚拟网络是否可以处理更多流量，测量主机上的“\网络接口(*)\输出队列长度”性能计数器以确定队列长度。如果主机网络适配器队列长度大于 2，应该添加另外的物理网络适配器，创建绑定到该物理网络适配器的新外部虚拟网络，并将虚拟机重新分配到该新外部虚拟网络以平衡网络负载。
- 为 Hyper-V 服务器的所有物理和虚拟网络适配器配置相同的最大传输单位 (MTU)。

16

VMware 监控

此章节描述基于 VMware 的虚拟机监控的最佳实践。

此章节包括：

概述

VMware 是面向各种规模的企业的全球桌面到数据中心虚拟化解决方案的引领者之一。

VMware 提供的虚拟化产品范围广泛，从用于虚拟化桌面和服务器的免费软件到用于优化数据中心和 IT 基础结构的企业级综合平台。这些产品解决 IT 组织面对的不同难题，例如：服务器合并、基础结构优化、维护高可用性和灾难恢复、最小化宕机时间、自动执行实验室管理等等。

VMware 引入了 **ESX** 和 **ESXi** 管理程序，通过创建自动执行的动态数据中心，向应用程序提供最高级别的可靠性和性能。

VMware ESX 和 VMware ESXi 是 “裸机”管理程序。这意味着它们可以直接安装在物理服务器之上，可以分成多个虚拟机。它们可以同时运行，共享基础服务器的物理资源。每个虚拟机表示一个完整的系统（包括处理器、内存、网络资源、存储器和 BIOS），可以运行未经修改的操作系统和应用程序。

VMware ESX 和 ESXi 的功能和性能是相同的，两个管理程序之间的差异在于其体系结构和操作管理。 VMware ESXi 是 VMware 的最新管理程序体系结构。它占用的内存量极少，不依赖于通用操作系统，为安全性和可靠性设立了新标杆。VMware ESXi 的较小内存占用量和类似于硬件的可靠性，使得它还能够预安装到行业标准的 x86 服务器上。

本章重点介绍 VMWare ESX 服务器，因为它是最常用的 VMWare 虚拟化平台。核心目的是为了帮助性能工程师更好了解 ESX 服务器的体系结构。同时还使工程师能够成功监控性能测试并在必要时进行性能调优。

体系结构

VMware ESX 服务器是直接在物理硬件上运行的管理程序，它创建使很多虚拟机可以在完全隔绝的环境中共享相同物理资源的逻辑系统资源池。

ESX 服务器在系统硬件和虚拟机之间插入虚拟化层。这将系统硬件转变为逻辑计算资源池， ESX 服务器可以将这些资源动态分配到任何操作系统或应用程序。虚拟机上运行的来宾操作系统与虚拟资源交互，就好象它们是物理资源一样。

VMware 体系结构层

任何虚拟化环境都由几个组成部分所构成：

- 主机计算机
- 虚拟化软件
- 虚拟机
- 来宾操作系统

以下部分更详细地描述这些组成部分。

主机计算机

虚拟环境中的主机计算机向虚拟机提供资源。核心资源如下：CPU、内存、NIC、磁盘。主机计算机上的资源越多，可以受管的虚拟机就越多。

如果主机计算机自身要使用一些资源，则虚拟机将使用剩余的资源。

虚拟化软件

虚拟化软件层使每个虚拟机能够访问主机资源。它还负责在各种虚拟机之间计划物理资源。虚拟化软件是整个虚拟化环境的基础。它创建要使用的虚拟机、管理提供给虚拟机的资源、在争用特定资源时计划资源使用，以及提供虚拟机的管理和配置界面。

VMware 提供三个版本的虚拟化软件。

- **VMware 工作站**是可以在主机计算机的操作系统上安装的虚拟化软件包。使用 VMware 工作站的主要限制是只能在登录到主机工作站后才能运行虚拟机。注销时，虚拟机关机。VMware 工作站主要用作本地用户工具，这意味着没有远程管理功能。这不适合于生产环境。
- **VMware GSX 服务器**与 VMware 工作站类似，因为它也是可以在主机计算机的操作系统（Linux 或 Windows）上安装的虚拟化软件包。但是，VMware GSX 服务器比 VMware 工作站又先进了一点。它能够对虚拟机进行某些远程管理和远程控制台访问。各种虚拟机都可以配置为作为服务运行，无需任何控制台交互。主要限制是它必须通过主机操作系统从主机硬件使用资源。这确实会限制 GSX 的可扩展性和性能，因为虚拟机不能直接访问硬件。

- **VMware ESX 服务器**是一个完整的操作系统。ESX 服务器的设计用途完全是为了向虚拟机提供可能的最佳性能，并允许管理员对主机资源的共享和利用方式进行控制和调整。 ESX 服务器所提供的 VM 性能级别是 GSX 或工作站中绝不可能存在的。它还支持更高级的资源分配、性能微调、更好的 VM- 处理器比率，以及更高级的资源共享。VMware 发布了 ESX 服务器硬件兼容性列表 (HCL)。如果您用于 ESX 的硬件列在 HCL 上，则可以确信一切都将按预期工作。ESX 还使您能够杜绝主机操作系统上存在的一切问题，因为 ESX 没有主机操作系统。 ESX 服务器不仅是其自己的操作系统，同时还是虚拟化软件。

虚拟机

虚拟机实际上是提供给来宾操作系统的虚拟硬件（或组合虚拟硬件和虚拟 BIOS）。它是物理硬件的基于软件的虚拟化。来宾操作系统不会知道安装了它的硬件是虚拟的。

所有来宾操作系统都知道处理器类型、网卡类型、内存量、磁盘空间，等等。

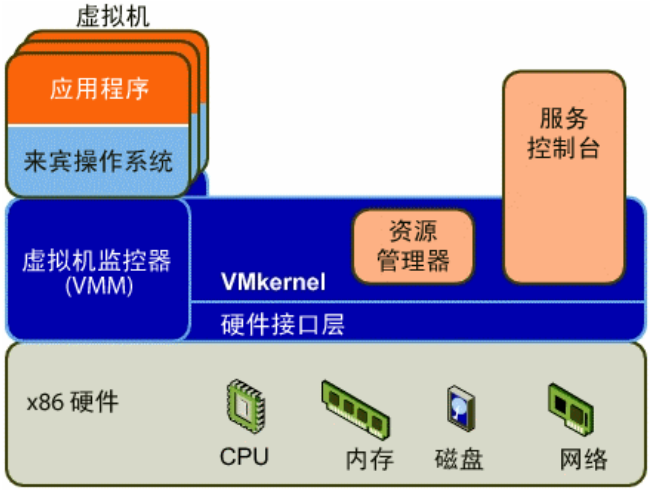
来宾操作系统

来宾操作系统是在虚拟机上运行的基于 Intel 的操作系统 （Windows、 Linux、 Novell、 DOS 等）。

来宾操作系统（或“来宾计算机”或简称“来宾”）是 VM 上安装的软件。设置操作系统后，可以安装通常在该操作系统上运行的任何应用程序。

虚拟化软件内部

现在来看看 ESX 服务器体系结构，即：前面所述的虚拟化软件层。下图描述 ESX 的主逻辑部分：



VMkernel

VMkernel 是 VMware 开发的直接在 ESX 服务器主机上运行的高性能操作系统。VMkernel 控制和管理硬件上大多数物理资源，其中包括：

- 内存
- 物理处理器
- 存储器
- 网络 Controller

VMkernel 实现 ESX 服务器的虚拟化、资源管理和硬件接口组件。

VMkernel 资源管理器

资源管理器将基础服务器的物理资源进行分区。它使用按比例共享机制将 CPU、内存和磁盘资源分配给开启的虚拟机。

用户可以指定每个虚拟机的份额、预留和限制。资源管理器在将 CPU 和内存分配给每个虚拟机时会考虑该信息。

VMkernel 硬件接口层

硬件接口对 ESX 服务器（和虚拟机）用户隐藏硬件差异。这样能够提供特定于硬件的服务，包括：

- 设备驱动程序 —— 直接与硬件设备交互。
- 虚拟机文件系统 (VMFS) —— 分布式文件系统。已针对特大型文件（例如，虚拟机磁盘和交换文件）进行了优化。

虚拟机监控器 (VMM)

虚拟机监控器 (VMM) 负责虚拟化 CPU。当虚拟机开始运行时，会将控制权交给 VMM，VMM 开始执行来自虚拟机的指令。控制权交给 VMM 涉及设置系统状态以使 VMM 能够在裸机上直接运行。

服务控制台

服务控制台是 Linux 的有限分配版，基于 Red Hat Enterprise Linux 3 Update 6。

服务控制台提供执行环境，用于监控和管理 ESX 服务器系统。服务控制台的目的是启动物理服务器计算机，并管理虚拟机。在计算机启动到服务控制台之后，VMkernel 将加载并

取得计算机的控制权。服务控制台支持非性能关键的设备，例如鼠标、键盘、屏幕、软盘驱动器、 CD-ROM、 COM 端口和并行端口。服务控制台还运行实现支持与管理功能的应用程序。

监控工具

VMware 基础结构提供了一些用于度量性能、可扩展性、可用性、可靠性、稳定性和可管理性的性能计数器。这些计数器帮助监控与 ESX 服务器一起运行的虚拟机和基础物理服务器的资源利用率。

有各种工具可以用于收集这些性能计数器。以下部分包括供应商提供的工具以及 HP 提供的推荐监控工具的详细说明。

虚拟中心

数据中心管理员最常用的用户友好监控工具。它允许管理员从单个管理控制台监控和控制 VMware 服务器和虚拟机组。它显示资源利用率概要，还提供 CPU、内存、网络和磁盘资源的历史图形。

安装了虚拟中心后，就可连接到要管理和监控的主机。主机将添加在 VC 客户端内，并且主机将在 ESX 服务器上“安装”代理程序并创建本地服务帐户，以便能够从 VC 客户端内执行命令。

添加主机之后可以立即查看其性能数据。度量将以每 20 秒的间隔收集，并且可以通过 VC 客户端在要查看主机的 “性能”选项卡中查看。然后可以选择查看数据的时间范围 （过去的天、周、月或年）并分析数据。

命令行工具

esxtop

提供每个虚拟机以及服务控制台和某些 VMkernel 系统服务的 CPU 和内存的实时视图。它还显示每个物理处理器的 CPU 利用率、内存利用率，以及对虚拟机可用的每个物理磁盘和网络设备的磁盘和网络带宽。

esxtop 命令可以在主机操作系统的本地控制台上运行，或者可通过与主机的 *ssh* 会话运行。 **esxtop** 命令是交互式的，将以每几秒的间隔更新度量。

要添加更多列以供查看，可以单击 f 键获取要联机监控的其他度量。除 **esxtop** 的交互选项以外，还有 4 个命令行参数可以使用。使用这些命令行参数可以：

- 设置屏幕刷新 (**d**)
- 将实用工具设置为 “安全模式”，同时停止交互式命令 (**s**)
- 使用 “批处理模式”输出数据以进行日志记录 (**b**)
- 指定要执行的刷新次数 (**n**)

vmkusage

显示运行 ESX 服务器及其关联虚拟机的单个物理主机的资源利用率的历史图形。此类显示可以很直观地表示利用率趋势。

数据以最近、每日、每周或每月图表显示，这些图表提供在主机上运行的控制台操作系统和虚拟机的趋势分析。各种数据分析将按每个时间间隔显示在 vmkusage 输出中。

vmkusage 包是在 ESX 安装过程中安装的，但是默认情况下不启用。为了正确配置 **vmkusage** 以开始收集信息和生成图形，应当运行命令“*vmkusagectl install*”。此命令设置相应的 cron 作业以每分钟运行一次 **vmkusage** 数据收集进程来收集数据。

该工具以网页形式生成图形，您可以访问以下地址进行查看：**http://<ESX 服务器名称 >.< 您的公司 >.com/vmkusage**

管理用户界面状态监控器

显示物理机和虚拟机性能统计信息的简单概要，但不提供用于趋势分析的历史图形。

状态监控器显示最后五分钟的结果平均值。将收集最后五分钟内的各个 20 秒样本并求出平均值。

要查看状态监控器，请打开连接到服务器的浏览器并输入：
https://< 主机名 >:8333。登录之后，将出现状态监控器页面。

HP SiteScope

HP SiteScope 提供用于监控 VMware 基础结构的综合工具。它基于 VMware 的虚拟中心功能，可以提供有关主机（物理基础结构）和来宾（逻辑计算机）的信息。

注：建议不要从运行于虚拟机上的来宾操作系统内运行性能工具。此类工具假定来宾操作系统是唯一运行的操作系统，不考虑共享基础物理 CPU 的情况，然而关键数据可能会丢失并且可能提供无效结果。

最重要的 VMware 计数器

VMware 提供多种自己的计数器，包括用于度量处理器、内存、网络和 I/O 设备利用率的计数器。监控时，建议与主机计算机的操作系统计数器关联。这样可以从另一个角度了解虚拟机性能。

	计数器	描述
CPU	usage	指示收集间隔内的 CPU 使用情况（以百分比表示）。
	usagemhz	指示收集间隔内的 CPU 使用情况（以 MHz 表示）。
	ready	指示过去更新间隔内等待 CPU 可用所耗用的时间。
	reservedCapacity	指示主机根资源池的直接子级的预留属性的总和。
	wait	指示最后更新间隔期间等待硬件或 VMkernel 锁定线程锁定所耗用的时间。
	swapwait	指示等待内存换入所耗用的时间。

	计数器	描述
内存	usage	指示可用内存量 （以百分点表示）。
	vmmemctl	指示其他 VM 请求的内存量。
	active	指示 VM 真正需要的内存量。
	granted	指示主机授予 VM 的内存量。
	consumed	指示来宾内存虚拟机所消耗的主机内存量。
	overhead	指示 VMkernel 用于维护和执行 VM 的内存。
	swapin	指示从磁盘换入内存的速率。
	swapout	表示启动以来 CLR 识别的线程总数。
	shared	指示共享内存的平均量。
I/O	usage	指示对主机或虚拟机的所有磁盘实例所读取和写入的数据总和。
	read	指示从磁盘读取数据的速率。
	write	指示数据写入磁盘的速率。
	numberRead	指示上一采样周期中的 I/O 读取操作数目。
	numberWrite	指示上一采样周期中的 I/O 写入操作数目。

	计数器	描述
网络	usage	指示对主机或虚拟机的所有 NIC 实例传输和接收的数据总和。
	received	指示所接收流量的平均网络吞吐量。
	transmitted	指示所传输流量的平均网络吞吐量。
	droppedRx	指示采样周期中丢弃的接收数据包数目。
	droppedTx	指示采样周期中丢弃的传输数据包数目。

CPU 计数器

此部分描述提供处理器利用率相关信息的计数器。

usage

正式名称	usage
计数器类型	间隔 （整个周期中采样）。
描述	收集间隔内的 CPU 使用情况（以百分比表示）。适用于主机和虚拟机。
使用备注	精度以百分比计，设为百分点的 1/100，即 1 = 0.01% 。此计数器范围值在 0 和 10000 之间。
性能	此度量允许比较不同速度的两个主机。 CPU 使用情况可以仅在 VM 级别上度量。 如果在主机系统上执行为所有 VM 创建快照或删除快照之类的操作，则此计数器的值可能发生显著变化。
操作	N/A
阈值	值 100% 表示系统上所有处理器核达到了完全的使用率。

usagemhz

正式名称	usagemhz
计数器类型	间隔 （整个周期中采样）。
描述	显示收集间隔内的 CPU 使用情况（以 MHz 表示）。适用于主机和虚拟机。
使用备注	以 MHz 计
性能	此度量允许比较不同速度的两个主机。 以 MHz 计的 CPU 使用情况可以针对每个 VM 和虚拟 CPU 的每个实例进行度量。 如果在主机系统上执行为所有 VM 创建快照或删除快照之类的操作，则此计数器的值可能发生显著变化。
操作	ESX 服务器和虚拟机使用的物理 CPU 可以通过 HostSystem.summary.quickStats.overallCpuUsage 和 VirtualMachine.summary.quickStats.overallCpuUsage 进行验证。quickStats 没有间隔。它是特定性能计数器在 VI 服务捕获其值时的值样本。
阈值	N/A

ready

正式名称	ready
计数器类型	间隔 （整个周期中采样）。
描述	过去更新间隔内等待 CPU 可用所耗用的时间。
使用备注	以毫秒计
性能	此度量允许比较不同速度的两个主机。 cpu.ready 计数器按每个 CPU 实例进行报告。
操作	有时， %ready 超过 100。这说明 VM 需要 CPU 但却无权访问它。如果此值只在短时间内超过 100，则不存在问题。如果 %ready 在相当长的时间里超过 100，则说明 VM 缺少 CPU 的时间过长。
阈值	N/A

reservedCapacity

正式名称	reservedCapacity
计数器类型	间隔 （整个周期中采样）。
描述	主机根资源池的直接子级的预留属性的总和。
使用备注	此度量报告主机状态
性能	N/A
操作	仅当父级标记为 reservationExpandable 时，子级的预留总和可以大于父级。
阈值	N/A

wait

正式名称	wait
计数器类型	间隔 （整个周期中采样）。
描述	等待时间是最后更新间隔期间等待硬件或 VMkernel 锁定线程锁定所耗用的时间。
使用备注	以毫秒计。
性能	
操作	此度量可以对虚拟 CPU 的每个实例进行度量
阈值	N/A

swapwait

正式名称	swapwait
计数器类型	间隔 （整个周期中采样）。
描述	交换等待时间是 VM 等待内存换入所耗用的时间。VM 在等待内存时处于不活动状态。
使用备注	以毫秒计。
性能	N/A
操作	此度量可以对每个虚拟机进行度量。
阈值	N/A

内存计数器

此部分描述属于 VMware 系统内存管理的计数器。它们提供有关内存消耗和内存池等的数据。

usage

正式名称	usage
计数器类型	间隔 （整个周期中采样）。
描述	收集间隔内的内存使用情况（以百分比表示）。适用于主机和虚拟机。
使用备注	精度以百分比计，设为百分点的 1/100，即 1 = 0.01%。此计数器范围值在 0 和 10000 之间。
性能	此度量允许比较不同内存容量的两个主机。
操作	N/A
阈值	值 100% 表示系统上所有内存达到了完全的使用率。

vmmemctl

正式名称	vmmemctl
计数器类型	间隔 （整个周期中采样）。
描述	涨缩 (balloon) 驱动程序当前请求的内存量，表示主机开始从内存富余的 VM 中收回内存，重新分配给需要大量活动内存的 VM。
使用备注	以 KB 计。
性能	此度量是指由于“涨缩”功能而收回的内存量。它使用机器页而不是物理页，因为涨缩页是 1:1 映射的。 但如果主机正在执行涨缩，请检查交换率 （swapin 和 swapout）是否反映了性能问题。
操作	N/A
阈值	N/A

active

正式名称	active
计数器类型	间隔 （整个周期中采样）。
描述	在过去较短时间内 VM 使用的内存量。
使用备注	以 KB 计。
性能	这是 VM 当前需要多少内存的真实数字。另外，不使用的内存可能被换出或进行涨缩，而不会影响来宾性能。
操作	N/A
阈值	N/A

granted

正式名称	granted
计数器类型	间隔 （整个周期中采样）。
描述	主机授予 VM 的内存量。
使用备注	以 KB 计。
性能	N/A
操作	N/A
阈值	N/A

consumed

正式名称	consumed
计数器类型	间隔 （整个周期中采样）。
描述	来宾内存虚拟机所消耗的主机内存量。
使用备注	以 KB 计。
性能	对于虚拟机：来宾内存虚拟机所消耗的主机内存量。 对于主机系统：此计数器可以按以下方法计算： **主机的总内存 - 可用内存** 它包括保留用于服务控制台的内存。注意，保留用于服务控制台的全部内存视为已使用。
操作	此计数器是指机器页。
阈值	N/A

overhead

正式名称	overhead
计数器类型	间隔 （整个周期中采样）。
描述	VMkernel 用于维护和执行 VM 的内存。
使用备注	以 KB 计。
性能	N/A
操作	N/A
阈值	N/A

swapin

正式名称	swapin
计数器类型	间隔 （整个周期中采样）。
描述	从磁盘换入内存的速率。
使用备注	以 KB 计。
性能	如果这个数字很大，表示存在缺少内存问题，并表明性能将因此而下降。
操作	N/A
阈值	N/A

swapout

正式名称	swapout
计数器类型	间隔 （整个周期中采样）。
描述	内存换出到磁盘的速率。
使用备注	以 KB 计。
性能	如果这个数字很大，表示存在缺少内存问题，并表明性能将因此而下降。
操作	N/A
阈值	N/A

shared

正式名称	shared
计数器类型	间隔 （整个周期中采样）。
描述	共享内存的平均量。
使用备注	以 KB 计。
性能	共享内存表示可以共享内存存量的整个内存池。
操作	N/A
阈值	N/A

I/O 计数器

此部分描述属于 VMware 系统 I/O 管理的计数器。

usage

正式名称	usage
计数器类型	间隔 （整个周期中采样）。
描述	对主机或虚拟机的所有磁盘实例所读取和写入的数据总和。
使用备注	以 KB 计。
性能	对于主机，此度量可以在堆叠图上按每个虚拟机表示。
操作	N/A
阈值	N/A

read

正式名称	read
计数器类型	间隔 （整个周期中采样）。
描述	磁盘读取速率。它指示性能间隔中读取的数据量。
使用备注	以 KB 计。
性能	将 blocksRead 乘以 blockSize 可以计算出此度量。
操作	N/A
阈值	N/A

write

正式名称	write
计数器类型	间隔 （整个周期中采样）。
描述	磁盘写入速率。它指示性能间隔中写入磁盘的数据量。
使用备注	以 KB 计。
性能	将 blocksWritten 乘以 blockSize 可以计算出此度量。
操作	N/A
阈值	N/A

numberRead

正式名称	numberRead
计数器类型	瞬时 （每个度量时段采样一次）。
描述	上一采样周期中的 I/O 读取操作数目。
使用备注	数字
性能	这些操作数目大小可以变化的，最大 64 KB。
操作	N/A
阈值	N/A

numberWrite

正式名称	numberWrite
计数器类型	瞬时 （每个度量时段采样一次）。
描述	上一采样周期中的 I/O 写入操作数目。
使用备注	数字
性能	这些操作数目大小可以变化的，最大 64 KB。
操作	N/A
阈值	N/A

网络计数器

此部分描述属于 VMware 系统网络性能的计数器。

usage

正式名称	usage
计数器类型	间隔 （整个周期中采样）。
描述	对主机或虚拟机的所有 NIC 实例传输和接收的数据总和。
使用备注	以 KBps 计。
性能	N/A
操作	N/A
阈值	N/A

received

正式名称	received
计数器类型	间隔 （整个周期中采样）。
描述	所接收流量的平均网络吞吐量。
使用备注	以 KBps 计。
性能	N/A
操作	按每个 NIC 计。
阈值	N/A

transmitted

正式名称	transmitted
计数器类型	间隔 （整个周期中采样）。
描述	所接收流量的平均网络吞吐量。
使用备注	以 KBps 计。
性能	N/A
操作	按每个 NIC 计。
阈值	N/A

droppedRx

正式名称	droppedRx
计数器类型	间隔 （整个周期中采样）。
描述	采样周期中丢弃的接收数据包数目。
使用备注	数字
性能	N/A
操作	按每个 NIC 计。
阈值	N/A

droppedTx

正式名称	droppedTx
计数器类型	间隔 （整个周期中采样）。
描述	采样周期中丢弃的传输数据包数目。
使用备注	数字
性能	N/A
操作	按每个 NIC 计。
阈值	N/A

优化和调优

为达到最佳性能，建议使用以下最佳实践和配置。

CPU 性能最佳实践

对于非常依赖 CPU 的应用程序，任何 CPU 虚拟化开销都会带来总体性能的降低。

为取得最佳 CPU 性能，建议使用以下最佳实践和配置：

- 使用尽可能少的虚拟 CPU —— 不使用的虚拟 CPU 仍然会对 ESX 服务器施加资源需求。
- 确保对 UP HAL/ 内核配置单处理器虚拟机。对于 SMP HAL/ 内核，必须配置多处理器虚拟机。
- 避免在服务控制台中运行耗用过多 CPU 或内存的程序。这会对虚拟机和 ESX 服务器的性能产生不利影响。
- ESX 3.0.1 完全支持 64 位来宾操作系统。 64 位来宾和应用程序性能比相应的 32 位版本更好。
- 来宾操作系统计时器速率可能对性能有影响。不同操作系统有不同计时器中断。传递很多虚拟时钟中断的开销会对来宾性能产生不利影响，并增加主机 CPU 消耗。如果有选择，请使用需要较低计时器速率的来宾。

内存性能最佳实践

有两种内存相关开销是由 ESX 服务器虚拟机导致的：用于访问虚拟机上的内存的额外时间，以及 ESX 服务器其自身代码和数据结构所需的额外内存。内存开销由两部分组成：用于服务控制台和 VMkernel 的系统范围的固定开销，和用于每个虚拟机的额外开销。

对于 ESX 服务器 3.0，服务控制台通常使用 272MB， VMkernel 使用的内存量较小。开销的内存包括保留用于虚拟机帧缓冲区和各种虚拟化数据结构的空间。开销的内存取决于虚拟 CPU 数目、来宾操作系统的配置内存，以及是使用 32 位还是 64 位来宾操作系统。

为取得最佳内存性能，建议使用以下最佳实践和配置：

- 确保主机的物理内存大于 ESX 要使用的内存总量与由任一时间运行的所有虚拟机使用的工作组大小合计数之和。
- 仔细选择分配给虚拟机的虚拟内存量，以允许有足够内存维持在虚拟机上运行的工作应用程序组。
- 如果可能，在 Linux 虚拟机上使用的数量小于来宾物理内存 896MB。如果物理内存量大于 896MB，则 Linux 将使用不同技术映射内核内存。这些技术会在虚拟机监控器上加上额外开销，并且会导致性能略有下降。
- 如果选择对 ESX 服务器过量使用内存，则需要确保 ESX 服务器上有足够交换空间。 ESX 服务器对每个虚拟机创建一个交换文件，其大小等于虚拟机的配置内存大小与其预留内存的差。此交换文件在开机时创建，因此其创建没有性能影响。此交换空间必须大于等于虚拟机的配置内存大小与其预留内存的差。

- 如果交换无法避免，为达到更好性能，确保将虚拟机的交换文件放在高速 / 高带宽的存储系统上。默认情况下，在虚拟机所在位置创建虚拟机的交换文件。这个默认位置是可以更改的，方法是将 sched.swap.dir 选项（在 VI 客户端，编辑设置 > 选项 > 高级 > 配置参数）设置为相应的位置路径。

I/O 性能最佳实践

存储性能问题通常是基础存储设备的配置问题引起的，并且不是 ESX 服务器特有的。很多工作负载对 I/O 操作滞后很敏感。因此，必须正确配置存储设备，这一点非常重要。

为取得最佳存储性能，建议使用以下最佳实践和配置：

- 通过检查 esxtop 报告的排队命令数目（QUED 计数器），确保 VMkernel 中 I/O 没有排队。
- 要优化存储阵列性能，请将 I/O 负载分摊到可用存储路径上（跨多个 HBA 和 SP）。
- 避免对 VMFS（一种分布式文件系统）分区上的文件执行过度的打开 / 关闭操作，因为这些操作的开销会很大。如果可能，访问文件时执行所有所需操作，然后关闭，而不是打开、执行一些操作再关闭，频繁地重复。
- 始终建议为主机试算和调整 VMFS 分区。
- 光纤通道 SAN 的性能通常比 NAS 和 iSCSI 要高。由于使用光纤通道协议，光纤通道 SAN 不会发生 NAS 和 iSCSI 很容易就发生的拥塞和过度订阅问题。
- 确保负载很重的虚拟机不会同时访问同一 VMFS 卷，并确保它们分布在多个 VMFS 卷。大量虚拟机同时访问同一 VMFS 卷时发生的繁重 SAN I/O 会造成磁盘性能很差。
- 避免需要大量文件锁定或元数据锁定的操作，例如，动态增长的 *vmdk* 文件和文件权限操纵。

- 配置 HBA 卡的最大队列深度。
- 如果需要，增加虚拟机的最大未决磁盘请求数
- 对于 iSCSI/NFS，确保不会将若干输入以太网链接转接到数目更少的输出链接中而导致链接过度订阅。在任何时候传输已接近最大容量的大量链接转接到数目更小的链接，都有可能发生过度订阅。
- 那些将大量数据写入存储器的应用程序或系统（例如，数据采集或事务记录系统）不应当共享连接到存储设备的以太网链接。此类应用程序在对存储设备使用多个连接时性能最佳。
- 来宾存储驱动程序通常将 I/O 大小的默认值设为 64K。如果应用程序发出大于 64K 的 I/O，则它们被拆分为 64K 的块。更改注册表设置以发出更大的块可以提高性能。

网络性能最佳实践

为取得最佳网络性能，建议使用以下最佳实践和配置：

- 从一个 vSwitch 连接到物理网络的多个网络适配器构成了 NIC 团队。此类 NIC 团队可以通过将流量分摊到这些物理网络适配器上来提高性能，并且可以在发生硬件故障或网络中断的情况下提供被动故障转移。
- 在 32 位来宾内模拟的默认虚拟网络适配器是配置了 VMware *vlance* 驱动程序的 AMD PCnet32 设备（用于 64 位来宾的 e1000）。但是， vmxnet 的性能比 vlance 要好得多，应该用其实现最佳性能。
- 使用单独 *vSwitch*（因此使用单独的物理网络适配器）以避免服务控制台、*VMkernel* 和虚拟机之间的争用，以及运行繁重网络工作负载的虚拟机之间的争用。
- *VMkernel* 网络设备驱动程序应当配置网络交换机的相同速度和双工设置，否则将出现带宽很低的问题。

- 要在位于同一 ESX 主机上的两个虚拟机之间建立网络，请将两个虚拟机连接到同一虚拟交换机。如果虚拟机连接到不同的虚拟交换机，则流量将流经线缆，从而导致不必要的 CPU 和网络开销。
- 在位于同一 ESX 服务器上的虚拟机之间出现低吞吐量（来宾驱动程序中的缓冲区溢出）的情况下，增加接收缓冲区的数目，并且/或者减少传输缓冲区的数目。
- 为达到最佳网络性能，建议网络适配器支持以下硬件功能：
  - 校验和卸载
  - TCP 分段卸载 (TSO)
  - 处理高位内存 DMA（即：处理 64 位 DMA 地址）的能力
  - 每个 Tx 帧处理多个分散收集（Scatter Gather）元素的能力


Java™
COMPATIBLE