

Web性能测试的专业工具书，
软件测试工程师的良师益友。

零成本实现 Web性能测试

——基于Apache JMeter和Gatling

温素剑 编著

零成本实现 Web性能测试

——基于Apache JMeter和Gatling

温素剑 编著

电子工业出版社

Publishing House of Electronics Industry

北京•BEIJING

内 容 简 介

本书不是一本讲述深奥测试理论的教科书，而是一本实战类的书籍。作者想要达到的目标就是——读者朋友们在认真读完这本书后，马上能在生产实践中用上所学内容。

本书首先介绍基础的性能测试理论，接着详细介绍如何使用 JMeter 来完成各种类型的性能测试，最重要的则是性能测试实战章节。实战章节中作者以测试某大型保险公司电话销售系统为例，手把手教会读者如何用 JMeter 来完成一个实际的性能测试任务。

第 1 章介绍性能测试理论，第 2 章到第 12 章详细介绍 JMeter 工具在各种场景下的使用方法，以及如何分析性能测试结果，第 13 章是性能测试实战。第 13 章以后，将会介绍一种新颖的性能测试工具 Gatling，作为 JMeter 的一种有益补充。

本书适合各类测试人员及计算机专业在校学生参考。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目 (CIP) 数据

零成本实现 Web 性能测试: 基于 Apache JMeter 和 Gatling / 温素剑编著. —北京: 电子工业出版社, 2016.4
ISBN 978-7-121-28287-4

I. ①零… II. ①温… III. ①计算机网络—程序设计 IV. ①TP393

中国版本图书馆 CIP 数据核字(2016)第 046467 号

策划编辑: 张月萍

责任编辑: 徐津平

印 刷: 北京中新伟业印刷有限公司

装 订: 三河市皇庄路通装订厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编: 100036

开 本: 787×980 1/16 印张: 30.75 字数: 684 千字

版 次: 2016 年 4 月第 1 版

印 次: 2016 年 4 月第 1 次印刷

定 价: 79.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zltz@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

前言

我曾经有幸为国内最大的电信设备供应商工作过一段时间，期间听一些资深老员工讲过一个故事。这个故事大概可以被当作笑话来听，不过笑笑之后却总也忘不掉。话说 20 世纪 90 年代初的某一天，国内第一台自主研发的大型固话交换机，终于千呼万唤地“闪亮”登场了。于是乎，这家公司马上向用户大力推销这款设备，但是用户提出了一个很实际的问题彻底难住了这家公司。问题很简单，那就是需要一份性能测试报告来证明这台设备真能支持宣称的话务容量。那时候还没有成熟的电信领域性能测试工具，该怎么办呢？活人岂能让尿憋死，幸好有聪明的领导想出了一个中国式的解决办法。

某天下午，全公司的员工都放下了手头的工作，每人怀抱一部老式电话机（还要靠转盘来拨号），等领导倒数“三、二、一”后集体打电话。据说当时人数不够，达不到用户要求的通话量，甚至出现了一个人操作两部电话机的情况。我没能一睹当时的盛况，一直深感遗憾。

幸好科学技术发展到今天，已经有了多款成熟的性能测试工具，否则测试人员一定会发疯。试想当前的电信交换机话务容量早已翻了不知多少倍，如果还要靠人海战术去测试，即使全公司的员工双手双脚去操作电话机，也肯定是忙不过来的。测试人员应该为测试技术的飞速发展而感到欢心鼓舞。那么现在是否就可以高枕无忧了？答案是否定的。当前测试人员面临的问题，不再是有没有性能测试工具，而是有没有合适的性能测试工具。

怎么界定“合适”一词？我想至少可以包含如下几个方面：

- 技术先进，功能强大。
- 支持多种测试类型（协议）。
- 易学易用。
- 拥有良好的可扩展性。
- 拥有良好的可移植性（跨平台）。
- 合理的价格。

当前性能测试工具很多，但要同时满足以上数个条件的却很少。在 Web 性能测试领域，目前有两种工具被广泛使用，一种是 LoadRunner，另一种就是 JMeter。不过 LoadRunner 并不是一款“合适”的工具，在此并不是否认 LoadRunner 是一款优秀的性能测试工具，它唯一的缺点就是过于昂贵（至于 LoadRunner Licence 及其支持服务的具体价格，感兴趣的朋友可以向 HP 公司了解，相信其价格会将中国 90% 以上的软件公司挡在门外）。JMeter 具备了 LoadRunner 95% 以上的功能，但其价格无限接近于零，可谓性价比极高。当然相对于商业工具 LoadRunner，JMeter 也有其自身的缺点。它最大的缺点就是没有专业的售后支持队伍，不过想想商业工具贵得令人咋舌的维护支持费用，也就能够释怀了。

写作背景

我目前在一家大型保险公司 IT 测试部门工作，带领一个测试团队负责测试公司的电话销售系统。这个系统非常庞大，由多个子系统构成，同时它又与很多公司内部/外部系统（例如银联、银行的交易系统）发生交互，目前公司有数万员工依赖它来完成每日的销售任务。如此复杂的一个系统，偏偏又拥有数量众多的用户，读者朋友可以试想一下，只要此系统稍有异常，业务部门的投诉绝对会让 IT 部门“吃不了兜着走”。

面对频繁的版本发布，严格的系统性能测试是不可或缺的。测试部门也花大价钱购买了商业工具 LoadRunner，但是实际工作中发现测试人员还是受到颇多限制。其一，测试资源存在瓶颈，公司购买的 Licence 是有限的，无法完全满足测试人员的需求，经常出现人等机器的情况，甚至影响到了软件版本的及时发布；其二，公司购买的并发数许可也是有限的，如果需要模拟更大的系统压力，那么对不起，公司还得再掏钱；其三，公司购买的协议类型是有限的，如果需要使用其他未购置的协议类型，对不起，公司依然需要再掏钱（测试人员很难用临时的测试需求，去说服领导拨上一大笔经费）。

在尝试说服领导增加预算失败后，只能转而寻求其他解决办法，那就是开源性能测试工具 JMeter。经过大规模的试用后，发现 JMeter 完全能够满足测试人员的需求。

“云计算”绝对是当前最热的 IT 词汇之一。“云”听起来很虚幻，其实就是瘦客户端加网格计算。今后客户端不再会有大量的计算任务，计算和存储都被放在云上。在作者看来，今后的客户端应该就是一个浏览器，用户的所有操作都是通过浏览器来实现的。Google 发布的操作系统 Chrome OS，就是基于这一理念设计的。B/S 和 C/S 架构的软件系统，应该会慢慢演进到 Browser/Cloud 模式。如此看来，在云计算时代，Web 性能测试依然很重要，而且会越来越重要。因此，本人萌生了写作一本关于 Web 性能测试的书籍的想法。

本书内容

本书着重介绍如何使用 JMeter 和 Gatling 开源性能测试工具，来构建 Web 性能测试体系。

本书不是一本讲述深奥测试理论的教科书，而是一本实战类的书籍。作者想要达到的目标就是——读者朋友们在认真读完本书后，马上就能在生产实践中用上所学内容。本书首先介绍基础的性能测试理论，接着详细介绍如何使用 JMeter 来完成各种类型的性能测试，最重要的则是性能测试实战章节。实战章节中作者以测试某大型保险公司电话销售系统为例，手把手教会读者如何使用 JMeter 来完成一个实际的性能测试任务。本书第 1 章介绍性能测试理论，第 2~12 章详细介绍 JMeter 工具在各种场景下的使用方法，以及如何分析性能测试结果，第 13 章是性能测试实战。从第 14 章开始，将会介绍一种新颖的性能测试工具 Gatling，作为 JMeter 的一种有益补充。

目标读者

本书的目标读者是初级和资深软件测试工程师，以及有意降低性能测试成本的测试经理。本书也适合应届毕业生，要想熟练掌握性能测试的方法和技巧，本书也是一块不错的敲门砖。

感谢

首先要感谢我的家人，正是由于他们默默的支持，我才能静下心来写作。其次还要感谢参与本书编写的部门同事刘兴翬、何邱、邓智、谷明、李喆、李坤、袁春梅、唐明娟、李颖、陈建红、徐雅、曾泗维。

温素剑

2015 年 12 月 3 日

目录

第 1 章 性能测试基础.....	1
1.1 初识性能测试.....	1
1.1.1 性能测试是什么.....	1
1.1.2 性能测试的目的.....	2
1.1.3 性能测试的常见分类.....	2
1.1.4 性能测试的常见观察指标.....	3
1.1.5 性能测试的基本流程.....	3
1.2 开源 Web 性能测试.....	8
1.2.1 Web 性能测试的重要性.....	8
1.2.2 开源性能测试.....	8
1.2.3 开源性能测试的优势.....	9
1.3 本章小结.....	9
第 2 章 JMeter 基础知识.....	10
2.1 JMeter 简介.....	10
2.1.1 JMeter 主要特点.....	11
2.1.2 JMeter 常用术语.....	12
2.1.3 JMeter 测试结果字段的意义.....	12
2.2 JMeter 工作原理.....	13
2.3 JMeter 的安装.....	13
2.3.1 JMeter 安装配置要求.....	13
2.3.2 JMeter 安装.....	13

2.4	如何运行 JMeter	14
2.5	配置 JMeter	20
2.6	JMeter 与 LoadRunner 优缺点对比	21
2.7	本章小结	21
第 3 章	Web 性能测试脚本录制与开发	22
3.1	JMeter GUI 基本操作	22
3.2	JMeter 常用测试元件	25
3.3	JMeter 脚本开发基础	32
3.3.1	JMeter 执行顺序规则	32
3.3.2	作用域规则	33
3.3.3	JMeter 属性和变量	34
3.3.4	使用变量参数化测试	35
3.3.5	创建 Web 测试计划	35
3.4	录制 Web 测试脚本	41
3.4.1	使用代理录制 Web 性能测试脚本	41
3.4.2	使用 Badboy 录制 Web 性能测试脚本	46
3.5	创建高级 Web 测试计划	51
3.6	本章小结	52
第 4 章	数据库性能测试脚本开发	53
4.1	创建数据库测试计划	53
4.2	九步轻松搞定 Oracle 数据库性能测试	58
4.3	本章小结	63
第 5 章	FTP 性能测试脚本开发	64
5.1	FTP 是什么	64
5.2	创建 FTP 测试计划	68
5.3	本章小结	72
第 6 章	LDAP 性能测试脚本开发	73
6.1	LDAP 是什么	73
6.2	创建 LDAP 测试计划	84
6.3	LDAP 常见操作指南	89

6.4	创建扩展 LDAP 测试计划.....	91
6.5	本章小结	100
第 7 章	Web Service 性能测试脚本开发	101
7.1	Web Service 是什么	101
7.1.1	技术和规则.....	102
7.1.2	目标.....	102
7.2	创建 Web Service 测试计划.....	105
7.3	本章小结	108
第 8 章	JMS 性能测试脚本开发.....	109
8.1	JMS 是什么.....	109
8.1.1	定义.....	109
8.1.2	简介.....	109
8.1.3	历史.....	110
8.1.4	体系架构.....	110
8.1.5	JMS 模型	110
8.1.6	传递消息方式.....	111
8.1.7	JMS 应用程序接口	112
8.1.8	JMS 提供者实现	113
8.2	创建 JMS 点对点测试计划.....	113
8.3	创建 JMS Topic 测试计划.....	116
8.4	本章小结	120
第 9 章	服务器监控测试脚本开发	121
9.1	创建监控测试计划	121
9.2	本章小结	125
第 10 章	详解 JMeter 测试元件	126
10.1	详解 JMeter 监听器 (Listeners)	126
10.2	详解 JMeter 逻辑控制器.....	135
10.3	详解 JMeter 配置元件	151
10.4	详解 JMeter 断言	175
10.5	详解 JMeter 定时器	188

10.6	详解 JMeter 前置处理器	195
10.7	详解 JMeter 后置处理器	205
10.8	详解 JMeter 采样器	217
10.9	详解 JMeter 其他测试元件	268
10.10	本章小结	286
第 11 章	JMeter 进阶知识	287
11.1	详解 JMeter 函数和变量	287
11.2	详解 JMeter 正则表达式	310
11.3	详解 JMeter 远程测试	314
11.4	详解 JMeter 最佳实践经验	319
11.5	一些小技巧	324
11.6	本章小结	326
第 12 章	性能测试结果分析	327
12.1	如何分析性能测试结果	327
12.2	如何借助监听器发现性能缺陷	330
12.2.1	监听器——性能测试分析的基石	330
12.2.2	巧用监听器——识别性能缺陷	354
12.3	借助 Ant 实现批量测试和报表生成	361
12.4	本章小结	363
第 13 章	JMeter 性能测试实战——电话销售系统	364
13.1	测试背景和测试目标	364
13.2	分析确定性能测试指标	364
13.3	录制创建性能测试脚本	366
13.4	运行性能测试脚本	369
13.5	分析性能测试结果	371
13.6	上报性能测试缺陷	374
13.7	本章小结	375
第 14 章	Gatling 基础知识	376
14.1	Gatling 与 JMeter 优缺点对比	378
14.2	Gatling 快速上手	379

14.3	Gatling 进阶指南	386
14.4	本章小结	392
第 15 章	Gatling 详解	393
15.1	Gatling 概念	393
15.2	Gatling 目录结构	395
15.3	Gatling 安装细节与如何适配操作系统	396
15.4	Gatling 配置	397
15.5	模拟器结构	399
15.6	模拟器初始化	401
15.7	场景	403
15.8	断言	409
15.9	报告	411
15.10	本章小结	416
第 16 章	Gatling 会话	417
16.1	Session API	417
16.2	EL 和表达式	419
16.3	填充器	420
16.4	验证	424
16.5	本章小结	426
第 17 章	Gatling HTTP 协议	427
17.1	HTTP 协议	427
17.2	HTTP 请求	433
17.3	HTTP 头	435
17.4	HTTP 检查点	442
17.5	SSL	447
17.6	HTTP 助手	448
17.7	HTTP 录制器	449
17.8	WebSocket	454
17.9	本章小结	458

第 18 章	Gatling JMS	459
18.1	JMS 协议	459
18.2	本章小结	461
第 19 章	Gatling 扩展	462
19.1	Graphite	462
19.2	SBT 插件	465
19.3	Maven 插件	465
19.4	Jenkins 插件	466
19.5	VTD XML	466
19.6	第三方插件	466
19.7	性能扩展	467
19.8	参数传递	467
19.9	处理 JSF	467
19.10	本章小结	469
第 20 章	Gatling 性能测试实战——搜索引擎	470
20.1	测试目标	470
20.2	构建 Gatling 场景	471
20.3	运行 Gatling 场景，分析结果	477
20.4	本章小结	479

第 1 章

性能测试基础

1.1 初识性能测试

1.1.1 性能测试是什么

关于“性能测试是什么？”这个问题，真可谓众说纷纭，我们先不妨博览众家之言，以期获得一个全面的认识。

性能测试是通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。负载测试和压力测试都属于性能测试，两者可以结合进行。通过负载测试，确定在各种工作负载下系统的性能，目标是测试当负载逐渐增加时，系统各项性能指标的变化情况。压力测试是通过确定一个系统的瓶颈或者不能接受的性能点，来获得系统能提供的最大服务级别的测试（图 1-1 是一个典型的性能测试曲线）。

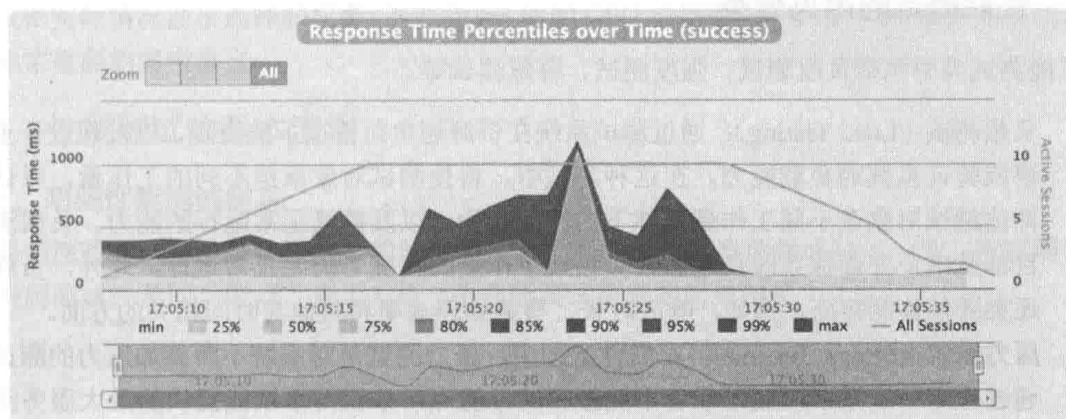


图 1-1 典型的性能测试曲线

以上是百度词条对性能测试的定义，下面再让我们看看维基百科对于性能测试的定义：

在计算机领域，软件性能测试被用来判断计算机、网络、软件程序或者驱动的速度和效率。这一过程会在同一实验环境下进行大量测试，以便衡量系统功能的响应时长或者 MIPS（每秒执行指令数目）等指标。其他系统特性，例如可靠性、可量测性、互用性，也可以用性能测试来衡量。性能测试通常与压力测试一起进行。

读者朋友应该可以发现以上两种定义的细微差别，百度词条认为性能测试由负载测试和压力测试构成，而维基百科定义的性能测试几乎等同于负载测试。作者更认同百度词条描述的性能测试定义，如果读者朋友有不同的看法，也没什么关系，毕竟本书不是一本深奥的测试理论书籍，而是一本测试实战类书籍。

1.1.2 性能测试的目的

性能测试的目的是验证软件系统是否能够达到用户提出的性能指标，同时发现软件系统中存在的性能瓶颈，进而优化软件，最后起到优化系统的目的。测试目的包括以下几个方面。

- 评估系统的能力，测试中得到的负荷和响应时长数据可以被用于验证所计划模型的能力，并帮助做出决策。
- 识别体系中的弱点：受控的负荷可以被增加到一个极端的水平，并突破它，从而修复体系的瓶颈或薄弱的地方。
- 系统调优：重复运行测试，验证调整系统的活动得到了预期的结果，从而改进性能。
- 检测软件中的问题：长时间的测试执行可导致程序发生由于内存泄露引起的失败，揭示程序中隐含的问题或冲突。
- 验证稳定性（resilience）和可靠性（reliability）：在一个生产负荷下执行测试一定的时间是评估系统稳定性和可靠性是否满足要求的唯一方法。

1.1.3 性能测试的常见分类

性能测试类型包括负载测试、强度测试、容量测试等。

- 负载测试（Load Testing）：通过测试系统在资源超负荷情况下的表现，以发现设计上的错误或验证系统的负载能力。在这种测试中，将使测试对象承担不同的工作量，以评测和评估测试对象在不同工作量条件下的性能行为，以及持续正常运行的能力。负载测试的目标是确定并确保系统在超出最大预期工作量的情况下仍能正常运行。此外，负载测试还要评估性能特征。例如，响应时长、事务处理速率和其他与时间相关的方面。
- 压力测试（Stress Testing）：在软件工程中，压力测试是对系统不断施加压力的测试，是通过确定一个系统的瓶颈或者不能接收的性能点，来获得系统能提供的最大服务级别的测试。例如，测试一个 Web 站点在大量的负荷下，何时系统的响应会退化或失败。

- 容量测试 (Volume Testing): 确定系统可处理同时在线的最大用户数。

1.1.4 性能测试的常见观察指标

在实际工作中我们经常会对两种架构的软件进行测试: B/S 和 C/S。它们关注的指标有哪些区别呢?

B/S 架构的软件一般会关注如下 Web 服务器性能指标。

- Avg Rps: 平均每秒响应次数=总请求时间/秒数。
- Avg time to last byte per terstion (mstes): 平均每秒业务脚本的迭代次数。
- Successful Rounds: 成功的请求。
- Failed Rounds: 失败的请求。
- Successful Hits: 成功的单击次数。
- Failed Hits: 失败的单击次数。
- Hits Per Second: 每秒单击次数。
- Successful Hits Per Second: 每秒成功的单击次数。
- Failed Hits Per Second: 每秒失败的单击次数。
- Attempted Connections: 尝试连接数。
- Throughput: 吞吐率。

对于 C/S 架构软件, 由于软件后台通常为数据库, 所以我们更注重数据库的测试指标。

- User Connections: 用户连接数, 也就是数据库的连接数量。
- Number of deadlocks: 数据库死锁。
- Butter Cache hit: 数据库 Cache 的命中情况。

请读者朋友们注意: 在实际性能测试过程中, 需要观察的性能指标并不限于以上提到的这些。需要根据实际情况做出选择和权衡, 有些指标(例如 CPU 占用率、内存占用率、数据库连接池等)也有非常重要的参考意义。

1.1.5 性能测试的基本流程

1. 明确性能测试需求

明确性能测试需求是性能测试的第一步, 有了好的开始, 事情就成功了一半。性能测试需求应该明确测试涉及的功能点。表 1-1 是一个性能测试需求的简单例子。

表 1-1 性能测试需求示例

测试需求名称	描 述
性能测试——简单查询	【参与者】：无 【概述】：测试简单查询性能 【前置条件】：无 【后置条件】：无 【业务数据】：无 【不可测试原因】：无 【流程规则】：无 【业务规则】： 涉及 UC： UC-XX-XX-B-021 快速查询功能 UC-XX-XX-B-024 下一任务功能 UC-XX-XX-B-038 系统操作历史显示 UC-XX-XX-B-039 客户接触历史显示 UC-XX-XX-B-043 内呼比对查询 UC-XX-XX-B-049 投保单查询 UC-XX-B-009 活动列表（有效） UC-XX-B-010 活动列表（其他状态） 性能指标参见测试方案 【页面规则】：无 【特殊规则】：无 【接口规则】：无 【检查内容】： 简单查询性能是否满足要求

2. 制定性能测试方案

性能测试方案应该详尽地描述如何进行性能测试，其中应该至少包括：

- 测试背景
- 测试目的
- 测试范围
- 测试进入条件
- 测试退出条件
- 测试指标要求

- 测试策略
- 测试时机
- 测试风险
- 测试资源

其中，测试范围、测试进入条件、测试退出条件、测试策略、测试风险、测试资源尤为重要。下面是一个性能测试方案的简单例子。

测试进入条件

- 不遗留 L1 的缺陷。
- 性能测试数据准备完毕。
- 系统功能测试已结束。

测试退出条件

- 各场景执行时间达到测试场景要求。
- 系统出现大量错误，暂停执行性能测试。

测试指标要求

- 关键功能的响应时长和用户数如表 1-2 所示。

表 1-2 性能测试指标要求示例

关键功能	平均使用次数	平均用户数	高峰段用户数	平均响应时长	可接受最长响应时长	使用时间段
xxx 功能		20000	23000	<4 s	≤7 s	8:00—21:00
xxx 功能		20000	23000	<3 s	≤6 s	8:00—21:00
xxx 功能		20000	23000	<3 s	≤6 s	8:00—21:00
xxx 功能		20000	23000	<2 s	≤4 s	8:00—21:00
xxx 功能		20000	23000	<6 s	≤15 s	8:00—21:00
xxx 功能		20000	23000	<3 s	≤6 s	8:00—21:00

测试策略

测试环境指标折算如下。

$$\text{测试环境平均并发数} = (\text{最大在线人数} \times 10\%) / n$$

n 是生产环境和测试环境服务器配置折算比，例如， $n = \text{公倍数}((\text{生产 Web 服务器数} / \text{测试 Web 服务器数}), (\text{生产 APP 服务器数} / \text{测试 APP 服务器数})) \times (\text{生产服务器内存} / \text{测试服务器内存})$ ，一般 $n = 4$ 。