

# LUPA 简介

LUPA 是开源高校推进联盟 (Leadership Of Open Source University Promotion Alliance) 的英文缩写, 于 2005 年 6 月 12 日在杭州成立。

LUPA 是中国开源运动的探索者和实践者, 也是“中国开源模式”的缔造者。LUPA 是中国开源软件推进联盟 OSS 的成员单位, 中国 Linux 产业战略联盟的核心发起单位。

LUPA 主张软件自主创新, 围绕学生“就业与创业”搭建起学校与企业沟通的桥梁。给在校学生或社会群体提供一个直接与产业对话的平台, LUPA 融合国际最新前沿技术, 打造新型、实用的标准化课件, 促进中国高校教学教程改革, 扶持高校学生自主创业和灵活就业, 是解决目前我国“就业瓶颈”的理想模式。

LUPA 的宗旨是充分发挥其在政府、企业与院校之间的纽带作用, 团结和协调各方力量, 维护国家信息安全, 发展民族信息产业, 提升我国开源软件产业的核心竞争力; 推进我国开源教育课程系统体系, 致力于开源人才的培养和开源技术在高校的推广、应用; 打造开源社区, 使在校学生与开源产业直接对话; 促进中日韩 (CJK) 及世界各国青年开源技术的合作对话, 为开源运动的推广树立崭新的模式。

目前 LUPA 已有清华大学、北京大学、浙江大学等 300 余所大专院校加盟, 成为国内规模最大、拥有院校资源最多、影响最为广泛的开源社团组织。LUPA 的推进模式, 获得了政府和国内外开源社区以及软件企业的充分肯定和重视, 已成为中国开源软件发展与普及的生力军, 在我国开源事业中发挥着越来越显著的作用。

开源是一种精神、开源是一种理念、开源是民族产业的未来。

# 序 言



LAMP 平台包括: Linux 操作系统、Apache 服务器、MySQL 数据库和 PHP 编程语言,这一新生事物已成为全球开源革命浪潮的前沿技术。它与 .NET、JAVA 商业软件形成三足鼎立之势,受到整个 IT 界的关注。越来越多的供应商、用户和企业投资者日益认识到,经过 LAMP 平台用来构建以及运行各种商业应用、协作和构建各种网络应用程序变为一种可能和实践,变得更加具有竞争力,更加吸引客户。LAMP 无论是性能、质量还是价格都将成为企业、政府信息化所必须考虑的平台。

中国政府及企业信息化,对于软件供应商而言是个巨大的市场,而 LAMP 所具有的简易性、低成本、执行灵活和应用广泛等特点使其得以在全球快速发展。越来越多的企业将平台架构在 LAMP 之上,促使了 LAMP 架构的崛起,这一信息化技术的“冲击波”同样也波及到微软 .NET 和一些 J2EE 厂商,将对全球以及中国信息化的发展带来机遇。并且对我国 IT 企业提供了新的商机,对就业、创业提供了新的方向和岗位。

LAMP 这一新技术的出现,为人们提供更多的商业模式。技术、商业模式、观念几个变量的发展及排列组合变换出了更多的商业机会。几乎每个时代都会有人感慨自己生不逢时,没有赶上前人暴富或者成功的机会。的确,大航海时代、美国西部淘金热,以及我国改革开放,股票市场,互联网神话,房地产热,我们都错过了机会。大可不必感慨这些发财成功的机会的逝去,倒是世界上一个全新的商机摆在了我们的面前,应该迅速作出反应,才是把握了人生的机会。

LAMP 是开源技术架构,而开源的核心内容在于资源共享。IT 人曾经在软件这个新兴行业大出风头,制造出互联网时代的神话,那是一个以信息科技为主导的时代。而今天世界将进入一个全新的时代,那就是以开源为核心的资源科技时代,Resource Technology, RT 时代。旧日里风光无限的 IT 人将被具有开



源理念、开源思想和 LAMP 类技术能力的 RT 人所代替。这对目前全世界特别是中国进入产能过剩的情况下,必定是个可供选择的道路。

学习 LAMP 架构,就像在将自己培养成一名厨师,这将是一种新的理念。目前,国内企业没有掌握源技术并不可怕,就像一个好的厨师不一定要从种菜开始。我们相信,本书将因其思想上的先进性、技术上的前沿性、内容上的通俗实用性与贴近实际可操作性而受到读者的广泛欢迎。并对 LAMP 技术的普及推广,以及我国信息化发展起到推进作用。为此,谨在此对参与本书编著的各位领导、专家学者以及开源运动的倡导与实践者深表敬意!

LAMP,正像一盏开源明灯指引我们前进的方向!

开放源代码高校推进联盟(LUPA)主席

中国开源软件(OSS)推进联盟副主席

浙江省 Linux 专业委员会主任

浙江省开放源代码公共服务中心主任

张建华

# 前 言

LAMP 是 Linux、Apache、MySQL、PHP/Perl/Python 的简称。1998 年 Michael Kunze 在电脑杂志 C'T 撰写的文章中首次使用了缩略语 LAMP。它所组成的各组件都是开源软件,能被很方便自由地获取,导致这些组件的广泛使用,并不断完善发展,初步形成新的软件系统格局,并可能成为开源软件系统工程化的核心模式。

本书的目的是教你如何使用 LAMP 技术来安装和设置网络服务器。当读完本书后,你将拥有一个强力而高效的服务器来支持你的个人网页、强大的电子商务或是商业网站以及其他任何网络需求。

本书所使用的软件完全是免费的,不需要任何费用和注册,除非你自己选择购买光碟。此外,所有的软件可以非常简单地通过互联网下载。这里要说明的是,开源软件不是免费软件。这里免费的概念是你可以自由获得软件的源代码、自由修改、再发布。当然前提是你必须遵守软件的许可证。如 Linux,遵守 GPL v2.0 许可证。

## LAMP 的组成

LAMP 具有简易性、低成本和执行灵活等特点,使其成为业内发展最快,应用最广的服务器系统。LAMP 架构的崛起,与 J2EE 架构和 .Net 架构形成了三足鼎立的竞争态势。通过证明,LAMP 是一组高效的软件,作为一个系统能够良好的运行。每个组成元素的开放式结构允许相互间顺畅而缜密的结合,从而形成了一个强大的组合。回到 1997 年,那些早期的 LAMP 技术的采纳者被认为是激进的,但在今天开源运动正在蓬勃发展,无论是大企业还是小公司都在采用 LAMP 的开发方式,因为 LAMP 的每一个应用程序的稳定性已经超过了那些比它昂贵许多的软件兄弟们。

### 1. Linux 操作系统



Linux 是一种计算机操作系统内核,基于 GNU GPL v2 许可证下发行,它具有性能好、安全性高、开放自由特点,你可以直接从网上下载得到并安装它。Linux 能够根据你的需要而决定是否运行图形化用户界面。(没有图形化用户界面的运行方式对于那些对 Linux 不太熟悉的 Windows 使用者来说就相当于运行 DOS 模式)

操作系统最重要的组成部分是内核。在 GNU/Linux 系统中,Linux 就是内核组件。而

该系统的其余部分主要是由 GNU 工程编写和提供的程序组成。因为单独的 Linux 内核并不能成为一个可以正常工作的操作系统,现在人们接触到的各种各样的 Linux 发行版,包含大量的 GNU 工程软件,包括了一个 shell 程序、工具、程序库、编译器及开发工具,还有许多其他程序,比如软件开发工具、数据库、Web 服务器(例如 Apache)、X Window、桌面环境(比如 GNOME 和 KDE)、办公套件(比如 OpenOffice.org)等等。所以我们更倾向使用 GNU/Linux 一词来表达人们通常所说的 Linux。

Linux 的创始人是 Linus Torvalds,当时他是芬兰赫尔辛基大学的学生。他的目的是想设计一个代替 Minix 的操作系统,这个操作系统可用于 386、486 或奔腾处理器的个人计算机上,并且具有 Unix 操作系统的全部功能,因而开始了 Linux 雏形的设计。1991 年 9 月 Linus 在网上发布 Linux 0.01 版,1994 年 3 月 Linux 内核 1.0 版问世。由于 Linux 的源代码是开放的,因此受到了全世界开发者的广泛支持,发展速度非常快,形成了以社区为中心的开发模式。就是人们通常说的开源社区,这种模式已经得到了充分肯定,越来越多的人参与到开源社区中来,使得 Linux 的发展越来越快,目前 Linux 的内核已经发展到 2.6 版本。

现在, Linux 已经成为了一种受到广泛关注和支持的操作系统。包括 IBM 和惠普在内的一些计算机业巨头也开始支持 Linux。很多人认为,和其他商用 Unix 系统以及微软 Windows 相比,作为自由软件的 Linux 具有低成本,安全性高,更加可信赖的优势。

## 2. Apache Web 服务器



Apache, 一种开放源代码的 HTTP 服务器,可以在大多数计算机操作系统中运行,由于它多平台和安全性,被广泛使用,成为最流行的 Web 服务器端软件之一。它快速、可靠并且可通过简单的 API 扩展,将 Perl/Python 等解释器编译到服务器中,得到了用户的广泛认可。

Apache 起初由伊利诺伊大学香槟分校的国家高级计算程序中心开发。此后, Apache 被开放源代码团体的成员不断的发展和加强。Apache 服务器凭借其牢靠可信的美誉,在超过半数的因特网 Web 服务器中得到应用。

开始, Apache 只是 Netscape 网页服务器(现在是 Sun ONE)之外的开放源代码选择。渐渐的,它开始在功能和速度超越其他基于 Unix 的 HTTP 服务器。1996 年 4 月以来, Apache 一直是 Internet 上最流行的 HTTP 服务器;1999 年 5 月它在 57% 的网页服务器上运行;到了 2005 年 7 月这个比例上升到了 69%。

Apache 支持许多特性,大部分通过编译的模块实现。这些特性从服务器端的编程语言支持到身份认证方案。一些通用的语言接口支持 Perl、Python、TCL 和 PHP 流行的认证模块包括 mod\_access, mod\_auth 和 mod\_digest。其他的例子有 SSL 和 TLS 支持(mod\_ssl), proxy 模块,很有用的 URL 重写(由 mod\_rewrite 实现),定制日志文件(mod\_log\_config),以及过滤支持(mod\_include 和 mod\_ext\_filter)。Apache 日志可以通过网页浏览器使用免费的脚本 AWStats 或 Visitors 来进行分析。

### 3. MySQL 数据服务器



MySQL 是一个开放源码的多用户、多线程 SQL 数据库服务器软件。开发者为瑞典 MySQL AB 公司。它能让你通过一种编写语言如 PHP 来存储和找回数据。可以快速而高效的存储多种类型的数据,如布尔类型、文本类型、整数类型、图像类型、二进制数据和 BLOB 数据。使用数据库对于创建动态网站是十分重要的。动态网站这一概念来自于基于用户互动基础上能够使用单页代码而显示不同信息。如果不使用数据库和编写语言如 PHP 来操控数据,这一切听起来都是不可能的。

MySQL 具有诸多特性,如数据复制、表格锁定、询问限制、用户账号、多层数据库、持续连接以及 MySQL5 的存储过程、触发器和视图。这些特性在接下来都会被更具体的介绍,而现在你应该知道使用这一伟大的数据库管理器你将收益颇丰。

目前 MySQL 被广泛地应用在 Internet 上的中小型网站中。由于其体积小、速度快、总体拥有成本低,尤其是开放源码这一特点,许多中小型网站为了降低网站总体拥有成本而选择了 MySQL 作为网站数据,对于中小型应用系统是非常理想的。MySQL 支持标准的 ANSI SQL 语句,它还支持多种平台,在 Unix 系统上该软件支持多线程运行方式,从而能获得相当好的性能。它同时支持 Linux、Windows、Solaris 等主流操作系统。

### 4. PHP 脚本语言



PHP 是一种流行的开放源代码的编程语言,主要用于开发服务器端应用程序及动态网页。PHP 原始的缩写是“Personal HomePage”,现在官方正式定为“PHP: Hypertext Preprocessor”的递归缩写。PHP 程序是开放源代码最流行的一种脚本语言,可以用于替代微软的 ASP 体系、Sun 微系统公司的 JSP/Java 体系,以及 CGI/Perl 等。它是一种嵌入 HTML 页面中的脚本语言。PHP 在 Web 服务器上运行。当 PHP 脚本被客户端请求时,被请求的程序开始执行,并把执行的结果返回给客户端的网页浏览器。发送给客户端浏览器的内容是普通的 HTML 文本,不包含 PHP 代码。这是与嵌入 HTML 的客户端脚本的最主要的区别。在有了 PHP-GTK 扩展的支持后,现在的 PHP 已经可以被用来编写窗口程序了,PHP4 版本以后,PHP 也可以用来编写后台 SHELL 脚本程序,甚至有用 PHP 程序编写 Web 服务器。

PHP 最初在 1995 年出现仅仅是一组简单的 Perl 语句,用来追踪 Rasmus Lerdorf 的在线简历。随着时间的推移,Lerdorf 开始编写更大的 C 语言程序以适应他日益扩大的对功能的要求,包括数据库的连接性问题。Lerdorf 接下来决定发布它的最初版本,是开源类型的 PHP/FI 以让所用人使用和改进。在当时,PHP/FI 代表的是个人主页/表格编译器。到了 1997 年,第二个版本(PHP/FI2.0)发布并开始吸引全球几千人的追随。尽管许多个人都对代码做出了贡献,但是 Lerdorf 仍然是所有开发的主要贡献者。

1997 年目睹了一个新的 PHP 时代的到来:PHP3。这一版本是由 Andi Gutmans 和 Zeev Suraski 对 PHP/FI2.0 的完全重写,他们不满足于之前他们所工作的大学一个项目上 PHP 所提供的功能。为了对 PHP 日益成长的用户群进行资本化,Lerdorf、Gutmans 和 Suraski 决

定以 PHP 的名字发布这一新版本,从而开始了我们今天所熟知的 PHP。在 1998 年冬天,PHP4 的开发由 Gutmans 和 Suraski 进行。他们在 2000 年 5 月公布了第一个官方版本。PHP4 具有更高超的表现力并向它日益扩大的“粉丝”群们推广包括 HTTP 环节、输出缓冲和更安全的处理用户输入方式在内的新技术。

我们相信 PHP5 将会创造出更大的效应。一个新的以目标为本并与 Zend Engine2、堆栈跟踪和例外处理相结合的模型被寄予厚望,并成为 PHP 在全球发展的动力。与此同时,引进同外部以目标为本的模型结合,例如 COM 和 Java,将会是向混合中投入一个通配符,这将使其他社区有能力与 PHP 实现无缝结合成为现实。这表明以前所写的 APIs 将能够更轻松的与 PHP 相结合,从而消除了其他社区不使用 PHP 的最后理由。

## 为什么要使用 LAMP

Linux 操作系统的核心是基于 GNU GPL v2 发行的。Linux 在 GPL 下注册的原因很简单:你被授权对软件进行修改,相应地你就要将你的版本公之于众,并随之公布源代码。这就保证了 GPL 下的“开源”,允许其他人对你所做的工作进行修改。

尽管并没有必要在 GPL 下注册任何开源软件,但 GPL 确实防止了代码不向公众公开的行为。任何人不能够通过改进一项软件之后关闭它的源代码,使得开源社区在项目开发上,从最初的源代码中获利。也是诸如 Linux 和大多数开源软件等伟大的操作系统发展的方式。这种类型的同级评审也正是开源运动的基础。

GPL 是由免费软件基金会编写和维护的。如果你对 GPL 感兴趣的话,可以访问他们的网站:<http://www.gnu.org/copy/gpl.html>。

Apache、MySQL 和 PHP 是本书中我们将涉及和使用的免费应用软件。尽管它们其中有些并没有在 GPL 下注册,但它们也同样都处于一个相似的许可证之下。这不仅使得我们能够共享全球无数开发者的工作,而且也使得我们能够自由地选择使用和传播这些应用软件。从我们基本的 Linux 操作系统到我们的邮件服务器,我们将会使用由那些喜欢做一件事情的人带给我们的免费软件,而那些人爱做的事就是:编写软件。

组成 LAMP 的各个组件,都是开源软件中的最为出色的项目,这些组件完全是开源的,所以 LAMP 完全是开源的。LAMP 代表着当今人类对科技发展的一种新的态度,新的理想,代表着科技发展的一种新的模式,一种新的文化理念。全球财富 500 强企业中,有 70% 的企业采用 Linux 承担企业核心业务;全球半数以上的互联网服务器采用开源软件。这里所涉及到的开源软件产品包括:Linux 操作系统(68%)、Apache(67%)、PHP 开源脚本语言(53%)和开源数据 MySQL(52%)。这些数据是在 2005 年 5~6 月期间,美国国家计算中心调查了 140 多个公司的 IT 高管人员后所得,作为开源软件组合 LAMP 已被称为开源软件中的启明之灯。

## 本书概况

《LAMP 从入门到精通》注重实际应用,重点介绍了 LAMP 平台的架构及基于 LAMP 平台的企业信息化解决方案。学习本书可以从事基于开源软件的信息化平台架构工作。本书是 LUPA 开源教育职业资格考试用书。

本书适合于从事 Linux 平台上 Web 服务、数据库服务的从业资格认证考试读者,也可以作为大学本专科计算机专业学生学习 LAMP 平台的教材及 LAMP 平台管理人员的参考用书,当然对于那些初学者,这也是一本不错的 LAMP 入门教程。

本书共分为 6 章,内容简单扼要,分别介绍了 Linux 安装使用、Apache 安装使用、MySQL 安装使用、LAMP 平台的性能优化、开源项目的调试和配置。

本书的作者对 LAMP 平台有着丰富的研究和经验,本书由 LUPA 主席、中国 OSS 副主席张建华主编、黄国荣为副主编,秦曦、李震、邵炜、周瑞星、钱旭伟等老师编写。由于时间仓促及作者的水平有限,书中难免存在疏漏有不妥之外,敬请广大读者批评和指正。

<b>第 1 章 Linux 安装与使用</b> .....	1
第 1 节 Linux 安装 .....	1
第 2 节 Linux 的使用 .....	12
第 3 节 安 全 .....	42
<b>第 2 章 Apache 安装和使用</b> .....	47
第 1 节 Apache1.3 与 Apache2.0 .....	47
第 2 节 Apache2.0 模块介绍 .....	48
第 3 节 Apache 的编译与安装 .....	51
第 4 节 Apache 的启动和运行 .....	53
第 5 节 了解 httpd.conf 文件 .....	54
第 6 节 使用 Apache 虚拟主机 .....	57
第 7 节 认证、授权和访问控制 .....	66
<b>第 3 章 MySQL 安装和使用</b> .....	71
第 1 节 下载 MySQL .....	71
第 2 节 安装 MySQL .....	71
第 3 节 配置 MySQL .....	75
第 4 节 MySQL 管理 .....	84
<b>第 4 章 PHP 安装和使用</b> .....	96
第 1 节 为什么要使用 PHP .....	96
第 2 节 PHP 版本 .....	96
第 3 节 PHP 开发工具 .....	97
第 4 节 安装 PHP .....	97
第 5 节 PHP 语法 .....	103
<b>第 5 章 优化测试 LAMP</b> .....	113
第 1 节 安装 Apache 虚拟主机 .....	113
第 2 节 准备 MySQL 数据库和用户 .....	114

---

第 3 节	测试 Apache, PHP 和 MySQL .....	114
第 4 节	Alternative PHP Cache .....	116
第 5 节	eAccelerator .....	119
第 6 节	Zend Optimizer .....	120
第 7 节	JPCache .....	122
第 8 节	memcached .....	123
第 9 节	如何选用缓存引擎 .....	127
<b>第 6 章</b>	<b>基于 LAMP 实例配置 .....</b>	<b>128</b>
第 1 节	架设 BLOG 平台 .....	128
第 2 节	架设 Wiki 平台 .....	136
第 3 节	架设 CRM 平台 .....	139
第 4 节	架设 CMS 系统 .....	145
第 5 节	LAMP 上的开源项目 .....	150



# Linux 安装与使用

Linux 是 LAMP 平台最重要的组成部分,是 LAMP 的基础平台。所有的应用和组件都是运行在 Linux 操作系统平台之上,因此本章重点介绍 Linux 使用。通过本章学习,你可以理解 Linux 的文件结构、文件系统、常用命令、用户和组的管理、软件包管理、shell 环境、网络配置,以及常用工具的使用,如 vi 等。

## 第1节 Linux 安装

### 1. 选择使用哪个 Linux 版本

由于 Linux 内核代码的开放性,其发行版有 300 多个,本节主要介绍目前比较流行的几款发行版。

#### 1) Debian

Debian 是由 Ian Murdock 在 1993 年 8 月 16 号发布的,可以算是迄今为止,最遵循 GNU 规范的 Linux 系统。Debian 系统分为三个版本分支(branch): stable、testing 和 unstable。其强大的包管理系统使得它越来越受到人们的青睐。更多信息可以访问官方网页:<http://www.debian.org> 查询。

#### 2) Ubuntu

Ubuntu 是 Debian 的家族产品,除了继承了 Debian 稳定性的优点外,还包括了安装更加方便,桌面的人性化操作更强。并且每六个月更新一次,弥补了 Debian 更新缓慢的不足。Ubuntu 也是目前众多 Linux 发行版中的最优秀的成员之一,受到了越来越多用户的欢迎。更多信息可以访问官方网页:<http://www.ubuntulinux.org> 查询。

#### 3) SUSE

Novell SUSE 前身是德国最著名的 Linux 发行版,在全世界范围中也享有较高的声誉。SUSE 自主开发的软件包管理系统 YaST 也大受好评。SUSE 于 2003 年年末被 Novell 收购。更多信息可以访问官方网页:<http://www.suse.com/> 查询。

#### 4) Mandriva

Mandriva 原名 Mandrake, 最早由 Gael Duval 创建并在 1998 年 7 月发布。Mandrake 最早是基于 Redhat 进行开发的, Mandrake 的开发完全透明化, 包括“cooker”。当系统有了新的测试版本后, 便可以在 cooker 上找到。更多信息可以访问官方网站: <http://www.mandriva-linux.com/> 查询。

#### 5) RedHat

RedHat 是全世界的 Linux 用户所最熟悉的版本。RedHat 最早由 Bob Young 和 Marc Ewing 在 1995 年创建。现在 RedHat 公司已经把主要技术支持转向了 RedHat Enterprise Linux (RHEL, RedHat 的企业版), 正统的 RedHat 版本早已停止技术支持, 最后一版是 RedHat 9.0。于是, 目前 RedHat 分为两个系列: 由 RedHat 公司提供收费技术支持和更新的 RedHat Enterprise Linux, 以及由社区开发的免费的 Fedora Core。Fedora Core 第一个版本发布于 2003 年年末, 而 FC 的定位便是桌面用户。FC 提供了最新的软件包, 同时, 它的版本更新周期也非常短, 仅六个月。更多信息可以访问官方网站: <http://www.redhat.com/> 查询。

#### 6) 国内 Linux 发行版

中标普华 Linux 前身是中软 Linux, 是国内最大的 Linux 软件提供商之一。秉承人性化、实用化、效率化的设计理念, 在基本操作、菜单布局、功能模块设置等方面最大限度地保持了与 Windows 的一致性, 充分满足用户在使用习惯方面的需要。更多信息可以访问官方网站: <http://www.cs2c.com.cn> 查询。

中科红旗 Linux 是众多 Linux 发行套件之一。中科红旗是较早涉足 Linux 产业的国内公司, 也是国内最早推出 Linux 发行版本的公司之一。中科红旗有 2 种发行套件, 一种是面向企业用户的, 一种是面向桌面用户的, 面向桌面用户的可以免费下载。

此外国内还有 Turbo Linux、新华 Linux、共创 Linux、MagicLinux 等发行版。

本书采用 Fedora Core 5 作为本教材的示范系统。

## 2. 了解硬件需求

安装 Fedora Core 5 之前, 如果用户系统仅仅安装纯文本界面, 系统配置最低要求:

- ◆ 200MHz Pentium 处理器
- ◆ 2GB 硬盘空间
- ◆ 64MB 内存

支持图形界面最低配置:

- ◆ 400MHz Pentium II 处理器
- ◆ 3GB 硬盘空间
- ◆ 192MB 内存

### 3. 获得 Fedora Core 5

目前,除官方之外,各大社区都有高速镜像下载,读者可以根据自己情况选择最快的下载站点,本书推荐地址:<http://fedoraproject.org/wiki/Distribution/Download>(官方)

以及速度非常快的 LUPA 社区 Fedora 镜像站:<http://mirror.lupaworld.com>  
将链接里的 5 个 ISO 文件都下载下来。

### 4. 安装 Fedora Core 5

Linux 安装分为光盘安装、硬盘安装和网络安装,本书介绍光盘安装和硬盘安装。

#### 1) 光盘安装

光盘安装很简单,在电脑启动时进入 BIOS 设置,将启动项里的 CD-ROM 设置为第一启动,重启电脑即可以用光盘安装。具体步骤看后面的配图说明。

#### 2) 硬盘安装

对于硬盘安装,需要我们进行下列操作:(假设你已经装了 Windows 系统)

- ◆ 提取 `vmlinuz` 和 `initrd.img` 两个文件,它们在第一张 ISO 文件的 `isolinux` 目录下面,将其拷贝出来放在 C:\盘下。
- ◆ 接下来,在 `grub` 命令下执行,如果系统启动引导是 `grub`,则按 C 键进入控制。如果没有装 `grub`,可以去 <http://tech.lupaworld.com/lamp/> 下载 `grub for dos` 并安装。
- ◆ 重启电脑,进入 DOS,在此界面下启动 `grub.exe` 程序,在 `grub` 环境下依次输入:

```
Grub > root (hd0,0) //这里假设启动文件都放 C 盘。因此为(hd0,0)。如果是 D 盘,为(hd0,5),依次类推。也可以输入(hd0,后按 TAB 键查看分区信息。
```

```
Grub > kernel/vmlinuz
```

```
Grub > initrd/initrd.img
```

```
Grub > boot
```

- ◆ 启动之后会要求用户选择安装方法,我们选择 `hard drive` 即可,后面的安装方法和光盘安装一样。

#### 3) 图示安装过程

- 启动界面。这个界面只有用光盘安装的时候才会出现,直接回车跳过。当用户要对系统进行修复的时候,可以使用光盘,进入此界面,在 `boot` 后面输入 `linux rescue` 进入修复模式。如图 1-1。

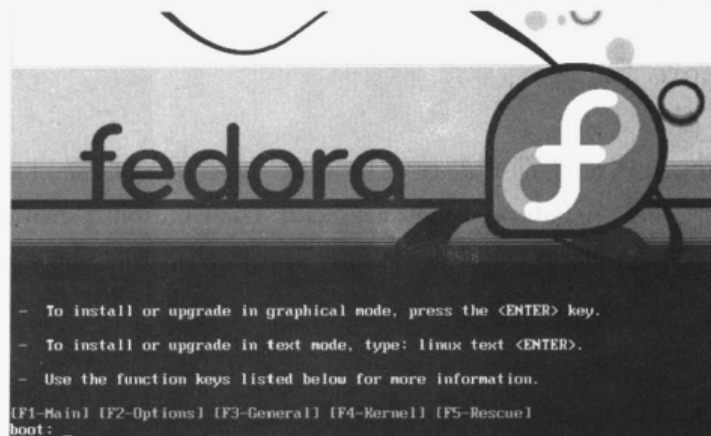


图 1-1

- 校验安装介质的完整性。如果确定自己的安装盘或者 ISO 文件完整,那么选择 Skip 跳过。如图 1-2。

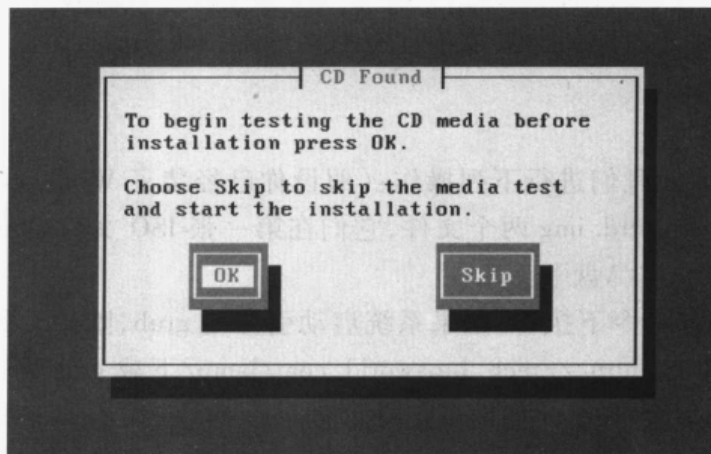


图 1-2

- 开始安装,点击下一步。如图 1-3。



图 1-3

- 选择安装语言。Linux 支持多个国家语言,这里我们选择简体中文。如图 1-4。

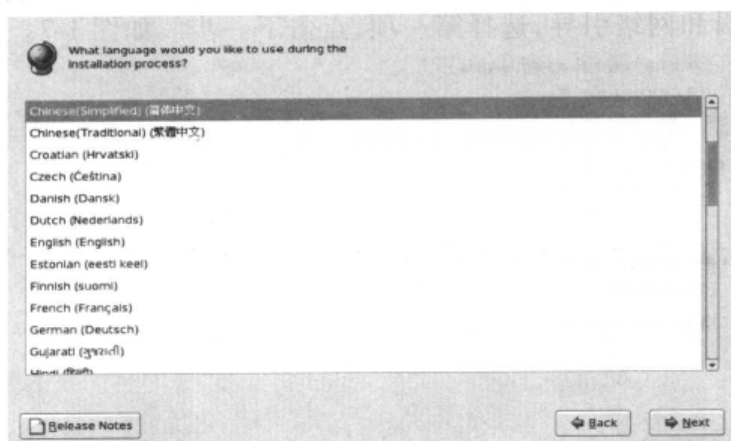


图 1-4

- 如果是初学者,建议选择 creat custom layout 自动创建 Linux 分区。对于经验丰富的用户,选择另外一个:手动分区。如图 1-5。

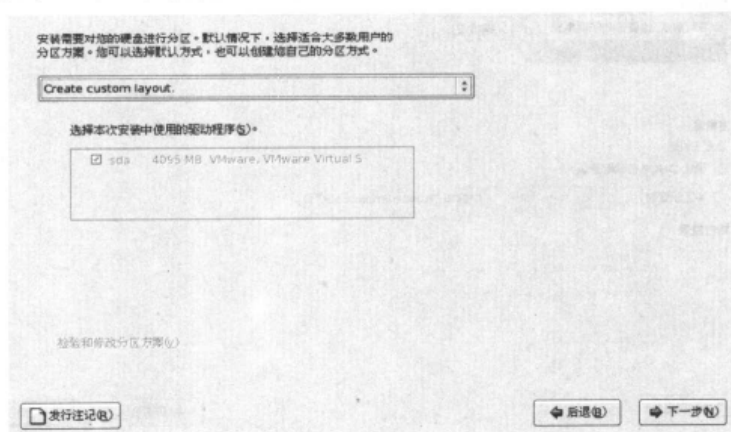


图 1-5

- 下面开始分区,对初学者来说最好创建一个 swap 分区,一般来说容量为内存的 2 倍或更高,再建立一个/根目录,建议分配 5G 以上空间。如图 1-6。

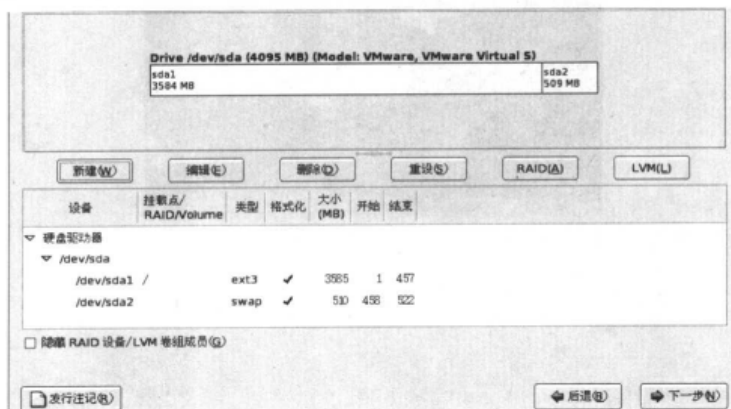


图 1-6

- 安装引导程序 grub。Linux 引导分为 grub 和 lilo, 目前用得最多的是 grub, 它支持交互性菜单编辑和网络引导, 选择第一项, 点击下一步。如图 1-7。

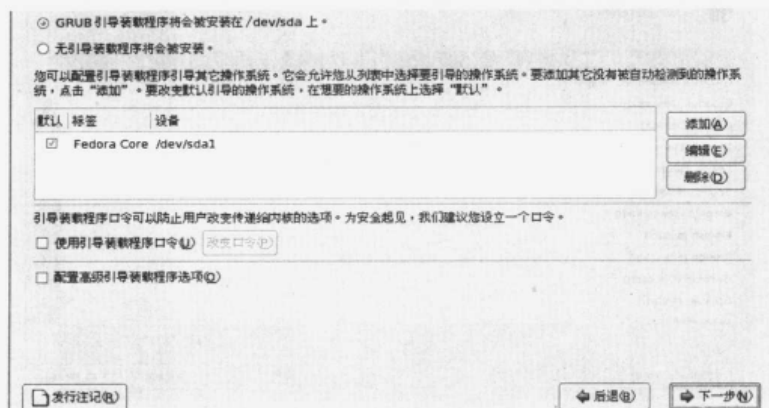


图 1-7

- 配置网络, ADSL 拨号或其他系统装好后再设置。这里我们都采用默认。如图 1-8。

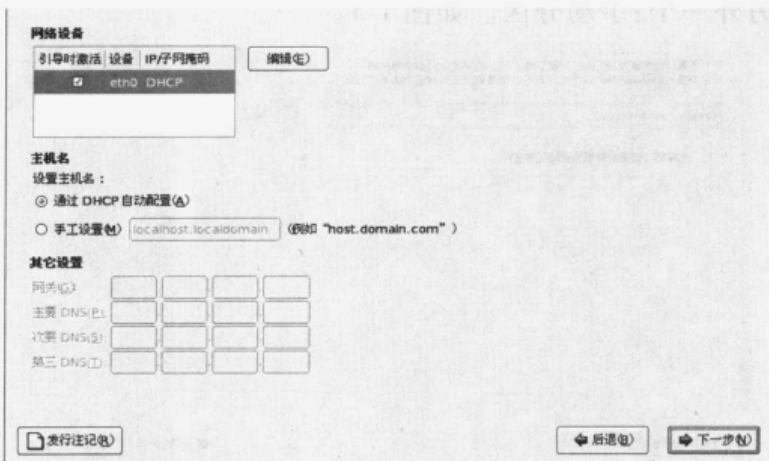


图 1-8

- 选择时区, 我们是亚洲/上海。如图 1-9。



图 1-9

- 设置 root 用户口令,也就是超级用户口令。这个口令对于用户非常重要,建议设置时不要过于简单,并牢记。如图 1-10。

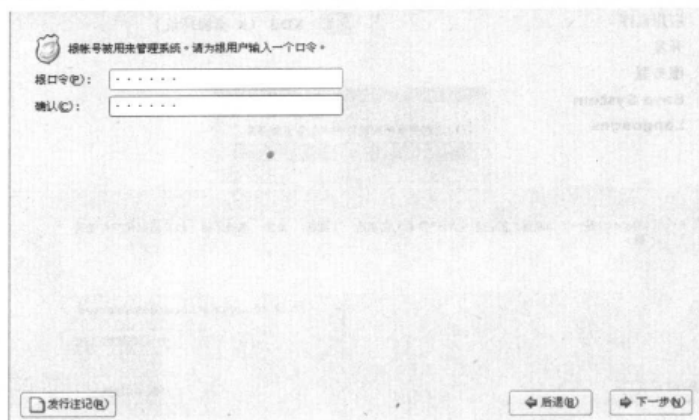


图 1-10

- 选择要安装的包。Linux 的安装定制性更强,它可以根据自己的需要选择安装包,默认选项里已经为我们提供了三种类型的安装,我们选择“定制”。如图 1-11。

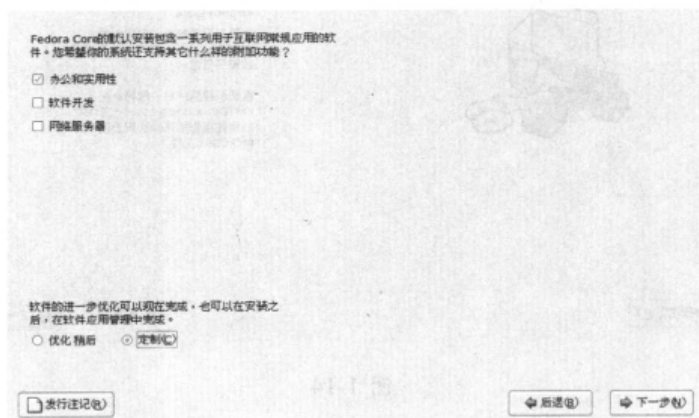


图 1-11

- 这一步你可以选择更详细的软件包。其中桌面环境有 GNOME 和 KDE 两种,选择其一。在服务器选项不安装任何软件包。如图 1-12。



图 1-12

- 包选择完成后,检验包的依赖关系。图 1-13。

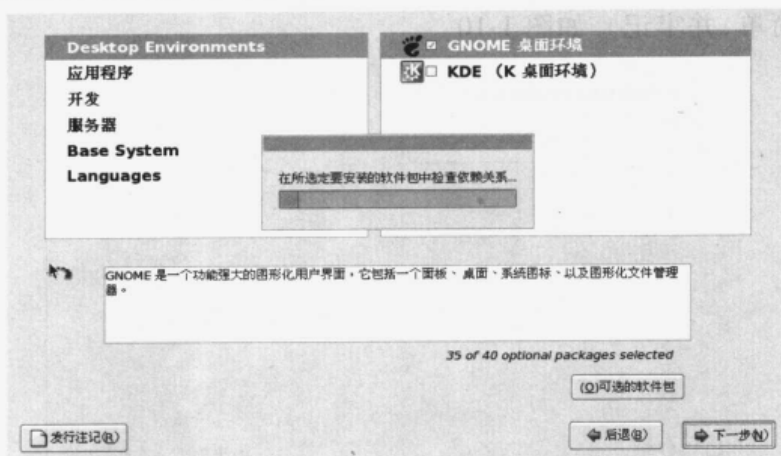


图 1-13

- 点击下一步,开始正式安装。如图 1-14。

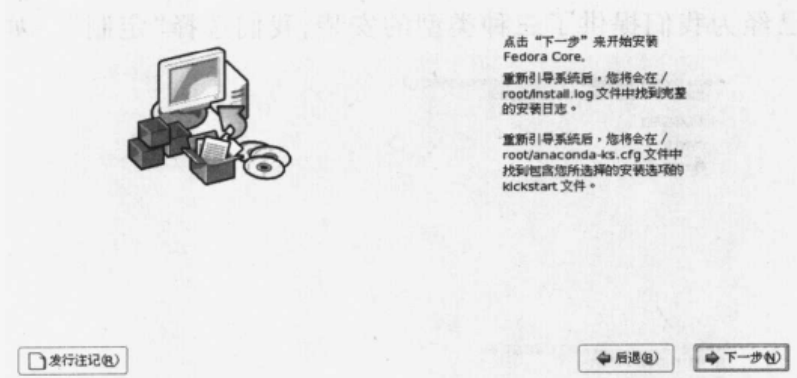


图 1-14

- 开始安装文件。如果是光盘安装,系统会提示更换光盘;如果是硬盘安装,我们不需要守候安装过程。如图 1-15。



图 1-15



- 根据硬件配置情况,安装时间大概需要 30 ~ 60 分钟,也可能会更长。安装完成后,重启系统。如图 1-16。

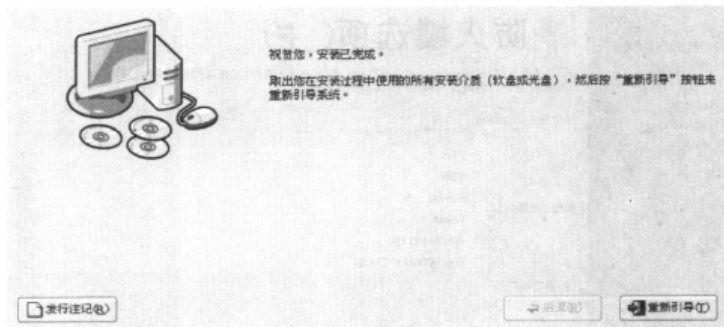


图 1-16

- 安装基本完成,接下来对系统进一步设置。如图 1-17。

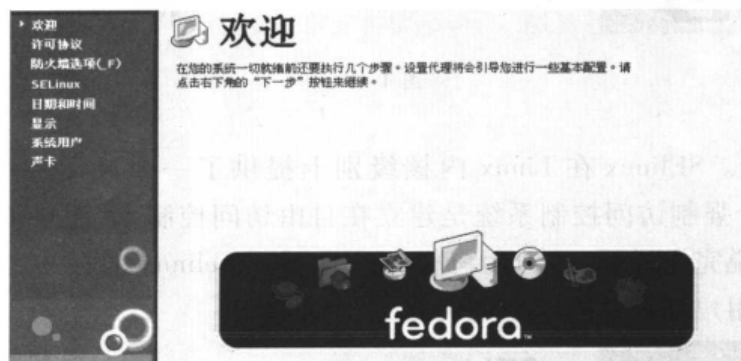


图 1-17

- 用户协议,同意并进入下一步。Fedora 是基于 GPL v2 许可证发布的,你必须接收 GPL 许可协议,才能使用这个系统。而对于用户来说,你接收了这个协议后,就意味着你可以自由获得其源代码,自由修改、传播这个系统。如图 1-18。

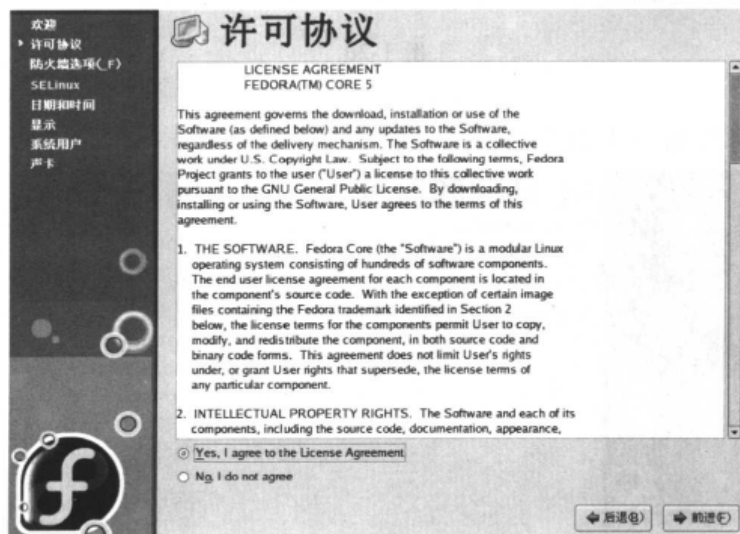


图 1-18

- 防火墙设置,根据你对系统安全的需要,选择是否开启这个功能。建议开启这个功能,这样系统会更加安全。如图 1-19。

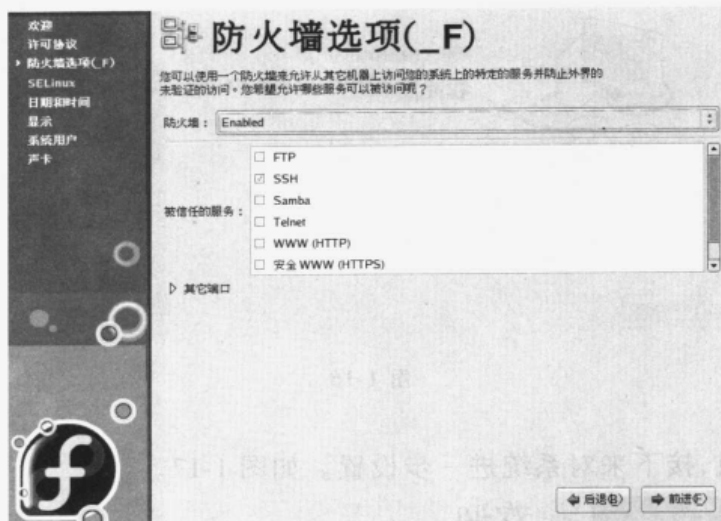


图 1-19

- SELinux 设置。SELinux 在 Linux 内核级别上提供了一个灵活的强制访问控制系统 (MAC),这个强制访问控制系统是建立在自由访问控制系统(DAC)之上的。SELinux 对于用户来说完全透明,普通用户根本感觉不到 Selinux 的存在,只有系统管理员才需要对这些用户环境及策略进行考虑。如图 1-20。

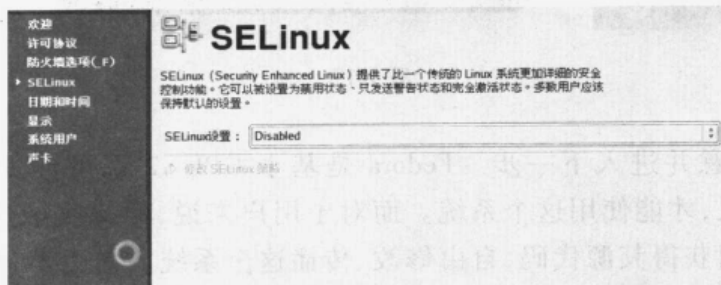


图 1-20

- 设定时间日期。如图 1-21。

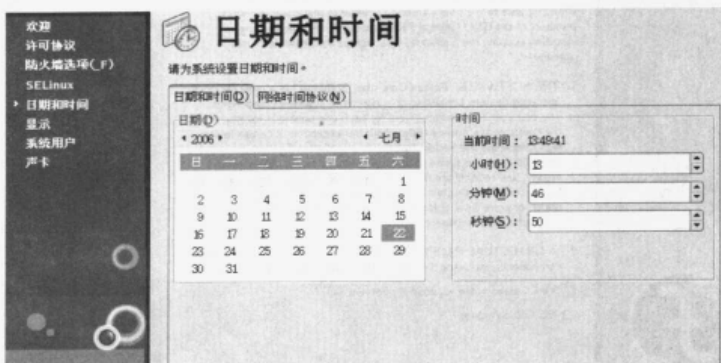


图 1-21

- 显示设置。一般情况下 Fedora 内置了显示驱动。如图 1-22。

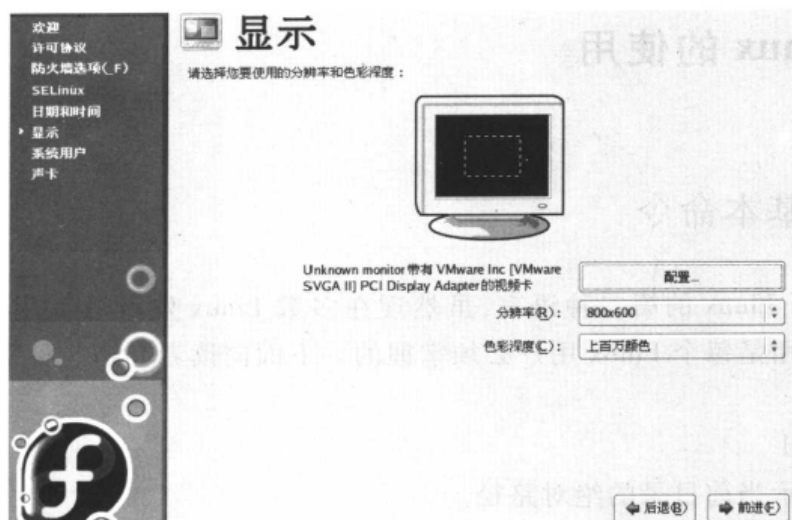


图 1-22

- 设置声卡。如图 1-23。

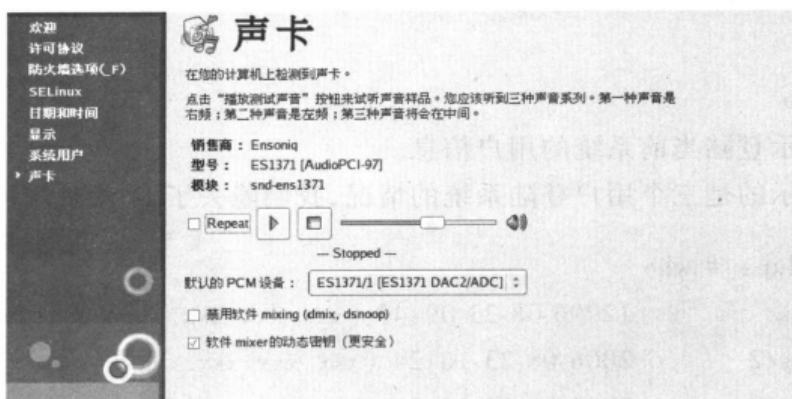


图 1-23

- 安装全部完成, 登录系统。如图 1-24。

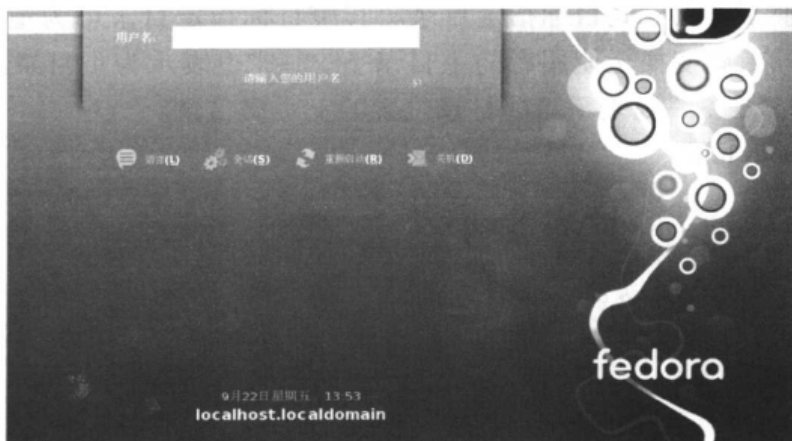


图 1-24。

## 第 2 节 Linux 的使用

### 1. Linux 常用基本命令

命令几乎成了 Linux 的第二种语言,虽然现在多数 Linux 发行版采用了图形化配置系统,但是命令的使用是每个 Linux 用户必须掌握的。下面向读者介绍几种常用的命令:

#### ● pwd

使用格式:pwd

功能说明:显示当前目录的绝对路径。

范例:显示当前的工作目录

```
[root@ lupa lupa]# pwd
/home/lupa
```

#### ● who

使用格式:who

功能说明:显示登陆当前系统的用户信息。

范例:下面显示的是三个用户登陆系统的情况,我们隐去了 IP 地址

```
[root@ lupa lupa]# who
root      pts/      12006-08-23 09:43 (xxx.xxx.xxx.xxx)
root      pts/2     2006-08-23 16:24 (xxx.xxx.xxx.xxx)
root      pts/3     2006-08-23 15:29 (xxx.xxx.xxx.xxx)
```

#### ● cd

使用格式:cd[目录]

功能说明:变换工作目录到目标指定目录。其中目标指定目录可为绝对路径或相对路径。若目录名称省略,则变换至使用者的主目录下,“~”也表示为主目录的意思,“.”则表示当前所在的目录,“..”则表示当前目录位置的上一层目录。

范例 1:跳到/usr/bin/

```
[root@ lupa /]# cd /usr/bin
```

范例 2:跳到自己的主目录(home 目录)

```
[root@ lupa /]# cd ~
```

范例 3:跳到当前目录的上一层

```
[root@ lupa /]# cd .. //注意 cd .. 之间有个空格;
```

范例 4:跳到当前目录的上上两层

```
[root@ lupa /]# cd ../../
```

### ● ls

使用格式:ls[选项][文件/目录]

功能说明:显示当前工作目录下的信息。不指定目录时,显示当前目录下的文件和子目录信息。

选项说明:

选项	说明
a	显示所有文档及目录。
l	除文件名称外,并且将文件类型、权限、拥有者、文件大小等信息详细列出。
t	将文件按照创建时间先后顺序列出。
R	列出目录下所有文件,而且还递归地显示各子目录中的文件和子目录信息。

范例 1:将/bin 目录以下所有目录及文件详细资料列出

```
[root@ lupa /]# ls-lR /bin
```

范例 2:列出当前工作目录下所有文件及目录

```
[root@ lupa /]# ls -a
```

### ● cat

使用格式:cat[选项][文件]

功能说明:cat 是 Linux 下常用的命令,它有两种功能:

- 功能 1:是用来显示文件,它依次读取所指文件的内容并将其输出。

范例 1:将文件 test 中的内容输出

```
[root@ lupa /]# cat test
```

- 功能 2:连接 2 个或多个文件(屏幕或加 > fileName 到另一个文件)

选项说明:

选项	说明
n	由 1 开始对所有输出的行数编号
b	和-n 相似,只不过对于空白行不编号
s	当遇到有连续两行以上的空白行,就代换为一行的空白行

范例 2:把 test1 的文件内容加上行号后输入 test2 这个文件里

```
[root@ lupa /]# cat -n test1 > test2
```

范例 3:把 test1 和 test2 的文件内容加上行号(空白行不加)之后将内容附加到 test3

```
[root@ lupa /]# cat -b test1 test2 > test3
```

### ● chmod

使用格式: `chmod [选项][文件名]`

功能说明: Linux/UNIX 的文件存取权限分为三级: 文件拥有者、群组、其他。利用 `chmod` 可以修改文件权限。

选项说明(分三部分):

对象	说明
u	文件所有者
g	同组用户
o	其他用户
操作符	说明
+	增加权限
-	删除权限
=	给定权限
权限	说明
r	读取权限
w	写入权限
x	执行权限

范例 1: 将文件 `file1.txt` 设为所有人皆可读取

```
[root@ lupa /]# chmod ugo + r file1.txt
```

范例 2: 将文件 `file1.txt` 设为同组用户可读可写

```
[root@ lupa /]# chmod g + rw file1.txt
```

范例 3: 将文件 `file1.txt` 设为其他用户不可写入、不可执行

```
[root@ lupa /]# chmod o-wx file1.txt
```

### ● find

使用格式: `find [路径][选项]`

功能说明: 一个非常有效的工具, 可以遍历当前目录甚至于整个文件系统来查找某些文件或目录。

选项说明:

选项	说明
name	按文件名查找
g	查找文件所属组群
user	显示归档或压缩文件的目录内容
type	文件类型
size	查找指定大小的文件



范例 1:查找/etc 目录中的文件名含字符 cpu \* 的文件

```
[root@ lupa /]# find /etc/-name cpu *  
/etc/rc.d/init.d/cpuspeed  
/etc/pam.d/cpufreq-selector  
/etc/security/console.apps/cpufreq-selector  
/etc/gconf/schemas/cpufreq-applet.schemas  
/etc/cpuspeed.conf
```

范例 2:查找/etc 目录下大于 1000k 的文件

```
[root@ lupa /]# find -size + 1000k  
./selinux/targeted/modules/previous/base.linked  
./selinux/targeted/modules/previous/base.pp  
./selinux/targeted/modules/active/base.linked  
./selinux/targeted/modules/active/base.pp  
./gconf/gconf.xml.defaults/% gconf-tree.xml  
./gconf/schemas/gnome-terminal.schemas  
./gconf/schemas/metacity.schemas  
./gconf/schemas/ekiga.schemas  
./gconf/schemas/apps_nautilus_preferences.schemas
```

### ● less

使用格式:less[文件]

功能说明:less 用来浏览文字文件的内容,不同的是 less 允许使用者来回卷动,以浏览已经看过的部分,同时因为 less 并未在一开始就读入整个文件,因此在遇上大型文件的开启时,会比一般的文本编辑器(如 vi)来得快速。

范例:显示文件 test 的内容

```
[root@ lupa /]# less test  
Hello world!
```

### ● ln

使用格式:ln[选项]原文件生成的连接文件

功能说明:Linux/UNIX 文件系统中,连结可分为两种:硬连结与软连结,硬连结的意思是一个文件可以有多个名称;而软连结的方式则是产生一个特殊的文件,该文件的内容是指向另一个文件的位置,相当于 Windows 下的快捷方式。

选项说明:

选项	说明
s	生成软连接

范例 1:将文件 test1 产生一个软连结 test2

```
[root@ lupa /]# ln -s test1 test2
```

范例 2:将文件 test1 产生一个硬连接 test2

```
[root@ lupa /]# ln test1 test2
```

### ● locate

使用格式:locate[选项][文件]

功能说明:可以让使用者快速地搜寻文件系统内是否有指定的文件。其方法是先建立一个包括系统内所有文件名称及路径的资料库,之后当寻找时只需查询这个资料库,而不必实际深入文件系统之中,这也是有别于 find 命令的地方。

选项说明:

选项	说明
u	建立资料库
e	将排除在寻找的范围之外
l	如果是 1,则启动安全模式
f	将特定的文件系统排除在外
q	不显示任何错误信息
n	最多显示指定个输出

范例 1:寻找所有叫 test 的文件

```
[root@ lupa /]# locate test
```

范例 2:寻找所有叫 test.out 的文件,但最多只显示 100 个

```
[root@ lupa /]# locate -n 100 test.out
```

范例 3:建立资料库

```
[root@ lupa /]# locate -u
```

## 2. Shell 环境

### 1) shell 的基本概念

shell 是一个命令语言解释器(command-language interpreter)。拥有自己内建的 shell 命令集。此外,shell 也能被系统中其他有效的 Linux 实用程序和应用程序(utilities and application programs)所调用。

不论何时,你键入一个命令,它都被 Linux shell 所解释。一些命令,比如打印当前工作目录命令(pwd),是包含在 Linux BASH 内部的(就像 DOS 的内部命令)。其他命令,比如拷贝命令(cp)和移动命令(rm),是存在于文件系统中某个目录下的单独的程序。而对用户来说,你不知道(或者可能不关心)一个命令是建立在 shell 内部还是一个单独的程序。



shell 的另一个重要特性是它自身就是一个解释型的程序设计语言,shell 程序设计语言支持在高级语言里所能见到的绝大多数程序控制结构,比如循环、函数、变量和数组。shell 编程语言很易学,并且一旦掌握后它将成为你的得力工具。任何在提示符下能键入的命令也能放到一个可执行的 shell 程序里,这意味着用 shell 语言能简单地重复执行某一任务。

## 2) shell 的三种重要分支

在大部分的 UNIX 系统,三种著名且广被支持的 shell 是 Bourne shell(AT&T shell,在 Linux 下是 BASH)、C shell(Berkeley shell,在 Linux 下是 TCSH)和 Korn shell(Bourne shell 的超集)。这三种 shell 在交谈(interactive)模式下的表现相当类似,但作为命令文件语言时,在语法和执行效率上就有些不同了。

### ● Bourne shell

是标准的 UNIX shell,以前常被用来作为管理系统之用。大部分的系统管理命令文件,例如 `rc start`、`stop` 与 `shutdown` 都是 Bourne shell 的命令档,且在单一使用者模式(single user mode)下以 root 登入时它常被系统管理者使用。Bourne shell 是由 AT&T 发展的,以简洁、快速著名。Bourne shell 提示符号的默认值是 `$`。

### ● C shell

是柏克莱大学(Berkeley)所开发的,且加入了一些新特性,如命令列历程(history)、别名(alias)、内建算术、档名完成(filename completion)和工作控制(job control)。对于常在交谈模式下执行 shell 的使用者而言,他们较喜爱使用 C shell;但对于系统管理者而言,则较偏好以 Bourne shell 来做命令档,因为 Bourne shell 命令档比 C shell 命令档来的简单及快速。C shell 提示符号的默认值是 `%`。

### ● Korn shell

是 Bourne shell 的超集(superset),由 AT&T 的 David Korn 所开发。它增加了一些特色,比 C shell 更为先进。Korn shell 的特色包括了可编辑的历程、别名、函数、正规表达式万用字符(regular expression wildcard)、内建计算、工作控制(job control)、共同处理(coprocessing)、和特殊的除错功能。Bourne shell 几乎和 Korn shell 完全向上兼容(upward compatible),所以在 Bourne shell 下开发的程序仍能在 Korn shell 上执行。Korn shell 提示符号的默认值也是 `$`。在 Linux 系统使用的 Korn shell 叫做 `pdksh`,它是指 Public Domain Korn Shell。

Fedora Core 5 使用的是 Bourne shell 系列也就是 `bash shell`。

## 3) bash shell 常用命令

在 `bash` 环境中,命令是用户和计算机打交道的语言,因此,掌握 shell 命令有助于你更加熟练地使用 Linux 系统。

`alias`:设置 `bash` 别名。

`bg`:使一个被挂起的进程在后台继续执行。

`cd`:改变当前工作目录。

`exit`:终止 shell。

`export`:使变量的值对当前 shell 的所有子进程都可见。

`fc`:用来编辑历史命令列表里的命令。

fg:使一个被挂起的进程在前台继续执行。

help:显示 bash 内部命令的帮助信息。

kill:终止某个进程。

pwd:显示当前工作目录。

unalias:删除已定义的别名。

bash 还有许多命令,但这些是最常用的,想了解更详细的情况,请参考 bash 手册或者在提示符下键入:

```
[root@ lupa /]# man bash
```

#### 4) shell 编程

在 Linux 系统里,有各种各样的图形化接口工具,但是 shell 仍然是一个非常灵活的工具,它不仅是一个命令收集器,而且是一门功能强大的编程语言。下面就简单介绍一下 shell 编程。

- 建立一个脚本

用你最喜欢的一款编辑器打开编辑窗口。在打开的编辑窗口必须写入的一段语句是:

```
#!/bin/sh
```

注意:符号#! 用来告诉系统它后面的参数是用来执行该文件的程序。在这个例子中我们使用/bin/sh 来执行程序。

假设将其保存为文件名 lupa。

执行:

```
[root@ lupa root]# chmod +x lupa //修改权限;  
[root@ lupa root]# ./lupa //执行程序;
```

- 输出 hello world

```
[root@ lupa root]# vi //打开编辑器;
```

写入下列代码:

```
#!/bin/sh  
a="hello world" //对变量赋值;  
echo "A is:"  
echo $ a //打印变量 a 的内容;
```

保存为 hello 文件,并执行:

```
[root@ lupa root]# chmod +x hello //修改权限;  
[root@ lupa root]# ./hello //执行文件;
```

打印出结果:

```
A is:  
hello world
```

以上仅仅介绍了 shell 编程的基本方法,目的是为了读者有 shell 编程的概念,感兴趣的读者可以购买 shell 编程的书自行学习。

## 5) 常用 shell 命令

### ● date

使用格式: `date[MMDDhhmm[YYYY]]`

功能说明: 查看或修改系统时间。

范例 1: 查看系统时间

```
[root@ lupa root]# date  
2006 年 08 月 24 日星期四 09:13:00 CST
```

范例 2: 修改系统时间为 8 月 25 日 10 点 23 分

```
[root@ lupa root]# date 08251023  
2006 年 08 月 25 日星期五 10:23:00 CST
```

### ● cal

使用格式: `cal[YYYY]`

功能说明: 显示日期。

范例:

```
[root@ lupa root]# cal  
八月 2006  


|    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|
|    |    | 1  | 2  | 3  | 4  | 5  |
| 6  | 7  | 8  | 9  | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 27 | 28 | 29 | 30 | 31 |    |    |


```

### ● history

使用格式: `history[数字]`

功能说明: 查看 shell 中使用过的历史命令。

范例: 查看最近使用过的 3 个 shell 命令

```
[root@ lupa root]# history 3  
550 cal  
551 date  
552 history 3
```

### ● more

使用格式: `more [文件]`

功能说明: 显示文本文件内容。

范例: 查看文件 `test` 文件的内容

```
[root@ lupa root]# more test
Hello world!
```

### ● shutdown

使用格式: `shutdown [选项] [时间]`

功能说明: 对电脑关机与重启进行控制。

选项说明:

选项	说明
h	关机
r	重启

范例 1: 20 分钟后关机

```
[root@ lupa root]# shutdown -h 20
```

范例 2: 20 分钟后重启

```
[root@ lupa root]# shutdown -r 20
```

范例 3: 12 点关机

```
[root@ lupa root]# shutdown -h 12:00
```

范例 4: 12 点重启

```
[root@ lupa root]# shutdown -r 12:00
```

补充说明: Linux 还提供了两个最简单的关机重启命令 `halt` 和 `reboot`, 例:

```
[root@ lupa root]# halt //关机;
[root@ lupa root]# reboot //重启;
```

## 3. 使用 vi

### 1) vi 入门

#### ● 进入 vi

在系统提示字符(如 `$`、`#`)下敲入 `vi <文件名称>`, `vi` 可以自动帮你载入所要编辑的文件或是开启一个新文件(如果该文件不存在或缺少文件名)。进入 `vi` 后屏幕左方会出现波浪符号, 凡是列首有该符号就代表此列目前是空的。

### ● 两种模式

如上所述,vi 存在两种模式:指令模式和输入模式。在指令模式下输入的按键将作为指令来处理:如输入 a,vi 即认为是在当前位置插入字符。而在输入模式下,vi 则把输入的按键当作插入的字符来处理。指令模式切换到输入模式只需键入相应的输入命令即可(如 a, A),而要从输入模式切换到指令模式,则需在输入模式下键入 ESC 键,如果不知道现在是处于什么模式,可以多按几次[ESC],系统如发出滴滴声就表示已处于指令模式下了。

#### 新增(append)

a:从光标所在位置后面开始新增资料,光标后的资料随新增资料向后移动。

A:从光标所在列最后面的地方开始新增资料。

#### 插入(insert)

i:从光标所在位置前面开始插入资料,光标后的资料随新增资料向后移动。

I:从光标所在列的第一个非空白字元前面开始插入资料。

#### 开始(open)

o:在光标所在列下新增一行并进入输入模式。

O:在光标所在列上方新增一行并进入输入模式。

### ● 退出 vi

在指令模式下键入“:q”、“:q!”、“:wq”或“:x”(注意:号不可少),就会退出 vi。其中“:wq”和“:x”是存盘退出,而“:q”是直接退出,如果文件已有新的变化,vi 会提示你保存文件而“:q”命令也会失效,这时你可以用“:w”命令保存文件后再用“:q”退出,或用“:wq”或“:x”命令退出,如果你不想保存改变后的文件,你就需要用“:q!”命令,这个命令将不保存文件而直接退出 vi。

### ● 基本编辑

配合一般键盘上的功能键,如方向键、[Insert]、[Delete]等等。vi 还提供其他许许多多功能让文字的处理更为方便。这里先介绍 vi 如何删除与修改。vi 有两个状态,一个是输入状态,一个是删除状态,正是因为这样,才使得 vi 更加人性化和安全化。

## 2) vi 详细指令表

### ● 基本编辑指令:

#### 新增(append)

a:从光标所在位置后面开始新增资料,光标后的资料随新增资料向后移动。

A:从光标所在列最后面的地方开始新增资料。

#### 插入(insert)

i:从光标所在位置前面开始插入资料,光标后的资料随新增资料向后移动。

I:从光标所在列的第一个非空白字元前面开始插入资料。

#### 开始(open)

o:在光标所在列下新增一行并进入输入模式。

O:在光标所在列上方新增一行并进入输入模式。

x:删除光标所在字符。

dd:删除光标所在的列。

r:修改光标所在字元,r 后接著要修正的字符。

R:进入取替换状态,新增文字会覆盖原先文字,直到按[ESC]回到指令模式下为止。

s:删除光标所在字元,并进入输入模式。

S:删除光标所在的列,并进入输入模式。

#### ● 光标移动指令:

由于许多编辑工作是由光标来定位,所以 vi 提供了很多移动光标的方式,具体命令如下:

0 移动到光标所在列的最前面 对应的功能键盘[Home]。

\$ 移动到光标所在列的最后面 对应的功能键盘[End]。

[CTRL][d] 向下半页 对应的功能键盘[PageDown]。

[CTRL][f] 向下一页。

[CTRL][u] 向上半页。

[CTRL][b] 向上一页 对应的功能键盘[PageUp]。

H 移动到视窗的第一列。

M 移动到视窗的中间列。

L 移动到视窗的最后列。

b 移动到下个字的第一个字母。

w 移动到上个字的第一个字母。

e 移动到下个字的最后一个字母。

^ 移动到光标所在列的第一个非空白字元。

n- 减号移动到上一列的第一个非空白字元前面加上数字可以指定移动到以上 n 列。

n+ 加号移动到下一列的第一个非空白字元前面加上数字可以指定移动到以下 n 列。

nG 直接用数字 n 加上大写 G 移动到第 n 列。

fx 往右移动到 x 字元上。

Fx 往左移动到 x 字元上。

tx 往右移动到 x 字元前。

Tx 往左移动到 x 字元前。

; 配合 f 和 t 使用,重复一次。

, 配合 f 和 t 使用,反方向重复一次。

/string 往右移动到有 string 的地方。

? string 往左移动到有 string 的地方。

编辑指令非常有弹性,基本上可以说是由指令与范围所构成。例如 dw 是由删除指令 d 与范围 w 所组成,代表删除一个字 d(delete) w(word)。

vi 环境里,删除与复制都会将指定范围的内容放到暂存区里,然后就可以用指令 p 贴到其他地方去,这是 vi 用来处理区段拷贝与搬移的办法。

#### ● 文件操作指令

文件操作指令多以:开头,这跟编辑指令有点区别。

:q 结束编辑(quit)。

:q! 不存档而要放弃编辑过的文件。

- :w 保存文件(write)其后可加所要存档的档名。
- :wq 即存档后离开。
- :zz 功能与:wq 相同。
- :x 与:wq 相同。

## 4. 文件系统与文件管理

我们知道,Windows 采用的是 FAT32 或 NTFS 文件系统格式,然而 Linux 采用的是 ext2 或 ext3 文件系统格式,交换分区采用的则是 swap 文件系统格式。

ext 文件系统系列是专门为 Linux 设计的,它继承了 UNIX 文件系统的主要特色,采用三级索引结构和目录树型结构,并将设备作为特别文件处理。从 1993 年 ext2 的诞生,现在已经推出了 ext3 增强版本,由 RedHat 公司随 Linux 7.0 版本推出。目前越来越多的 Linux 发行版本都转向 ext3 格式,我们采用的 Fedora Core 5 使用的也是 ext3 格式。ext3 文件具有以下特点:

- 高实用性

系统如果发生异常断电,重新启动时不需要检查文件系统,只需读取文件系统日志,提高了恢复 ext3 的速度。

- 数据完整性

ext3 能保持数据与文件系统状态的高度一致性,避免了关机以外对文件系统造成的破坏。

- 速度更快

ext3 文件系统的日志功能对磁盘的驱动器读写进行优化,读写文件速度更快。

- 方便的数据转换

不用进行备份、格式化分区,可以轻松地将 ext2 转换为 ext3;ext3 文件系统也可以直接挂载成为 ext2 文件系统。

### 理解 Linux 文件结构的设计

Linux 系统中,所有一切都从“根”开始,用“/”代表,并且延伸到子目录。我们知道在 DOS、Windows 中有不同的分区,同时目录都存于分区上。Linux 则通过“加载”的方式把所有分区都放置在“根”下制定的目录里。Windows 下最接近于“根”的是 C 盘。因此对 Linux 目录结构的了解是必要的:

- /:根目录,包含整个 Linux 的所有文件目录
- /bin:操作系统所需的各种命令程序
- /boot:系统启动必须读取的文件
- /dev:外围设备代号文件
- /etc:系统设置管理相关文件
- /etc/rc.d:系统启动、关机时所需的执行脚本文件
- /etc/X11:X-Windows 配置文件目录
- /home:用户专属目录

/lib:共享函数库  
/lib/modules:系统核心模块  
/lost + found:出错文件的存放之处  
/mics:空目录,公共杂物  
/mnt:挂载目录  
/proc:核心与执行程序之间的信息  
/root:系统管理员专用目录  
/sbin:系统启动所需要执行的程序  
/tmp:用户暂时放置文件的位置  
/usr:系统命令及程序等信息  
/usr/bin:用户可以执行的命令程序  
/usr/include:供 C 语言加载的头文件目录  
/usr/local:用户放置个人安装文件  
/usr/sbin:管理员使用的程序  
/usr/share/man:多种链接帮助文件  
/usr/src:Linux 源代码存放处  
/usr/X11R6:X-Windows 文件目录,除/etc/X11 目录当中的配置文件  
/var:临时记录数据,临时文件目录

● mv

使用格式:mv[选项][文件][文件或路径]

功能说明:将文件移至另一个目录或者修改文件名。

选项说明:

选项	说明
b	若存在同名文件,覆盖前备份原来的文件
f	强制覆盖同名文件

范例 1:将文件 test1 改名为 test2

```
[root@ lupa/]# mv test1 test2
```

范例 2:将文件 test1 移动到/home/lupa 目录下,使用强制覆盖:

```
[root@ lupa/]# mv -f test1 /home/lupa/
```

● cp

使用格式:cp[选项][文件][文件或路径]

功能说明:将文件拷贝至另一目录,或将文件改名。与 mv 不同的是原文件保留。

选项说明:

选项	说明
a	尽可能将文件状态、权限等资料完整复制
r	将所拷贝文件下的所有目录全部拷贝出去
f	强行拷贝覆盖



范例 1:将文件 test1 复制到/home/lupa 目录下并命名为 test2

```
[root@ lupa/]# cp -r test1/home/lupa/test2
```

范例 2:将文件 test1 复制到/home/lupa 目录下并命强行覆盖

```
[root@ lupa/]# cp -rf test1/home/lupa/
```

#### ● tar

使用格式:tar[选项][归档或压缩文件][文件或目录]

功能说明:将文件或目录归档为 tar 文件,使用相关选项还可以对文件进行解压归档。

选项说明:

选项	说明
c	创建归档或压缩文件
r	向归档或压缩文件追加文件和目录
t	显示归档或压缩文件的目录内容
u	更新归档或压缩文件
x	解压归档或压缩文件的文件和目录
v	显示解压进度
z	使用 gzip 方式压缩或解压文件
j	使用 bzip2 方式压缩或解压文件

范例 1:将 test 文件归档为 test.tar 文件

```
[root@ lupa/]# tar -cf test.tar test
```

范例 2:将归档文件 text.tar.gz 解压

```
[root@ lupa/]# tar -zxvf text.tar.gz
```

范例 3:将归档文件 text.tar.bz2 解压

```
[root@ lupa/]# tar -jxvf text.tar.gz
```

#### ● rm

使用格式:rm[选项][归档或压缩文件][文件或目录]

功能说明:删除文件。

选项说明:

选项	说明
i	逐一询问用户是否删除
f	强行删除文件
r	删除整个文件,包括其目录下的文件

范例:删除所有 test 下的文件

```
[root@ lupa/]# ls test
test2 test3
[root@ lupa/]# rm -rf test
```

#### ● mount

使用格式: mount [选项] [设备名] [目录]

功能说明: 将磁盘设备挂载到指定目录。

选项说明:

选项	说明
t	挂载指定的文件系统类型
r	以只读方式挂载文件系统

范例 1: 查看已经挂载的文件系统

```
[root@ lupa/]# mount
/dev/mapper/VolGroup00-LogVol00 on / type ext3 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/hda7 on /boot type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
automount(pid1679) on /net type autofs (rw,fd=4,pgrp=1679,minproto=2,maxproto=
4)
```

范例 2: 挂载软盘

```
[root@ lupa/]# ls /mnt/floppy
[root@ lupa/]#
[root@ lupa/]# mount -t auto /dev/fd0 /mnt/floppy
[root@ lupa/]# ls /mnt/floppy
text text1 text2
```

范例 3: 挂载 U 盘

```
[root@ lupa/]# mkdir /mnt/usb
[root@ lupa/]# mount -t vfat /dev/sda1 /mnt/usb
[root@ lupa/]# ls /mnt/usb
Lupa. bmp lamp
```

## 5. 用户和用户组管理

用户 (User) 和用户组 (Group) 管理是每个 Linux 系统用户, 特别是 LAMP 工程师这样的使用者必须掌握的一项基本技能。

### 1) 用户管理

Linux 用户管理可以通过用户管理工具以及修改用户配置文件来实现, 但是用户管理工具最终也是通过修改用户配置文件来完成的。因此, 我们直接学习修改用户配置文件。

#### ● passwd 文件

用户配置文件是 `/etc/passwd`, 它是系统识别用户的文件, 它包含了系统所有用户的记录。例如, 我们用 lupa 这个账号登录系统, 通过 UID 来确认用户和身份, 如果存在则读取 `/etc/shadow` 影子文件所对应的 lupa 密码, 核实无误后登陆系统, 读取用户的配置文件。

用命令查看和编辑 passwd 文件:

```
[root@lupa/]# vi /etc/passwd
```

可以看到下列内容(部分):

```
root:x:1:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
squid:x:23:23:./var/spool/squid:/sbin/nologin
gdm:x:42:42:./var/gdm:/sbin/nologin
lupa:x:500:500:lupa:/home/lupa:/bin/bash
mysql:x:501:501:./home/mysql:/bin/bash
```

我们看到, 每一行信息都包含有七个字段, 各个字段有不同的含义。

第一字段: 用户名; 在上面的例子中, 我们看到登陆名为 lupa (倒数第二行)。

第二字段: 口令; 在例子中我们看到的是一个 x, 密码已被映射到 `/etc/shadow` 文件中。

第三字段: UID; UID 是用户的 ID 值, 在系统中每个用户的 UID 的值是唯一的。

第四字段: GID; GID 和 UID 类似, 是一个正整数或 0, GID 从 0 开始, GID 为 0 的组让系统赋予给 root 用户组。

第五字段: 用户名全称, 这是可选的, 可以不设置, 在本例中全名也是 lupa。

第六字段: 用户的家目录所在位置。

第七字段: 用户所用 shell 的类型, lupa 用的是 bash; 所以设置为 `/bin/bash`。

#### ● 关于 UID

UID 是用户的 ID 值, 在系统中每个用户的 UID 的值是唯一的, 更确切地说每个用户都要对应一个唯一的 UID, 系统管理员应该确保这一规则。系统用户的 UID 的值从 0 开始, 是一个正整数, 至于最大值可以在 `/etc/login.defs` 查到, 一般 Linux 发行版约定为 60000。在 Linux 中, root 的 UID 是 0, 拥有系统最高权限。在 Fedora 系统会把前 499 个 UID 和 GID 预留出来, 我们添加新用户时的 UID 从 500 开始的, GID 也是从 500 开始。至于其他系统, 有

的系统可能会把前 999 个 UID 和 GID 预留出来,以各个系统中/etc/login.defs 中的 UID\_MIN 的最小值为准。Fedora 系统 login.defs 的 UID\_MIN 是 500,而 UID\_MAX 值为 60000,也就是说我们通过 adduser 默认添加的用户的 UID 的值是 500 到 60000 之间,而 Slackware 通过 adduser 不指定 UID 来添加用户,默认 UID 是从 1000 开始。

### ● 关于 GID

GID 和 UID 类似,是一个正整数或 0,GID 从 0 开始,GID 为 0 的组让系统赋予给 root 用户组。系统会预留一些较靠前的 GID 给系统虚拟用户(也被称为伪装用户)之用。每个系统预留的 GID 都有所不同,比如 Fedora 预留了 500 个,我们添加新用户组时,用户组是从 500 开始的,而 Slackware 是把前 100 个 GID 预留,新添加的用户组是从 100 开始。查看系统添加用户组默认的 GID 范围应该查看/etc/login.defs 中的 GID\_MIN 和 GID\_MAX 值。

## 2) 用户组管理

具有某种共同特征的用户集合起来就是用户组(Group)。用户组(Group)配置文件主要有/etc/group 和/etc/gshadow,其中/etc/gshadow 是/etc/group 的加密信息文件。

/etc/group 文件是用户组的配置文件,内容包括用户和用户组,并且能显示出用户是归属哪个用户组或哪几个用户组。因为一个用户可以归属一个或多个不同的用户组。同一用户组的用户之间具有相似的特征。比如我们把某一用户加入到 root 用户组,那么这个用户就可以浏览 root 用户主目录的文件,如果 root 用户把某个文件的读写执行权限开放,root 用户组的所有用户都可以修改此文件。如果是可执行的文件(比如脚本),root 用户组的用户也是可以执行的。

用户组的特性在系统管理中为系统管理员提供了极大的方便,但安全性也是值得关注的,如某个用户下有对系统管理最重要的内容,最好让用户拥有独立的用户组,或者是把用户下的文件的权限设置为完全私有;另外 root 用户组一般不要轻易把普通用户加入进去。

用命令查看和编辑 passwd 文件。

```
[root@lupa/]# vi /etc/group
```

可以看到下列内容(部分):

```
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
lupa:x:500:
mysql:x:501:
```

与 passwd 文件一样,group 文件每一行信息所包含的段也有自己的含义。

第一字段:用户组。

第二字段:用户组密码,这个段可以是空的或!,如果是空的或有!,表示没有密码。

第三字段:用户组管理者,字段可为空,如果有多个用户组管理者,用“,”号分割。

第四字段:组成员,如果有多个成员,用“,”号分割。

### 3) 管理用户的 shell 命令

#### ● useradd

使用格式: `useradd [选项] 用户名`

功能说明: 建立新的用户账号。限超级用户使用。

选项说明:

选项	说明
c	指定用户全称
d	指定用户主目录
e	指定用户账号的期限
f	指定口令过期后多久关闭账号
g	指定用户所属的主要组群
G	指定用户所属的附加组群
s	指定用户登陆后启动的 shell 类型
u	指定用户的 UID

范例 1: 新建一个默认的 lupa2 账号

```
[root@ lupa lupa]# useradd lupa2
```

范例 2: 新建一个 lupa3 账号, 组群为 WORKGROUPD

```
[root@ lupa lupa]# useradd -g WORKGROUPD lupa3
```

#### ● passwd

使用格式: `passwd [选项] [用户]`

功能说明: 管理用户口令。

选项说明:

选项	说明
d	删除用户口令, 该用户将无口令登陆
l	暂时锁定用户账号
u	解除锁定的用户账号
s	显示指定用户账号状态

范例 1: 修改 lupa 账号口令

```
[root@ lupa lupa]# passwd lupa
```

New UNIX password:

Retype new UNIX password:

passwd: all authentication tokens updated successfully.

## 范例 2:删除口令

```
[root@ lupa lupa]# passwd -d lupa
Removing password for user lupa.
passwd: Success
```

## ● userdel

使用格式: userdel[选项][用户]

功能说明: 删除指定的账号。

选项说明:

选项	说明
r	加入该参数将删除账号及其主目录, 不使用则仅删除账号

范例:

```
[root@ lupa lupa]# userdel -r lupa
```

## ● id

使用格式: id[用户]

功能说明: 查看用户和 UID、GID 所属群信息。

范例:

```
[root@ lupa lupa]# id lupa
uid = 500( lupa) gid = 501( lupa) groups = 501( lupa)
```

## 4) 管理组群的 shell 命令

## ● groupadd

使用格式: groupadd[选项][组群名]

功能说明: 新建组群。

选项说明:

选项	说明
g	指定组群的 GID

范例: 建立一个名为 lamp 的组群

```
[root@ lupa lupa]# groupadd lamp
```

## ● groupdel

使用格式: groupdel[组群名]

功能说明: 删除组群。

范例: 删除名为 lamp 的组群

```
[root@ lupa lupa]# groupdel lamp
```



## 6. 进程管理与系统监视

### 1) 进程状态

进程是 Linux 系统资源分配和调度的基本单位, Linux 系统在创建新进程时, 都会为其指定一个唯一的号码, 即进程号 (PID), 并以此区别不同的进程。

进程分以下三个基本状态:

- 就绪状态 进程已获得除 CPU 以外运行所需的全部资源
- 运行状态 进程占用 CPU 正在运行
- 等待状态 进程等待某一资源

我们可以在桌面环境下依次点击“桌面——管理——系统监视器”, 打开“系统监视器”选中“进程”, 可以查看当前进程情况。我们也可以在 shell 终端用 ps 命令查看详细的进程信息, 具体使用下一部分将作详细介绍。

### 2) 进程管理

在使用 Linux 系统的过程中, 我们可能会对当前运行的一些程序进行管理, 比如禁止或启动, 那么这就需要对进程进行一定的管理, 对系统管理最行之有效的方法是 shell 命令控制, 下面我们将向读者介绍 shell 控制台下, 进程的管理命令。

#### ● ps

使用格式: ps [选项]

功能说明: 显示进程状态。

选项说明:

选项	说明
a	显示当前终端所有进程
e	显示系统所有进程
l	显示进程信息及其他用户进程信息
u	显示进程信息及 CPU、内存等使用率
x	显示后台进程信息
t	显示指定终端上的进程信息

范例: 显示当前进程的信息

```
[root@ lupa lupa]# ps -l
```

```
F  UID  PID  PPID  PRI  NI   VSZ  RSS   WCHAN  STAT  TTY   TIME COMMAND
4   0    1978    1    18    0   1588   352      -   Ss +  tty1   0:00/sbin/minge
4   0    1979    1    19    0   1588   380      -   Ss +  tty2   0:00/sbin/minge
4   0    1980    1    18    0   1584   352      -   Ss +  tty3    0:00
4   0    1983    1    18    0   1584   352      -   Ss +  tty6   0:00/sbin/minge
4   0    2302   2299   15    0   4968  1580      -   Ss +  pts/4  0:00-bash
4   0    3597  1894   16    0   4348   832      -   R +  pts/1  0:00 ps l
```

主要输出说明:

输出	说明
STAT	进程状态:R 运行,S 休眠,T 暂停或终止,Z 僵死
UID	进程启动者的用户 ID
PID	进程号
PPID	父进程号
NI	进程优先级的值
TTY	显示指定终端上的进程信息
TIME	进程已运行的时间

#### ● kill

使用格式:kill 进程号

功能说明:终止进程。

范例:

```
[root@ lupa lupa]# kill 1979
```

如果不能终止进程,可以加选项执行:

```
[root@ lupa lupa]# kill -9 1979
```

#### ● nice

功能格式:nice[ 优先级参数][ 命令]

功能说明:为将要启动的进程设置优先级。

范例:启动 samba 服务,设置优先级为 4

```
[root@ lupa lupa]# nice -4 samba
```

#### ● renice

功能格式:renice[ 优先级][ 参数]

功能说明:修改运行进程的优先级。

选项说明:

输出	说明
p	修改指定进程的优先级
u	修改指定用户所启动进程的默认优先级
g	修改组群中所有用户所启动进程的默认优先级

范例:将 lupa 用户的进程调高到-3

```
[root@ lupa lupa]# renice -3 -u lupa
100:old priority-1, new priority-3
```



### 3) 系统监视

我们除了知道进程情况之外,还要对 linux 系统进行监视或查看系统硬件调度情况。我们可以在桌面环境下依次点击“桌面——管理——系统监视器”,打开“系统监视器”选中“资源”,可以看到相关的硬件调度,还可以通过 shell 控制台查看更详细的情况。

#### ● top

使用格式:top[-d][秒数]

功能说明:动态显示 CPU、内存等调度情况。

范例:监视系统性能,每 5 秒刷新一次

```
[root@lupa lupa]# top -d -5
```

top-14:39:57 up 1 day, 21:23, 4 users, load average:0.01, 0.46, 0.62

Tasks:110 total, 1 running, 108 sleeping, 0 stopped, 1 zombie

Cpu(s):0.0% us, 0.3% sy, 0.0% ni, 99.7% id, 0.0% wa, 0.0% hi, 0.0% si

Mem:248168k total, 199060k used,49108k free, 5892k buffers

Swap:524280k total,48756k used, 475524k free,64820k cached

PID	USER	PR	NI	VIRT	RES	SHR	S	% CPU	MEMTIME +	COMMAND
1	root	5-11	1992	652	560	S	0.0	0.3	0:02.05	init
2	root	4-11	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/0
3	root	RT-11	0	0	0	S	0.0	0.0	0:00.00	watchdog/0
4	root	4-11	0	0	0	S	0.0	0.0	0:00.02	events/0
5	root	9-11	0	0	0	S	0.0	0.0	0:00.01	khelper
6	root	9-11	0	0	0	S	0.0	0.0	0:00.00	kthread
8	root	4-11	0	0	0	S	0.0	0.0	0:00.24	kblockd/0
9	root	9-11	0	0	0	S	0.0	0.0	0:00.00	kacpid
92	root	9-11	0	0	0	S	0.0	0.0	0:00.00	khubd
147	root	4-11	0	0	0	S	0.0	0.0	0:00.10	pdflush
148	root	9-11	0	0	0	S	0.0	0.0	0:00.11	pdflush
150	root	9-11	0	0	0	S	0.0	0.0	0:00.00	aio/0
149	root	9-11	0	0	0	S	0.0	0.0	0:00.67	kswapd0
237	root	9-11	0	0	0	S	0.0	0.0	0:00.01	kseriod
315	root	9-11	0	0	0	S	0.0	0.0	0:00.00	kpsmoused
333	root	9-11	0	0	0	S	0.0	0.0	0:00.00	kmirrord
345	root	4-11	0	0	0	S	0.0	0.0	0:07.78	kjournald

我们可以看到,系统已经运行了一天 21 小时 23 分,有 4 个用户在使用系统,当前系统使用了 0.3% 的资源,有 99.7% 是空闲的,下面是相关的 CPU 和内存信息。

#### ● free

使用格式:free[选项]

功能说明:显示内存和交换分区的相关信息。

选项说明:

输出	说明
m	以 MB 为单位显示
t	增加显示内存和交换分区的总和信息
s	指定动态显示时候的刷新频率

范例:

```
[root@ lupa lupa]# free -m
```

	total	used	free	shared	buffers	cached
Mem:	242	204	38	0	6	64
-/+ buffers/cache:		132	109			
Swap:	511	47	464			

可以看出,共有 242MB 内存大小,使用了 204MB 内存,空闲 38MB。交换分区共 511MB 大小,使用了 47MB,空闲 464MB。

#### ● df

使用格式:df[选项]

功能说明:显示系统硬盘相关信息。

选项说明:默认的显示大小为 kb,加参数 h 之后就 以 MB 显示。

范例:查看系统硬盘使用情况

```
[root@ lupa lupa]# df -h
```

/dev/mapper/VolGroup00-LogVol00	8.5G	3.9G	4.2G	49% /
/dev/hda7	99M	9.7M	84M	11% /boot
tmpfs	122M	0	122M	0% /dev/shm

## 4) 日志管理

系统的日志可以让我们更详细地了解到系统的运行状态,并在系统运行出现情况之后,有助于我们能更快地排除故障。Linux 日志文件保存于/var/log 目录中。log 目录包括以下日志文件:

输出	说明
boot. log	记录系统引导相关信息
cron	记录 cron 调度执行情况
dmesg	记录内核启动时的信息,包括硬件和系统文件信息
maillog	记录邮件服务器信息
spooler	记录新闻服务器信息
rpm_pkgs	记录已安装的 RPM 软件包信息
secure	记录系统安全信息
messages	记录系统运行过程相关信息
Xfree86. 0. log	记录图形化用户截面的 Xfree86 服务器信息

## 7. yum 包管理系统

### 1) 建立 yum 运行环境

如果您使用过 debian 系列操作系统,对 apt-get 命令一定不会陌生,同样地,Fedora 系列为我们提供了与 apt-get 类似的包管理命令 yum,它可以从一个软件库里调出用户指定的软件,然后进行安装,并且也具备卸载功能。本节向大家介绍如何添加软件库,在下一节里我们将对 yum 的使用进行说明。

首先我们访问 <http://tech.lupaworld.com/lamp>。

下载 freshrpms-release-1.1-1.fc.noarch.rpm 包。假设将其下载到/home/lupa 目录下,我们进行安装:

```
[root@ lupa lupa]# rpm -ivh freshrpms-release-1.1-1.fc.noarch.rpm
```

安装好之后即可以使用 yum 命令了。

### 2) 轻松使用 yum 命令

如果你是第一次使用 yum 命令,请不要担心,下面我们将向您介绍 yum 最常用的使用方法,相信你很快就能熟练掌握。

#### ● 搜索应用程序

```
[root@ lupa lupa]# yum search 程序名
```

yum 可以搜索所有的启动仓库,告诉用户从哪里得到需要的软件包。

#### ● 了解包安装信息

```
[root@ lupa lupa]# yum info 程序名
```

#### ● 安装应用程序

```
[root@ lupa lupa]# yum install 程序名
```

- 列出 rpm 清单

```
[root@ lupa lupa]# yum list extras
```

- 删除 rpm 包

```
[root@ lupa lupa]# yum remove 程序名
```

- 系统升级

```
[root@ lupa lupa]# yum update
```

- 查看是否有升级

```
[root@ lupa lupa]# yum check-update
```

- 列出最近升级的 rpm 包

```
[root@ lupa lupa]# yum -qa--last | tac
```

- 本地安装

```
[root@ lupa lupa]# yum localinstall/path/to/the/rpm //后面的路径表示 rpm 包路径
```

除此之外,yum 还有很多用法,上面仅仅列举了几条常用参数,更多使用方法可以用 man 命令查看:

```
[root@ lupa lupa]# man yum
```

### 3) 环境优化

对于习惯了 Windows 的你来说,可能 Linux 系统,特别是字体在你眼里非常难看,那我们就用 yum 命令对 Linux 系统进行一些简单的优化,让你改变对 Linux 的看法。

- 安装桌面美化程序

```
[root@ lupa lupa]# yum -y install gdesklets //可以用下面的方法打开它美化桌面
```

应用程序-附件-gDesklets

- 安装附加字体

```
[root@ lupa lupa]# yum -y install xfonts-arabic
[root@ lupa lupa]# yum -y install xfonts-chinese
[root@ lupa lupa]# yum -y install xfonts-gujarati
[root@ lupa lupa]# yum -y install xfonts-hebrew
[root@ lupa lupa]# yum -y install xfonts-hindi
[root@ lupa lupa]# yum -y install xfonts-japanese
[root@ lupa lupa]# yum -y install xfonts-xorg-truetype
[root@ lupa lupa]# /etc/init.d/xfs restart
```

- 安装基本编译器

```
[root@ lupa lupa]# yum -y install gcc //必须安装,后面几章安装软件都要它来编译  
[root@ lupa lupa]# yum -y install gcc-c++ //同上
```

- 安装集成开发环境

```
[root@ lupa lupa]# rpm --import/usr/share/rhn/RPM-GPG-KEY  
[root@ lupa lupa]# yum -y install anjuta
```

- 安装 NTFS 支持

```
[root@ lupa lupa]# yum -y install kmod-ntfs
```

- 升级 firfox 浏览器

```
[root@ lupa lupa]# yum -y update firfox //刚装好的系统有很多软件需要升级,这里  
仅仅列举一个对 firfox 浏览器升级的例子,其他的软件升级用同样方法即可,包括内核。
```

## 8. 配置基本网络

除了利用图形化配置网络外,我们也必须掌握命令配置方法,因为 Linux 的世界里,用命令说话才是最重要的。

### 1) 设置主机名和域名

修改主机名,可以通过修改配置文件/etc/hosts 来完成。/etc/hosts 的内容一般有如下类似内容:

一般情况下 hosts 的内容是主机名的定义,每一行为一个主机名,每行主机名由三个部分组成,分别为网络 IP 地址、主机名. 域名(主机名和域名之间有个点)、主机名的别名。在这里说明一下,主机名是主机的名字,而域名是用来解析 IP 的。

查看主机名:

```
[root@ lupa/]# hostname //查看当前主机名;  
lupa
```

修改主机名:

```
[root@ lupa/]# hostname linux //修改主机名为 linux;  
[root@ lupa/]# hostname //查看修改结果;  
linux
```

### 2) 设置 IP 地址

和 Windows 一样,在使用 Linux 的过程中,常常需要手动修改 IP 地址,在这里,我们介



绍两种修改方法。

- 临时配置

```
[root@ lupa/]# ifconfig eth0 xx.xx.xx.xx netmask xx.xx.xx.xx broadcast xx.xx.xx.xx
```

说明:eth0 后面跟的是 IP 地址;netmask 后面跟的是子网掩码;broadcast 后面跟的是广播地址。比如我们要把 IP 修改为 192.168.2.100,子网掩码为 255.255.255.0,广播地址为 192.168.2.1 则这样修改:

```
[root@ lupa/]# ifconfig eth0 192.168.2.100 netmask 255.255.255.0 broadcast 192.168.2.1
```

- 修改配置文件

如果你的网卡安装正常,就可以发现如下文件:

```
[root@ lupa/]# vi /etc/sysconfig/network-scripts/ifcfg-eth0
```

可能会看到如下信息:

```
DEVICE = eth0    //网络接口;  
ONBOOT = yes    //开机引导时激活;  
BOOTPROTO = static //采用静态 IP 地址;  
IPADDR = 192.168.2.100 //IP 地址;  
NETMASK = 255.255.255.0 //子网掩码;  
GATEWAY = 192.168.2.1 //网关;
```

根据自己的需要来更改对应的选项。

### 3) 拨号上网

在 Linux 的各大发行版本中,基本上都采用 RP-PPPOE 拨号软件。如果您的系统预安装了 RP-PPPOE,则执行命令:

```
[root@ lupa/]# adsl-setup //配置 pppoe 拨号;  
[root@ lupa/]# adsl-start //启动拨号;  
[root@ lupa/]# adsl-stop //断开连接;
```

如果没有预安装 RP-PPPOE,则可以执行

```
[root@ lupa/]# yum install rp-pppoe
```

如果是源码安装,我们要自己来编译。

登陆站点 <http://tech.lupaworld.com/lamp> 下载 rp-pppoe-3.8.tar.gz 软件。

```
[root@ lupa/]# tar zxvf rp-pppoe-3.8.tar.gz  
[root@ lupa/]# cd rp-pppoe-3.8
```

```
[root@ lupa rp-pppoe-3.8]#. /go
```

Welcome to the Roaring Penguin PPPoE client setup. First, I will run some checks on your system to make sure the PPPoE client is installed properly...

Looks good! Now, please enter some information:

USER NAME

> > > Enter your PPPoE user name (default bxxxxnxx@sympatico.ca): //在这里填写你的拨号用户名,就是服务商提供的;

> > > Enter the Ethernet interface connected to the DSL modem

For Solaris, this is likely to be something like /dev/hme0.

For Linux, it will be ethn, where 'n' is a number.

(default eth0):eth0 //如果是以太接口的 ADSL,就要在这里写上连接猫的网络接口号,此处是 eth0;

Do you want the link to come up on demand, or stay up continuously?

If you want it to come up on demand, enter the idle time in seconds after which the link should be dropped. If you want the link to stay up permanently, enter 'no' (two letters, lower-case.)

NOTE: Demand-activated links do not interact well with dynamic IP addresses. You may have some problems with demand-activated links.

> > > Enter the demand value (default no): //默认回车;

> > > Enter the DNS information here:xxx.xxx.xxx.xxx //在这里写上 DNS 服务器地址;

Please enter the IP address of your ISP's secondary DNS server.

If you just press enter, I will assume there is only one DNS server.

> > > Enter the secondary DNS server address here:xxx.xxx.xxx.xxx //这是第二个 DNS 服务器地址;

> > > Please enter your PPPoE password: //在这里输入用户的密码;

> > > Please re-enter your PPPoE password: //输入确认密码;

The firewall choices are:

0-NONE: This script will not set any firewall rules. You are responsible for ensuring the security of your machine. You are STRONGLY recommended to use some kind of firewall rules.

1-STANDALONE: Appropriate for a basic stand-alone web-surfing workstation

2-MASQUERADE: Appropriate for a machine acting as an Internet gateway for a LAN

> > > Choose a type of firewall (0-2):2 //在这里写上 2,可以共享上网;

Ethernet Interface:eth0

User name:dxxx

```
Activate-on-demand: No
Primary DNS: 202.96.134.133
Secondary DNS: 202.96.128.143
Firewalling: MASQUERADE
> > > Accept these settings and adjust configuration files (y/n)? //是否保存配置;
```

## 9. 备份数据和资料

数据对于每个用户来说都是重要的,假设数据丢失,我们过去在电脑上所做的一切都将前功尽弃。虽然有些数据可以再找回来,但是会浪费很多精力,更何况有部分数据是无法找回的,所造成的损失不可估量。因此我们必须掌握如何安全高效地做好数据备份。

和其他操作系统一样,对于 Linux 系统,不是所有东西都需要备份的,因此我们要了解哪些文件进行备份。

一般来说,除了备份用户文件(/home)和系统配置文件(在/etc 下)外,还有/usr 和/var 目录下的某些文件,包括:

```
/usr/lib/rn
/usr/lib/smail
/usr/lib/trn
/usr/local/src
/usr/local/bin
/usr/local/lpfont
/usr/local/sbin
/usr/local/man
/usr/local/thot
/usr/openwin
/usr/X11R6/lib/X11/app-defaults
/usr/X11R6/lib/X11/initrc
/var/named
/var/openwin
/var/texfonts
/etc/printcap
/etc/fstab
/etc/inittab
/etc/SF86Config
```

将这些需要备份的目录做一个.tgz 文件储存在备份媒体上即可。

在 Linux 下,提供了很多优秀的备份工具,如 tar、cpio、dump 和 restore。



### 1) tar 工具介绍

tar 是一个已移植到 Linux 中的经典 Unix 命令。tar 是 tape archive(磁带归档)的缩写,最初设计用于将文件打包到磁带上,现在我们大都使用它来实现备份某个分区或者某些重要的文件目录。我们使用 tar 可以打包整个目录树,这使得它特别适合用于备份。归档文件可以全部还原,或从中展开单独的文件和目录。备份可以保存到基于文件的设备或磁带设备上。文件可以在还原时重定向,以便将它们重新放到一个与最初保存它们的目录(或系统)不同的目录(或系统)。tar 是与文件系统无关的,因而它的适用范围很广,它可以使用在 ext2、ext3、jfs、Reiser 和其他文件系统上。

使用 tar 非常类似于使用诸如 Windows 环境下的 Winzip、WinRAR 这样的文件实用工具。只需将它指向一个目的(可以是文件或设备),然后指定想要打包的文件,就可以通过标准的压缩类型来动态压缩归档文件,或指定一个自己选择的外部压缩程序。要通过 bzip2 压缩或解压缩文件,可使用 tar-z 命令。

下面是一个简单的使用该工具进行数据备份的例子:

```
[root@ lupa/]# tar czvf /home/user/tmp/user. tgz
```

(将/home/user 目录下的所有程序文件打包备份到/tmp/user. tgz)

```
[root@ lupa/]# tar xzvf /tmp/user. tgz/home/user
```

(将备份的目录文件恢复到指定目录)

### 2) cpio 工具介绍

cpio 命令可以从 tar 或者 cpio 文件文件中拷入或者拷出文件。cpio 命令和 tar 命令兼容,但是这个命令具备一些 tar 命令没有的功能:

- 支持 cpio 和 tar 两种文件文件格式。
- 支持许多老式磁带数据格式。
- 能够通过一个管道(pipe-line)读取文件的文件名。

目前看来,只有很少的 Linux 软件包是以 cpio 格式发行的。用户如果对 cpio 命令的详细情况感兴趣,可以通过“man cpio”命令阅读它的使用手册。

### 3) dump 和 restore 介绍

dump 可以执行类似 tar 的功能。然而,dump 倾向于考虑文件系统而不是个别的文件。dump 检查 ext2 文件系统上的文件,并确定哪些文件需要备份。这些文件将出于安全保护而被复制到给定的磁盘、磁带或其他存储媒体上。在大多数媒体上,容量是通过一直写入直至返回一个 end-of-media 标记来确定的。

配合 dump 的程序是 restore,它用于从转储映像还原文件。restore 命令执行转储的逆向功能。可以首先还原文件系统的完全备份,而后续的增量备份可以在已还原的完全备份之上覆盖。可以从完全或部分备份中还原单独的文件或者目录树。

dump 和 restore 都能在网上运行,因此用户可以通过远程设备进行备份或还原。

dump 和 restore 使用磁带驱动器和提供广泛选项的文件设备。然而,两者都仅限用于 ext2 和 ext3 文件系统。如果使用的是 JFS、Reiser 或者其他文件系统,将需要其他的实用工具,比如 tar。

例如:

```
[root@ lupa/]# dump 0f /dev/nst0 //将 ext2 文件系统备份到第一个 SCSI 设备;  
[root@ lupa/]# restore -xf /dev/nst0/home/code //将备份的 SCSI 设备中的/home/  
code 目录下的所有数据还原到磁盘;
```

另外,现在市场上还有诸如 Tivoli Storage Manager 之类的商业化存储备份工具,这些都是可视化的工具,用户可以方便地通过用户界面来实现备份与恢复。值得注意的是,这是一款商业软件,因而并不是免费的,对于很多用户来说,它所提供的功能与我们上述的 Linux 自带的备份与恢复工具比较相似,只是更加人性化和友好,使用户从枯燥的命令行方式下摆脱出来,从而方便用户使用。

## 第3节 安 全

安全始于观念,对 Fedora Core 5 进行安全设置之前,有必要了解安全的基本概念。

- ◆ 确保硬盘位于 BIOS 启动顺序第一位
- ◆ 防止闯入者使用 Linux Live CD 损坏、浏览、共享整个硬盘
- ◆ 防止闯入者安装其他操作系统
- ◆ 确保设置 BIOS 密码
- ◆ 确保计算机位于安全地点
- ◆ 创建 8 位以上复杂密码
- ◆ 创建带有字母、数字、大小写混合的密码
- ◆ 确保禁用控制台下历史命令列表
- ◆ 确保在控制台下设置了文件/文件夹删除、复制、移动命令的交互选项
- ◆ 日常应用,用以普通用户登陆
- ◆ 安装防火墙
- ◆ 进行弱点检查

### 1. 禁用 GRUB 菜单的交互编辑

在刚启动电脑的时候,我们会看到一个选择系统的菜单,这个叫 grub 引导菜单,由于 Linux 的人性化操作,我们可以在这里任意编写和修改启动内容,甚至修改系统密码,在方便用户的同时也有很多安全隐患,因此,我们有必要为 grub 加一个密码。

在终端执行:

```
[root@ lupa/]# /sbin/grub/ //在终端启动 grub;
```

出现下列提示符：

```
grub > //grub 提示,在这里可以执行一切 grub 的命令;
```

执行：

```
grub > md5crypt //启用 md5 码,更加安全;
Password: * * * * (lupa) //输入 lupa,显示的是 * 号;
Encrypted:$1$HuZQW1$SBvWMXvSjKeAVGILKZ6Tm //自动生成的 MD5 码,将其
记下来,后面要用到;
grub > quit //退出 grub;
[root@ lupa/]# cp /boot/grub/menu.lst/boot/grub/menu.lst_backup //为 grub 加密
的配置文件,在修改任何重要配置文件之前,我们都做个备份,这是好习惯;
[root@ lupa/]# vi /boot/grub/menu.lst
```

找到下面一段：

```
...
title Fedora Core (2.6.15-1.2054_FC5)
    root (hd0,6)
    kernel/vmlinuz-2.6.15-1.2054_FC5 ro root=/dev/VolGroup00/LogVol00 rhgb quiet
    initrd/initrd-2.6.15-1.2054_FC5.img
...
```

替换为：

```
...
password -md5 $1$HuZQW1$SBvWMXvSjKeAVGILKZ6Tm //前面生成的 MD5 码;
title Fedora Core (2.6.15-1.2054_FC5)
    root (hd0,6)
    kernel/vmlinuz-2.6.15-1.2054_FC5 ro root=/dev/VolGroup00/LogVol00 rhgb quiet
    initrd/initrd-2.6.15-1.2054_FC5.img
...
```

保存退出。

## 2. iptables 防火墙设置

### 1) 防火墙概念

防火墙在计算机网络中的位置至关重要,它主要实现以下四个作用：

- 实现一个公司的安全策略

防火墙的主要意图是强制执行用户安全策略,如用户安全策略需要对 FTP 服务器限制

流量,那么我们需要在防火墙上强制实施这些策略。

- 创建一个阻塞点

防火墙在一个公司的私有网络和分网间建立一个检查点。这种实现要求所有的流量都要经过这个检查点。一旦检查点被建立,防火墙就可以监视,过滤和检查所有进出的流量。网络安全中称为阻塞点。通过强制所有进出的流量都通过这些检查点,管理员可以集中在较少的地方来实现安全目的。

- 记录 internet 活动

防火墙还能强制记录日志,并且提供警报功能。通过在防火墙上实现日志服务,管理员可以监视所有从外部网或互联网的访问。一个好的网络管理员,要有查看日志的好习惯。

- 限制网络暴露

防火墙在你的网络周围创建了一个保护的边界。并且对于公网隐藏了内部系统的一些信息以增加保密性。当远程节点侦测你的网络时,他们仅仅能看到防火墙。远程设备将不会知道你内部网络的布局都有些什么。防火墙提高认证功能和对网络加密来限制网络信息的暴露。通过对进入的流量进行检查,以限制外部发动攻击。

## 2) iptables

在 Linux 2.4.X 版本以上的包过滤机制,多数采用 iptables 作为包防火墙。iptables 内建有三个表,分别为 MANGLE、NAT、FILTER,当未指定规则表时,则默认为 FILTER 表,若要将 rule 加到其他表中,则要用 -t 来指明。

MANGLE 表的意思就是,会对数据包的一些传输特性进行修改,在 MANGLE 表中允许的操作是 TOS、TTL、MARK。也就是说,今后只要我们见到这个词能理解它的作用就行了。建议不要在这个包中做任何过滤。

NAT 表就是地址转换,可以做 DNAT、SNAT,可做一对一、一对多、多对多转换,该表有 prerouting 和 postrouting 两条规则链。DNAT 操作主要用在这样一种情况,你有一个合法的 IP 地址,要把对防火墙的访问重定向到其他的机子上(比如 DMZ)。也就是说,我们改变的是目的地址,以使包能从路由到某台主机。SNAT 改变包的源地址,这在极大程度上可以隐藏你的本地网络或者 DMZ 等。一个很好的例子是我们知道防火墙的外部地址,但必须用这个地址替换本地网络地址。有了这个操作,防火墙就能自动地对包做 SNAT 和 De-SNAT(就是反向的 SNAT),以使 LAN 能连接到 Internet。如果使用类似 192.168.0.0/24 这样的地址,是不会从 Internet 得到任何回应的。因为 IANA 定义这些网络(还有其他的)为私有的,只能用于 LAN 内部。MASQUERADE 的作用和 SNAT 完全一样,只是计算机的负荷稍微多一点。因为对每个匹配的包,MASQUERADE 都要查找可用的 IP 地址,而不像 SNAT 用的 IP 地址是配置好的。当然,这也有好处,就是我们可以使用通过 PPP、PPPOE、SLIP 等拨号得到地址,这些地址是由 ISP 的 DHCP 随机分配。FILTER 表是我们最常用的,包过滤就是通过它实现的,系统默认建有三个链,INPUT、OUTPUT 还有 FORWARD,这个表用来进行封包过滤的处理(如:DROP、ACCEPT、LOG、REJECT、ULOG 等),基本规则都建在这个表中。

- 定义政策

iptables 政策指,当你的封包不在你的规则之内时,则该封包的通过与否,以 Policy 的设定为准,即当我们一个 IP 包试图通过防火墙的时候,如果我们制定的 IP 规则都不适用于

它,那么这个包将依据 Policy 规则为准。

```
[root@ lupa/]# /sbin/iptables -F //清除所有已定制的 ip 规则。不要轻易使用此参数;
```

```
[root@ lupa/]# /sbin/iptables -X //关掉所有使用者建立 chain 即 tables;
```

```
[root@ lupa/]# /sbin/iptables -Z //将所有的 chain 记数与流量统计清零;
```

如果我们使用-F 命令来清除所有定制的 IP 规则之后,我们需要自己定义 IP 规则,下面为读者举一个最简单的 iptables 配置实例。一般来说,我们按照下面这种格式为 IP 定制规则。

```
[root@ lupa/]# /sbin/iptables [-t tables] [-P] [INPUT, OUTPUT, FORWARD, PREROUTING, OUTPUT, POSTROUTING] [ACCEPT, DROP]
```

参数说明:

-t: 定义 table。tables: table 的名称。

-P: 定义政策 (Policy)。

INPUT: 封包为输入主机的方向。

OUTPUT: 封包为输出主机的方向。

FORWARD: 封包为不进入主机而向外再传输出去的方向。

PREROUTING: 在进入路由之前进行的工作。

OUTPUT: 封包为输出主机的方向。

POSTROUTING: 在进入路由之后进行的工作。

将预设的政策都定义为接受:

```
[root @ test/root]# /sbin/iptables -P INPUT ACCEPT
```

```
[root @ test/root]# /sbin/iptables -P OUTPUT ACCEPT
```

```
[root @ test/root]# /sbin/iptables -P FORWARD ACCEPT
```

```
[root @ test/root]# /sbin/iptables -t nat -P PREROUTING ACCEPT
```

```
[root @ test/root]# /sbin/iptables -t nat -P OUTPUT ACCEPT
```

```
[root @ test/root]# /sbin/iptables -t nat -P POSTROUTING ACCEPT
```

#### ● 增加、插入规则

本段教用户如何定义规则,预设规则通用格式如下:

```
[root @ test /root]# /sbin/iptables [-A] [INPUT, OUTPUT, FORWARD] [-i interface] [-p TCP, UDP] [-s IP/network] [--sport ports] [-d IP/network] [--dport ports] -j [ACCEPT, DROP]
```

参数说明:

-A: 新增加一条规则,该规则增加在最后面一行。

-I: 在第一条规则加入。

INPUT: 封包为输入主机的方向。



OUTPUT:封包为输出主机的方向。

FORWARD:封包为不进入主机而向外再传输出去的方向。

-i:流入的网卡接口。

-o:流出的网卡接口。

interface:网络卡接口,例如 ppp0、eth0、eth1 等。

-p:封包协议啦。

TCP:封包为 TCP 协议的封包。

UDP:封包为 UDP 协议的封包。

-s:来源封包的 IP 或者是 Network。

--sport:来源封包的 port 号码。

-d:目标主机的 IP 或者是 Network。

--dport:目标主机的 port 号码。

-j:动作,可以接底下的动作。

ACCEPT:接受该封包。

DROP:丢弃封包。

举几个简单的例子:

```
[root @ test/root] #/sbin/iptables -A INPUT -i lo -j ACCEPT
```

接受所有的来自 lo 这个接口的封包。

```
[root @ test/root] #/sbin/iptables -A INPUT -i eth0 -p TCP -s 192.168.0.1 -j ACCEPT
```

接受来自 192.168.0.1 这个 IP 的封包。

```
[root @ test/root] #/sbin/iptables -A INPUT -i eth0 -p TCP -s 192.168.1.0/24 -j ACCEPT
```

接受来自 192.168.1.0 C Class 的网域的任何一部计算机。

```
[root @ test/root] #/sbin/iptables -A INPUT -i eth0 -p TCP -s 192.168.1.25 -j DROP
```

丢弃来自 192.168.1.25 的 IP 的封包。

```
[root @ test/root] #/sbin/iptables -A INPUT -i eth0 -p TCP --dport 21 -j DROP
```

丢弃访问 21 这个端口的封包。

```
[root @ test/root] #/sbin/iptables -A INPUT -i eth0 -p TCP -s 192.168.0.24 --dport 22 -j ACCEPT
```

端口 22 接受来自 192.168.0.24 主机的访问。

请注意:防火墙的规则是,一行一行依序来检查,若符合任何一条规则,则予以动作(接受或丢弃),否则继续往下直到检查最后一条。

# Apache 安装和使用

Apache 服务器拥有牢靠可信的美誉,已用于大多数 Web 服务中,大多数知名网站都采用了它。本章详细讲解了 Apache 的安装和使用,从编译与安装开始,到 Apache 的主文件配置、访问控制、虚拟主机、认证授权和访问控制等设置都做了详细的介绍,读者可以边学边做,这样学起来比较容易上手。本章参考了 Apache 官方手册部分内容。

## 第1节 Apache1.3 与 Apache2.0

开源 Web 服务器 Apache 版本现在已更新到 2.0。作为多年来 Web 服务器的首选,Apache 和 Web 同步发展,对现代网络的发展作出巨大贡献。早在 1.X 版本时,Apache 已经足够稳定和强大,但仍然存在一些功能上的不足。现在,一个解决了以前 Apache 版本的各种问题,适应今天各种 Web 主机的多样性需求的全新版本已经出现。新版本支持多种平台,通过 APR (Apache Portable Runtime),Apache 将各种功能的实现从具体的服务器环境抽象了出来,从而减少了为使 Apache 在不同平台上运行所需要进行的工作。现在 Apache 可以运行在几乎所有的 UNIX/Linux 平台,以及各种 Windows 平台,Netware 和 OS/2。

新版本的编译和安装也简单了许多。Apache 使用了标准 GNU 自动设置工具,对于 Windows 平台,它提供图形化的安装界面。在设置方面,新版本去掉了一些指示符号,不过大部分工作仍然是基于一个文件的配置文件。这对于习惯图形界面设置工具的用户一开始或许不适应。设置工具与核心的发布包是分离的,一些类似 ApacheConf 可以免费或以很低的价格获取,利用这些工具可以很方便的对 Apache 进行各种设置。从运行方面看,新版本比以前版本更强大,更具扩展性。一个多线程/多进程的混合模块使它可以充分利用各种规模的主机性能,这正是 Apache1.3 所缺乏的。新版本以插件的方式提供了对一些特殊平台和几个通用模块的支持。对 IPv6 的官方支持已经为互联网的升级做好了准备,对 HTTP1.1 的支持提高了代理服务器的性能。

下表例示了 Apache 1.3 版本和 2.0 版本的主要区别:

特性	Apache1.3	Apache2.0
IPv6	需要非官方的补丁	完全支持
线程	可扩展性较差	支持多线程,可扩展性更佳
编译设置	APACI	GNU Autoconf
服务器设置		减少了一些迷惑提示
平台支持	平台有限,问题比较多	用 APR 带来了大量的扩展
Unicode 支持	Windows 下面很差	NT/XP Unicode 扩展
国际化支持		加强,多语种的错误信息
代理支持	HTTP1.0	HTTP1.1
MS ISAPI	不支持	支持
SSH	没有正式支持	通过 OpenSSL 的 mod_ssl 模块提供支持

## 第 2 节 Apache2.0 模块介绍

### 1. 核心的增强

#### 1) Unix 线程

在支持 POSIX 线程的 Unix 系统上,现在 Apache 能在混合多进程、多线程模式下运行,使许多(但不是全部的)配置的可扩缩性得到改善。

#### 2) 新的编译系统

重写了旧的编译系统,现在是基于 autoconf 和 libtool 的,使得 Apache 的配置系统与其他软件包更加相似。

#### 3) 多协议支持

Apache 已经拥有了能够支持多协议的底层构造。mod\_echo 就是一个例子。

#### 4) 对非 Unix 平台更好的支持

Apache 2.0 在诸如 BeOS, OS/2 和 Windows 等非 Unix/Linux 平台上有了更快的速度和稳定性。随着平台特定的 multi-processing modules (MPMs) 和 Apache Portable Runtime (APR) 的引入,Apache 在这些平台上的指令由它们本地的 API 指令实现。避免了以往使用 POSIX 模拟层造成的 bug 和低性能。

#### 5) 新的 Apache API

2.0 中模块的 API 有了重大改变。很多 1.3 中模块排序、模块优先级的问题已经得到



解决。2.0 自动处理了很多这样的问题,模块排序现在用 per-hook 的方法进行,从而拥有了更强的灵活性。同时,增加了新的调用以提高模块的性能,而无需修改 Apache 服务器核心。

## 6) IPv6 支持

在所有能够由 Apache Portable Runtime 库提供 IPv6 支持的系统上,Apache 默认获得 IPv6 侦听套接字。另外,Listen、NameVirtualHost 和 VirtualHost 指令支持 IPv6 的数字地址串。

## 7) 过滤

Apache 的模块现在可以写成过滤器的形式,当内容流经它到服务器或从服务器到达的时候进行处理。比如,可以用 mod\_include 中的 INCLUDES 过滤器将 CGI 脚本的输出解析为服务器端包含指令。mod\_ext\_filter 允许外部程序充当过滤器的角色,就像用 CGI 程序做处理器一样。

## 8) 多语种错误回馈

返回给浏览器的错误信息现在已经用 SSI 文档实现了多语种化。管理员可以利用此功能进行定制以达到观感的一致。

## 9) 简化了的配置

很多易混淆的配置项已经进行了简化。经常产生混淆的 Port 和 BindAddress 配置项已经取消了;用于绑定 IP 地址的只有 Listen 指令;ServerName 指令中指定的服务器名和端口仅用于重定向和虚拟主机的识别。

## 10) 本地 Windows NT Unicode 支持

Apache 2.0 在 Windows NT 上的文件名全部使用 utf-8 编码。这个操作直接转换成底层的 Unicode 文件系统,由此为所有以 Windows NT(包括 Windows 2000 和 XP)为基础的安装提供了多语言支持。这一支持目前尚未涵盖 Windows 95, 98 or ME 系统,因为它们仍使用机器本地的代码页进行文件系统的操作。

## 11) 正则表达式库更新

Apache 2.0 包含了兼容 Perl 的正则表达式库(PCRE)。所有的正则表达式现在都使用了更为强大的 Perl 5 的语法。

# 2. Apache2.0 的增强模块

## 1) mod\_ssl

Apache 2.0 中的新模块。此模块是一个面向 OpenSSL 提供的 SSL/TLS 加密协议的一个接口。

## 2) mod\_dav

Apache 2.0 中的新模块。此模块继承了 HTTP 分布式发布和版本控制规范,用于发布和维护 Web 内容。

## 3) mod\_deflate

Apache 2.0 中的新模块。此模块允许支持此功能的浏览器请求页面内容在发送前进行压缩,以节省网络带宽。

## 4) mod\_auth\_ldap

Apache 2.0.41 中的新模块。此模块允许使用 LDAP 数据库存储 HTTP 基本认证所需的信息。随之而来的另一个模块:mod\_ldap,则提供了连接池和结果的缓冲。

## 5) mod\_auth\_digest

利用共享内存实现了对跨进程的 session 缓冲的额外支持。

## 6) mod\_charset\_lite

Apache 2.0 中的新模块。这个试验模块允许针对字符集的转换和重新编码。

## 7) mod\_file\_cache

Apache 2.0 中的新模块。这个模块包含了 Apache 1.3 中 mod\_mmap\_static 模块的功能,另外进一步增加了缓冲能力。

## 8) mod\_headers

此模块在 Apache 2.0 中更具灵活性。现在,它可以更改 mod\_proxy 使用的请求头信息,并可以有条件地设置回复头信息。

## 9) mod\_proxy

经过重新编写的代理模块可以充分利用新的过滤器结构的优势,从而实现一个更为可靠的兼容 HTTP/1.1 的代理模块。另外,新的 <Proxy> 指令提供了更具可读性(而且更快)的代理站点控制;同时,不再支持重载 <Directory“proxy:...”> 指令的方法。这个模块现在依照协议支持分为 proxy\_connect、proxy\_ftp 和 proxy\_http 三个部分。

## 10) mod\_negotiation

新的 ForceLanguagePriority 指令可以确保在所有情况下客户端都收到单一的一个文档,以取代不可接受或多选择的回应。另外,negotiation 和 MultiViews 算法已经进行了优化以提供更完美的结果,并提供了包括文档内容的新类型表。

## 11) mod\_autoindex

经自动索引后的目录列表现在可被配置为使用 HTML 表格而使格式更清晰,而且允许

更为细化的排序控制,包括版本排序和通配符过滤目录列表。

### 12) mod\_include

新的指令集允许修改默认的 SSI 元素的开始和结束标签,而且允许以主配置文件里的错误提示和时间格式的配置取代 SSI 文档中的相应部分。正则表达式(现在已基于 Perl 的正则表达式语法)的解析和分组结果可以用 mod\_include 的变量 \$ 0.. \$ 9 取得。

### 13) mod\_auth\_dbm

现在,可以使用 AuthDBMType 支持多种类似 DBM 的数据库。

## 第3节 Apache 的编译与安装

Apache 2.0 的安装配置环境与以前版本差别非常大。Apache 1.3 使用了一个自定义的脚本来简化安装;而 Apache2.0 现在也像其他许多开源项目一样,利用 libtool 和 autoconf 来建立安装环境。假设我们目前所处的位置为 / 目录下。

### 1. 系统要求

编译 Apache 需要满足以下要求:

#### 1) 磁盘空间

至少 50MB 以上的临时磁盘剩余空间。Apache 安装完毕后大约需要 10MB 的空间,实际磁盘需求会因编译设置和是否安装第三方模块而有所不同。

#### 2) ANSI-C 编译器及其支持环境

必须装有 ANSI-C 编译器,推荐使用来自 <http://www.gnu.org> 的 GNU C compiler (GCC) (4.1.1 版)。假如没有 GCC,那么要确保现有的编译器符合 ANSI 标准,并且 PATH 中必须包含指向基本编译工具比如 make 的路径。

#### 3) 确保正确的时间

因为 HTTP 协议的元素都会用到时间,所以有必要熟悉你的系统的时间同步机制。在基于 Network Time Protocol (NTP) 的系统中,一般是用 ntpdate 或 xntpd 来同步时间。

#### 4) Perl 5 (可选)

部分用 Perl 写的支持脚本,如 apxs 和 dbmmanage,需要 Perl 5 解释器(5.003 或以上版本就可以了),如果 configure 没找到这样的解释器也没关系,不会影响 Apache 2.0 的编译和安装,只是这些支持脚本不能用而已。如果系统中存在多个 Perl 解释器,比如有厂商提供的 Perl4,还有你自己找来的 Perl5,建议使用 --with-perl 选项来确保脚本 ./configure 使用正确

的版本。

## 2. 下载 Apache

如何获得 Apache? 你可以通过官方网站:<http://www.apache.org/dist/httpd/>及镜像站点:<http://mirror.lupaworld.com/apache/httpd/>下载。版本号带有 alpha 的是预测试版本,其运行可能正常也可能不正常;带有 beta 的是仍需要测试和纠错的比较可靠的版本。最好选择下载最新的 Apache HTTP Server 的发行版,并且是文件名中既没有 alpha 也没有 beta 的版本。本书使用 Apache2.0 系列中的 Apache HTTP Server 2.0.59 作为范例。

下载完毕后,特别是从镜像站点下载的,建议你对下载来的 tar 包作 PGP 签名校验,这样做原因很简单,可以保证其完整,而且未被篡改。校验工作分两个步骤进行,第一步,必须从 <http://mirror.lupaworld.com/apache/httpd> 获得一个密钥(KEYS)(为确保此密钥文件本身未被篡改,推荐使用 Apache 较早发行版中的或者导入来自公共密钥服务器的密钥),用以下命令导入到你的个人密钥环中(根据你的 PGP 版本不同存在差异):

```
[root@lupa/]# pgp < KEYS
```

或者

```
[root@lupa/]# gpg--import KEYS
```

第二步,从 <http://mirror.lupaworld.com/apache/httpd> 获得 PGP 签名以校验 tar 包。这个签名的文件名由其原 tar 包文件名后附加.asc 组成。接着,可以使用如下命令来校验发行版(根据你的 PGP 版本不同存在差异):

```
[root@lupa/]# pgp httpd-2_0_NN.tar.gz.asc
```

或者

```
[root@lupa/]# gpg--verify httpd-2_0_NN.tar.gz.asc
```

能够看到这样的信息:

```
Good signature from user "Martin Kraemer <martin@apache.org>".
```

或者是“the relationship between the key and the signer of the key cannot be verified”的信息,其原因与你的密钥环中的信任关系有关,如果你信任这个密钥文件,那就没关系。

## 3. 解压安装包

解压 Apache Httpd 的 tar 包非常简单,假如安装包放在/home 目录下面。

```
[root@lupa home]# tar-zxvf httpd-2.0.59.tar.gz
```

这样就在当前目录下建立了一个包含发行版源代码的目录,必须 cd 进入这个目录以继

续服务器的编译。

```
[root@ lupa home]# cd httpd-2.0.59
```

## 4. 编译 Apache

首先通过 `./configure` 命令检验操作系统的环境是否支持 apache 源代码的编译过程,同时在参数中制定 Apache 的安装目录。

```
[root@ lupa httpd-2.0.59]# ./configure--prefix = /usr/local/apache
```

执行这个命令就可以编译 Apache 了。

```
[root@ lupa httpd-2.0.59]# make
```

对一个基本的配置的编译,实际需要的时间将根据你的硬件和选择的模块数量来决定。

## 5. 安装 Apache

下面开始安装 Apache,执行:

```
[root@ lupa httpd-2.0.59]# make install
```

接着,通过修改 `/usr/local/apache/conf/` 下的配置文件,来配置 Apache HTTP 服务器。

```
[root@ lupa httpd-2.0.59]# vi /usr/local/apache/conf/httpd.conf
```

<http://man.lupaworld.com/content/manage/ApacheManual/index.html> 有完整的配置指令参考。

# 第 4 节 Apache 的启动和运行

## 1. Apache 是如何启动的

若配置文件中 `Listen` 的定义为默认的 80 端口(或其他 1024 以下的端口),则启动 Apache 将需要 root 权限以将它绑定在这个端口上。当服务器开始启动并完成了一些比如打开日志文件之类的操作,它将创建很多子进程来完成一些如侦听和响应客户端请求的工作。httpd 主进程仍然以 root 用户的权限运行,而子进程以一个较低权限的用户运行。这将依照你所选择的多道处理模块进行控制。

建议使用 `apachectl` 控制脚本来启动 httpd 执行文件。该脚本设置了在某些操作系统中正常运行 httpd 所必需的环境变量。`apachectl` 会传递命令行的所有参数,因此大部分 httpd

的选项在 `apachectl` 中也适用。你可以直接修改 `apachectl` 脚本, 改变头部的 `HTTPD` 变量使之指向 `httpd` 文件的正确位置, 或者设置任意命令行参数, 使之有效。

如果正常启动, 服务器将与终端分离并几乎同时出现平时的命令行提示符。这意味着服务器已经启动并开始运行。然后你就可以用浏览器去访问你的服务器来查看 `DocumentRoot` 目录下的测试文档及其页面链接里的其他文档的本地拷贝。

## 2. 随系统启动

如果你想让 Apache 服务器在系统重启后自动运行, 那就应该把 `httpd` 或者 `apachectl` 的调用加入到你的系统启动文件中。这就会以 `root` 权限启动 Apache。当然, 在此之前, 必须保证你的服务器已经完成了安全和访问权限的设定。

编辑文件 `rc.local`:

```
[root@ lupa ~]# vi /etc/rc.local
```

在配置文件里加入下列一行:

```
/bin/sh-c '/usr/local/apache/; ./bin/apachectl start'
```

保存退出即可。

## 3. 启动测试

现在, 可以执行这个命令立即启动你的 Apache HTTP 服务器:

```
[root@ lupa ~]# /usr/local/apache/bin/apachectl start
```

你可以用 URL `http://localhost/` 来访问你的第一个网页了, 这个网页位于 `DocumentRoot` 目录下, 通常是 `PREFIX/htdocs/`。随后, 可以输入如下命令停止服务器:

```
[root@ lupa ~]# /usr/local/apache/bin/apachectl stop
```

## 第 5 节 了解 `httpd.conf` 文件

Apache 的配置文件是包含了若干指令的纯文本文件。主配置文件一般叫 `httpd.conf`, 其位置由编译时确定, 也可以用命令行参数 `-f` 来改变。可以用 `Include` 指令附加其他配置文件。所有配置文件都可以使用所有指令。主配置文件的改变将在启动或重新启动后生效。

自 Apache 1.3 起加入了一个新功能, 即, 若配置文件是一个目录, Apache 会解析该目录及其子目录中的所有文件作为配置文件。该功能的用途是, 可以通过在这个目录中建立小的配置文件来设置虚拟主机, 这样就可以简单地增加或者删除虚拟主机, 而不需要修改其他任何文件, 使得类似的操作简单许多。服务器还会读取一个包含 `MIME` 文件类型的文件, 其

文件名由 TypesConfig 指定,默认是 mime.types。

## 1. 配置文件的语法

Apache 配置文件中每行包含一个指令,行末使用反斜杠“\”可以换行,但是反斜杠与下一行中间不能有任何其他字符(包括空格)。

配置文件中的指令是不区分大小写的,但是指令的参数通常是区分大小写的。以“#”开头的行表示为注释。注解不能出现在指令的后边。指令前边的空行和空白字符会被忽略,因此可以采用缩进保持层次清晰。

可以用 `apachectl configtest` 或者命令行参数 `-t` 检查配置文件中的错误,而不必启动 Apache 服务器。

## 2. 指令的作用域

主配置文件中的指令作用范围是整个服务器。把指令嵌入到诸如 `<Directory>`, `<DirectoryMatch>`, `<Files>`, `<FilesMatch>`, `<Location>`, 以及 `<LocationMatch>` 等段中,可以限制指令的作用域为文件系统中的某个位置或者 URL,同时可以嵌套。

Apache 可以同时支持许多不同站点,称为 Virtual Hosting。 `<VirtualHost>` 也会限制其中的指令的作用域,使之仅对特定站点有效。

大多数指令可以包含在任意的段中,但是部分指令的作用域没有意义,比如控制进程建立的指令只是针对主服务器。

## 3. 指令介绍

类似于其他 Linux 应用程序,Apache 也有配置文件,在运行时间里可被系统管理员用来配置和控制。Apache 使用名为 `httpd.conf` 的文件,此文件在安装 Apache 时存放在 `conf/` 目录里。让我们来看看这个文件的指令:

### 1) ServerRoot

`ServerRoot` 指令设置了服务器存在的目录。通常它将包含 `conf/` 和 `logs/` 子目录。其他配置文件的相对路径即基于此目录。举例如下:

```
ServerRoot/usr/local/apache
```

### 2) DirectoryIndex

很多情况下,URL 中并没有指定固定的名字,而是给出了一个目录名。那么 Apache 服务器就自动返回这个目录下由 `Directory Index` 定义的文件,当然可以指定多个文件名,系统会在这个目录下顺序搜索。生成这个目录下的所有文件列表,提供用户选择。此时该目录的访问控制选项中的 `Index` 选项必须打开,使得服务器能够生成目录列表,否则 Apache 将

拒绝访问。

DirectoryIndex 可以配置一个文档,也可以配置多个文档,多个文档用空格分开,如:

```
DirectoryIndex index.html index.htm default.htm
```

### 3) AddDefaultCharset

这个指令是设置服务器的字符编码设置,AddDefaultCharset Off 这样的设置将会禁用此功能。AddDefaultCharset On 将根据此指令的需要启用 Apache 内部的默认编码 iso-8859-1。

如果禁用这个功能,这样仅凭 HTML 文件头中 META 标签设置的语言来决定网页语言。很多网页出现乱码情况,都是由于编码设置不对引起的,对于 Apache 服务器来说,你也可以指定使用另外一个字符编码,比如简体中文。示例如下:

```
AddDefaultCharset GB2312
```

### 4) PidFile

当服务器启动时,服务器需要将其进程 ID 号存放在此文件中。

### 5) ScoreBoardFile

保存服务器内部的处理信息。

### 6) Timeout

接收和发送数据的超时设置,秒数。

### 7) KeepAlive

是否支持持久连接(而不是每个请求建一个连接),设 off 关闭此功能。持续作用扩展自 HTTP/1.0 和 HTTP/1.1 的长连接特性,提供了长效的 HTTP 会话,用以在同一个 TCP 连接中进行多次请求。在某些情况下,这样的方式会对包含大量图片的 HTML 文档造成的延时起到 50% 的加速作用。在 Apache 1.2 版本以后,可以设置 KeepAlive On 以启用持续作用。默认值:KeepAlive on。

### 8) MaxKeepAliveRequests

限制了当启用 KeepAlive 时,每个连接允许的请求数量。将此值设为“0”,则不限制请求的数目。建议设置一个比较大的值,以确保服务器性能。默认值:MaxKeepAliveRequests 100。示例如下:

```
MaxKeepAliveRequests 500
```

### 9) KeepAliveTimeout

Apache 在关闭连接前等待下一次请求的时间。一旦收到一个请求,超时值将会被设置



为 Timeout 指令指定的值。如果是高负荷的服务器,把 KeepAliveTimeout 设置成一个比较大的值将导致一些性能方面的问题:超时值越大,与空闲客户端保持连接的服务器进程越多。  
默认值:KeepAliveTimeout 15

### 10) MinSpareServers

这个指令设定最小闲置子进程数量。闲置子进程是指目前没有处理要求的进程。示例如下:

```
MinSpareServers 5
```

### 11) MaxSpareServers

设定最大闲置子进程数量。

### 12) StartServers

设定启动时建立的子进程数量。示例如下:

```
StartServers 5
```

### 13) MaxClients

是非常重要的指令,设定的是 Apache 可以同时处理的请求数,是对 Apache 性能影响最大的参数。默认值 150 是绝对不够的,若请求总数已达到这个值,那么后面的请求就要排队等候,直到某个请求处理完毕。这也是系统资源足够多而 HTTP 访问却很慢的主要原因。系统管理员可以根据硬件配置和负载情况来动态调整这个值。虽然理论上这个值越大,可以处理的请求就越多,但 Apache 默认的限制不能大于 256。如果把这个值设为大于 256,那么 Apache 将无法启动。可以通过 ServerLimit 指令,使得 Apache 可以加大 MaxClients。

### 14) MaxRequestsPerChild

服务进程中允许的最大请求数目。

### 15) Listen

允许你将 Apache 绑定到指定的 IP 地址或端口,而不是默认端口。像下面这样指定侦听的 IP 地址,防止 Apache 抢占所有绑定的 IP 地址。

```
Listen 12.34.56.78:80  
Listen 12.34.56.79:80
```

## 第6节 使用 Apache 虚拟主机

虚拟主机是指在一台机器上运行多个站点(比如:www.test1.com 和 www.test2.com)。

若每个网络站点拥有不同的 IP 地址,那么虚拟主机可以是“基于 IP”的;如果只有一个 IP 地址,也可以是“基于主机名”的。

基于 IP 地址的虚拟主机方式,需要不同的主机名解析到不同的 IP 地址,提供虚拟主机服务的机器上同时设置有这些 IP 地址。因此在提供虚拟主机服务的机器上设立多个 IP 地址,比较浪费 IP 地址,因此这种方式越来越少使用。

基于主机名的虚拟主机工作方式是,当客户程序向服务器发出请求时,用户想要访问的主机名通过请求头中的 Host 语句传递给服务器。比如:www.test1.com, www.test2.com 都对应于同一个 IP 地址。这个方式,只要提供虚拟主机服务的机器上有一个 IP 地址就行,而且占用资源少,管理方便。

## 1. 虚拟主机配置

配置指令:使用 Apache 的命令行参数 -t -D DUMP\_VHOSTS,可以帮助调试虚拟主机的配置,如:

```
[root@lupa/]# /usr/local/apache/bin/httpd-S
```

### 1) < VirtualHost >

包含仅适用于指定主机名或 IP 地址的指令: < VirtualHost > 和 < /VirtualHost > 用于封装一组仅适用于特定虚拟主机的指令。任何在虚拟主机配置中可以使用的指令也同样可以在这里使用。当服务器接受了一个特定虚拟主机的文档请求时,它会使用封装在 < VirtualHost > 配置段中的指令。地址可以是:

虚拟主机的 IP 地址。

虚拟主机 IP 地址对应的完整域名。

字符 \*, 仅与 NameVirtualHost \* 配合使用以匹配所有的 IP 地址。

字符串 \_default\_, 与基于 IP 的虚拟主机联用以捕获所有没有匹配的 IP 地址。

示例:

```
< VirtualHost 122.122.122.122 >
    ServerAdmin webmaster@test.com
    DocumentRoot /usr/local/apache/htdocs/test.com
    ServerName test.com
    ErrorLog logs/test.com-error_log
    TransferLog logs/test.com-access_log
< /VirtualHost >
```

IPv6 的地址必须放入方括号中指定,否则作为可选项的端口号将无法确定。示例如下:

```
< VirtualHost [ fe80::a00:20ff:fea7:ccea ] >
    ServerAdmin webmaster@test.com
    DocumentRoot/usr/local/apache/htdocs/test.com
    ServerName test.com
    ErrorLog logs/test.com-error_log
    TransferLog logs/test.com-access_log
</VirtualHost>
```

每个虚拟主机所对应的 IP 地址、端口号或主机名不能相同。在第一种情况下,服务器所在物理机器必须配置为可以为多个地址接受 IP 包。(在机器没有多个网络硬件界面的情况下,如果你的操作系统支持,你可以使用 `ipconfig alias` 命令来达到这个目的)。

在使用基于 IP 的虚拟主机时, `_default_` 可以作为匹配任何 IP 地址的虚拟主机(前提是 `_default_` 没有匹配上其他列出的虚拟主机)。当没有进行 `_default_` 虚拟主机的设定时,在没有 IP 与请求匹配的情况下,将使用“主服务器”(包括所有在虚拟主机配置段之外的配置)的配置。(注意:任何匹配 `NameVirtualHost` 指令的 IP 地址既不会使用“main”服务器配置,也不会使用 `_default_` 虚拟主机的配置)。也可以指定一个端口来改变匹配的端口。如果没有指定,它将使用主服务器中最近的那个 `Listen` 语句指定的端口。或者指定: `*` 来匹配那个地址上的所有端口。(当使用 `_default_` 时,建议使用本方法。)

注意: `< VirtualHost >` 的使用不会影响到 Apache 侦听的地址。你可以使用 `Listen` 来确保 Apache 侦听着正确的地址。

## 2) NameVirtualHost

`NameVirtualHost` 指令是配置基于域名的虚拟主机所必需的指令之一。地址参数可以使用主机名,但建议使用 IP 地址。例如:

```
NameVirtualHost 122.122.122.122
```

使用 `NameVirtualHost` 指令,可以指定一个基于域名的虚拟主机将使用哪个 IP 地址来接受请求。当防火墙或是其他代理接受了请求并把它转到服务器所在的另外一个 IP 地址时,必须指定响应请求的机器物理界面上的 IP 地址。对于多个地址使用了多个基于域名的虚拟主机,应该为每个地址使用此指令。

注意:“主服务器”和任何其他默认服务器都不会响应发送到 `NameVirtualHost` IP 地址的请求。(指定了 `NameVirtualHost` 但没有为这个地址指定任何虚拟主机的情况除外)。

同时,可以为使用的基于域名的虚拟主机指定一个端口号。例如:

```
NameVirtualHost 122.122.122.122:8080
```

IPv6 地址必须封装在一对方括号内,如下例所示:

```
NameVirtualHost [ fe80::a00:20ff:fea7:ccea ] :8080
```

为接受所有界面的请求,可以使用参数 `*`

```
NameVirtualHost *
```

< VirtualHost > 指令的参数。

请注意 < VirtualHost > 指令的参数必须与 NameVirtualHost 指令的参数完全一致。

```
NameVirtualHost 122.122.122.122
< VirtualHost 122.122.122.122 >
...
< /VirtualHost >
```

### 3) ServerName

ServerName 指令设置了服务器用于辨识自己的主机名和端口号。这主要用于创建转向 URL。例如,一个放置 Web 服务器的主机域名为 simple.test.com,但同时有一个 DNS 别名 www.test.com。可以使用如下的指令:

```
ServerName www.test.com:80
```

如果没有指定 ServerName,服务器会尝试着对 IP 地址进行反向查询以得到主机名。如果在服务器名中没有指定端口号,服务器将使用接受请求的端口。为加强可靠性和可预测性,建议使用 ServerName 显示的指定一个主机名和端口号。

若使用基于域名的虚拟主机,在 < VirtualHost > 配置段中的 ServerName 将指定为了匹配这个虚拟主机,在请求的 Host 头中必须出现的主机名。

### 4) ServerAlias

ServerAlias 指令与基于域名的虚拟主机联用,设定虚拟主机的别名。

示例如下:

```
< VirtualHost * >
    ServerName www.test.com
    ServerAlias www1.test.com
    ServerAlias www2.test.com
    ...
< /VirtualHost >
```

### 5) ServerPath

ServerPath 指令为主机设置了保守的(legacy)URL 路径名,用于和基于域名的虚拟主机配合使用。

## 2. 虚拟主机的例子示范

### 1) 在一个 IP 地址上运行多个基于域名的 Web 站点

服务器只有一个 IP 地址,而在 DNS 中有很多域名解析到这台机器。需要在这个机器上运行 `www.test1.com` 和 `www.test2.com` 两个站点。

注意:在 Apache 服务器的配置中创建一个虚拟主机并不会自动在 DNS 中对主机名做相应更新。必须手工把域名解析到你的 IP 地址。否则别人是无法看到你的 Web 站点的。你也可以在 `hosts` 文件中添加相应条目进行测试,但这种方法仅适用于本机或那些有这些条目的机器来使用。为了测试成功,我们需要手动设定域名的解析。将 `www.test1.com` 和 `www.test2.com` 解析到本地 IP 地址 `127.0.0.1`,操作如下:

```
[root@lupa/]# echo "127.0.0.1 www.test1.com" >> /etc/hosts
[root@lupa/]# echo "127.0.0.1 www.test2.com" >> /etc/hosts
```

你也可通过编辑 `hosts` 文件来达到目的,操作如下:

```
[root@lupa/]# vi /etc/hosts //打开 hosts 文件
```

把以下二条记录写到 `hosts` 文件中:

```
127.0.0.1 www.test1.com
```

```
127.0.0.1 www.test2.com
```

服务器配置

```
<VirtualHost *:80>
    DirectoryIndex index.html index.htm //定义索引页;
    ServerAdmin webmaster@test1.com //管理员邮箱;
    DocumentRoot /usr/local/apache/htdocs/test1 //定义主目录,需要新建 test1 目录;
    ServerName www.test1.com //定义域名;
    ErrorLog logs/www.test1.com-error_log
    CustomLog logs/www.test1.com-access_log combined
</VirtualHost>
<VirtualHost *:80>
    DirectoryIndex index.html index.htm //定义索引页;
    ServerAdmin webmaster@test2.com //管理员邮箱;
    DocumentRoot /usr/local/apache/htdocs/test2 //定义主目录;
    ServerName www.test2.com //定义域名;
    ErrorLog logs/www.test2.com-error_log
    CustomLog logs/www.example2.com-access_log combined
</VirtualHost>
```



由于 \* (星号) 匹配所有的地址, 所以主服务器不会接收任何的请求。因为 `www.test1.com` 首先出现在配置文件中, 所以它拥有最高优先级, 可以认为是默认或首要服务器。这意味着如果一个接受的请求不能与某个 `ServerName` 指令相匹配, 它将会由第一个 `VirtualHost` 所响应。

注意: 可以用确定的 IP 地址来取代 \*。在这种情况下, `VirtualHost` 的参数必须与 `NameVirtualHost` 的参数相符:

```
NameVirtualHost 122.122.122.122
<VirtualHost 122.122.122.122>
```

不论如何, 当 IP 地址无法确定的时候, 使用 \* 是很方便的。例如, ISP 给你配置的是动态 IP 地址, 而你使用了某种动态域名解析系统时。因为 \* 匹配任何 IP 地址, 所以在这样的情况下, 不论 IP 地址如何变化, 你都不需要另外进行配置。

上述配置就是你在绝大多数情况下使用基于域名的虚拟主机时将要用到的。事实上, 仅在一种情况下这样的配置不会让你满意: 当你想为不同的 IP 地址或是端口提供不同的内容时。

## 2) 在不同的端口上运行不同的站点

如果想让同一 IP 的不同端口响应多个域名。可以借助在 “`NameVirtualHost`” 标签中定义端口这样的方法来达到这个目的。如果想使用不带 `NameVirtualHost name:port` 的 `<VirtualHost name:port>` 或是直接用 `Listen` 指令, 那么配置将无法生效。

服务器配置

```
Listen 80
Listen 8080
NameVirtualHost 127.0.0.1:80
NameVirtualHost 127.0.0.1:8080
<VirtualHost 127.0.0.1:80>
ServerName www.test1.com
DocumentRoot /usr/local/apache/htdocs/test1-80
</VirtualHost>
<VirtualHost 127.0.0.1:8080>
ServerName www.test1.com
DocumentRoot /usr/local/apache/htdocs/test1-8080
</VirtualHost>
<VirtualHost 127.0.0.1:80>
ServerName www.test2.com
DocumentRoot /usr/local/apache/htdocs/test2-80
</VirtualHost>
<VirtualHost 127.0.0.1:8080>
ServerName www.test2.com
DocumentRoot /usr/local/apache/htdocs/test2-8080
</VirtualHost>
```

### 3) 建立基于 IP 的虚拟主机

一台服务器有两个 IP 地址 (112.112.112.111 和 112.112.112.112) 分别对应域名 www.test1.com 和 www.test2.com 的服务配置如下:

服务器配置

```
Listen 80
<VirtualHost 112.112.112.111>
DocumentRoot /usr/local/apache/htdocs/test1
ServerName www.test1.com
</VirtualHost>
<VirtualHost 112.112.112.112>
DocumentRoot /usr/local/apache/htdocs/test2
ServerName www.test2.com
</VirtualHost>
```

如果存在主服务器 (main server), 那么对没有出现在任一个 <VirtualHost> 指令中的请求 (比如, 对 localhost 的请求) 都会由主服务器来响应。

### 4) 混用基于端口和基于 IP 的虚拟主机

如果服务器有两个 IP 地址 (112.112.112.111 和 112.112.112.112) 分别对应域名 www.test1.com 和 www.test2.com。对每个域名, 都要求在 80 端口和 8080 端口发布网站。可以这样配置:

服务器配置

```
Listen 112.112.112.111:80
Listen 112.112.112.111:8080
Listen 112.112.112.112:80
Listen 112.112.112.112:8080
<VirtualHost 112.112.112.111:80>
    DocumentRoot /usr/local/apache/htdocs/test1-80
    ServerName www.test1.com
</VirtualHost>
<VirtualHost 112.112.112.111:8080>
    DocumentRoot /usr/local/apache/htdocs/test1-8080
    ServerName www.test1.com
</VirtualHost>
<VirtualHost 112.112.112.112:80>
    DocumentRoot /usr/local/apache/htdocs/test1-80
    ServerName www.test2.com
</VirtualHost>
<VirtualHost 112.112.112.112:8080>
    DocumentRoot /usr/local/apache/htdocs/test1-8080
    ServerName www.test2.com
</VirtualHost>
```

## 5) 混用基于域名和基于 IP 的虚拟主机

如果你想在一些地址上配置基于域名的虚拟主机而在另外一些配置基于 IP 的虚拟主机。

服务器配置

```
Listen 80
NameVirtualHost 112.112.112.111
<VirtualHost 112.112.112.111>
DocumentRoot /usr/local/apache/htdocs/test1
ServerName www.test1.com
</VirtualHost>
<VirtualHost 112.112.112.111>
DocumentRoot /usr/local/apache/htdocs/test2
ServerName www.test2.com
</VirtualHost>
<VirtualHost 112.112.112.111>
DocumentRoot /usr/local/apache/htdocs/test3
ServerName www.test3.com
</VirtualHost>
IP-based
<VirtualHost 112.112.112.112>
DocumentRoot /usr/local/apache/htdocs/test4
ServerName www.test4.com
</VirtualHost>
<VirtualHost 112.112.112.113>
DocumentRoot /usr/local/apache/htdocs/test5
ServerName www.test5.com
</VirtualHost>
```

## 6) 使用 \_default\_ 虚拟主机

为所有端口配置 \_default\_ 虚拟主机, 这样配置, 以获得任何指向没指定的 IP 地址和端口的请求。比如说: 一个没被任何虚拟主机使用的地址/端口对。

服务器配置

```
<VirtualHost_default_: * >
DocumentRoot /usr/local/apache/htdocs/default
</VirtualHost>
```

使用这样一个应用通配符端口的默认虚拟主机可以有效地防止请求被主服务器接收。



如果一个地址、端口对已经被一个基于域名的虚拟主机使用,那么默认虚拟主机决不会处理发向这个地址、端口的请求。如果一个请求的 Host 头中包含未知信息,或者干脆就没有,那么它会被第一个基于域名的虚拟主机(就是在配置文件中首先出现的使用了那个地址、端口对的虚拟主机)处理。

可以用 AliasMatch 或 RewriteRule 来重写任何请求,使它指向一个简单信息页面(a single information page)(或脚本)。

## 7) 为不同的端口部署\_default\_虚拟主机

与第一种一样,但我们想让服务器侦听很多端口而第二个\_default\_虚拟主机单独侦听 80 端口。

服务器配置

```
<VirtualHost_default_:80>
DocumentRoot/usr/local/apache/htdocs/default80
...
</VirtualHost>
<VirtualHost_default_:*>
DocumentRoot/usr/local/apache/htdocs/default
...
</VirtualHost>
```

侦听 80 端口的默认虚拟主机(必须出现在所有使用通配符端口的虚拟主机之前)会捕获所有发向一个没有指定的 IP 地址的请求。主服务器将不会用于响应任何请求。

## 8) 为一个端口配置\_default\_虚拟主机

如果我们只想在 80 端口上建立唯一的一个默认虚拟主机,我们应该这样配置:

服务器配置

```
<VirtualHost_default_:80>
DocumentRoot/usr/local/apache/htdocs/default
...
</VirtualHost>
```

发向一个未进行设定 80 端口地址的请求将会为这个虚拟主机响应;而发向未进行设定其他端口地址的请求为主服务器响应。

## 9) 将一个基于域名的虚拟主机移植为一个基于 IP 的虚拟主机

如果一个具有 www.example2.org 域名的虚拟主机(就是基于域名配置示例中的第二个)得到了自己的 IP 地址。为了避免一些域名服务器或代理服务器在移植期间仍对这个域名做老的解析,我们可以采用一种过渡方法:同时提供新旧两个 IP 地址的解析。

达到这个目的很简单。因为我们只要简单的把新地址(112.112.112.112)加入 Virtual-

Host 指令就行了。

服务器配置

```
Listen 80
ServerName www.test1.com
DocumentRoot /usr/local/apache/htdocs/test1
NameVirtualHost 112.112.112.111
<VirtualHost 112.112.112.111 112.112.112.112>
DocumentRoot //usr/local/apache/htdocs/test2
ServerName www.test2.com
...
</VirtualHost>
<VirtualHost 112.112.112.111>
DocumentRoot /usr/local/apache/htdocs/test3
ServerName www.test3.com
ServerAlias * .test3.com
...
</VirtualHost>
```

现在这个虚拟主机就可以用新地址(基于 IP 的虚拟主机)和旧地址(基于域名的虚拟主机)同时进行访问了。

## 第 7 节 认证、授权和访问控制

如果网络站点上有敏感信息或只希望为一个小群体所访问,这里讲述的方法能确保使用户只能访问允许被访问的资源。

### 1. 准备工作

本节中讨论的指令应该放入主服务器的配置文件(通常在 <Directory> 段)或者针对单个目录的配置文件(.htaccess files)中。

如果需要使用 .htaccess 文件,那么必须设置服务器以允许在这些文件中使用认证指令,即,用 AllowOverride 指令指定哪些指令在针对单个目录的配置文件中有效。

这里讨论认证,可以对 AllowOverride 这样设置:

```
AllowOverride AuthConfig
```

如果你希望把这些指令直接写入主服务器配置文件,当然就需要有对配置文件的写权限。你需要对服务器的目录结构有所了解,以确定某些文件的位置。

## 2. 启用认证

先介绍用密码来保护服务器上的目录。首先需要建立一个密码文件。这个文件可以放在不能被网络访问的位置,以避免被下载。例如,如果 `/usr/local/apache/htdocs` 以外的空间不能被网络访问,那么可以考虑把密码文件放在 `/usr/local/apache/passwd` 目录中。

Apache 在其安装目录的 `bin` 子目录中提供了叫 `htpasswd` 的工具,以建立密码文件,可以这样使用:

```
[root@ lupa/]# mkdir /usr/local/apache/passwd
[root@ lupa/]# htpasswd-c /usr/local/apache/passwd/. htpasswd user
```

`htpasswd` 会要你输入密码,并要求重新输入以确认:

```
[root@ lupa/]# htpasswd-c /usr/local/apache/passwd/. htpasswd user
New password: mypassword
Re-type new password: mypassword
Adding password for user user
```

下来是对目录 `lamp` 进行保护,将以下内容写入 `httpd.conf` 文件中。

```
< Directory /usr/local/apache/htdocs/lamp >
    Allowoverride none
    order allow,deny
    Allow from all
    AuthType basic
    Authname "Input user and password!"
    Authuserfile /usr/local/apache/htdocs/lamp/. htpasswd
    Require valid-user
< /Directory >
```

其中, `AuthType` 指令选择了对用户实施认证的方法,最常用的是由 `mod_auth` 提供的 `Basic`。`Basic` 认证方法并不加密来自用户浏览器的密码,因此,不应该用于高度敏感的数据。Apache 在最近的版本中还有另一种更安全的认证方法,即由 `mod_auth_digest` 提供的 `AuthType Digest`。

`AuthName` 指令设置了使用认证的领域,它起两个作用,首先,此领域说明会出现在显示给用户的密码提问对话框中,其次帮助客户端程序确定应该输入哪个密码。所以,如果一个用户已经在“`Restricted Files`”领域通过了认证,则客户端就可以尝试使用“`Restricted Files`”的密码来访问同一个服务器的其他任何领域,从而使多个受限领域共享密码,以避免用户重复输入。当然,考虑到安全,如果服务器变了,客户端始终会要求重新输入密码。

`AuthUserFile` 指令设置了密码文件,也就是刚才我们已经用 `htpasswd` 建立的。如果用户过多,则认证速度会很慢,因为对每个请求都必须搜索这个纯文本文件,对此,Apache 还支

持把用户信息存入快速数据库, `mod_auth_dbm` 模块提供了指令 `AuthDBMUserFile`, 并可以用 `dbmmanage` 程序建立和操作这些数据库。Apache 模块数据库中还提供了其他许多认证选项。

最后, `Require` 指令设置了允许访问的用户。

### 3. 允许多人访问

上述指令只允许一个叫 `user` 的用户访问这个目录, 但是多数情况下, 都需要允许多人访问, 所以就要用到 `AuthGroupFile`。

首先, 需要建立一个组文件以确定组中的用户。其格式很简单, 可以用你喜欢的编辑器建立, 例如:

```
[root@ lupa/]# vi/usr/local/apache/passwd/groups
```

把以下内容, 加入到 `groups` 中:

```
lampgroup:user user1 user2 user3 user4
```

它只是一个每组一行的用空格分隔的组成员列表。

向已有的密码文件中增加一个用户, 可以输入:

```
[root@ lupa/]# htpasswd/usr/local/apache/passwd/. htpasswd user1
```

程序的提示和上面的一样, 但是, 它会附加到已有的文件中, 而不是建一个新的(参数 `-c` 可以强制建立新的密码文件)。

现在, 需要修改 `htaccess` 文件如这样:

```
AuthType basic
AuthName "Input user and password!"
AuthUserFile /usr/local/apache/passwd/. htpasswd
AuthGroupFile /usr/local/apache/passwd/groups
Require group lampgroup
```

如此, `lampgroup` 组中的成员都在密码文件中有一个相应的记录, 从而允许他们输入正确的密码以进入。

除了建立组文件, 还有另一种途径允许多人访问, 即使用如下指令:

```
Require valid-user
```

### 4. 测试举例

现在我们举个实例, 用 Apache 服务器实现用户验证。

### 1) 建立需要用户验证的目录

我们在 `/usr/local/apache/htdocs/` (apache 的主页根目录) 下建立一个 `aaa` 目录, 现在我们要对 `aaa` 进行认证和授权访问。

```
[root@lupa/]# mkdir /usr/local/apache/htdocs/aaa
```

### 2) 第2步: 编辑 `httpd.conf` 主件

使用 `vi` 打开主配置文件 `httpd.conf`

```
[root@lupa/]# vi /usr/local/apache/conf/httpd.conf
```

添加以下内容

```
<Directory /usr/local/apache/htdocs/aaa> //验证的目录;
    AllowOverride none //表示进行身份验证;
    order allow,deny
    allow from all //信任任何地方的 IP;
    authtype basic //指定认证类型:basic。还有几种认证类型,如:MD5;
    authname "Input user and password!" //验证时弹出的窗口上所显示的内容;
    authuserfile /usr/local/apache/passwd/.htpasswd //密码文件存放的地方;
    require valid-user //指定哪些用户或组才能被授权访问;
</Directory>
```

### 3) 创建 Apache 的验证用户

```
[root@lupa/]# mkdir /usr/local/apache/passwd
[root@lupa/]# htpasswd -c /usr/local/apache/passwd/.htpasswd user
```

第一次创建用户要用到 `-c` 参数,第二次添加用户时,就不用 `-c` 参数。现在就创建了一个用户为 `user`,而密码是在你执行了这个命令后,系统会提示你输入密码。如果想修改密码,可以如下操作:

```
[root@lupa/]# htpasswd -m .htpasswd user
```

### 4) 重启 Apache 服务

重新启动 Apache 服务,然后访问 `http://127.0.0.1/aaa`。如果顺利的话,应该能看到一个用户验证的弹出窗口,只要填入第3步创建的用户名和密码就行。



## 5. 访问控制

### 1) 访问控制的配置指定

- Order

用于指定执行允许访问和拒绝访问规则的先后顺序。

Order 指令有两种形式：

- ◆ Order Allow, Deny

在执行拒绝访问规则之前先执行允许访问规则,默认情况下将会拒绝所有没有明确被允许的客户。

- ◆ Order Deny, Allow

在执行允许访问规则之前先执行拒绝访问规则,默认情况下将会允许所有没有明确被拒绝的客户。

- Deny

定义拒绝访问列表。

- Allow

定义允许访问列表。

Deny 和 Allow。Deny 和 Allow 指令的后面需要跟访问列表,访问列表可以使用如下的几种形式：

- ◆ All:表示所有客户。
- ◆ 域名:表示域内的所有客户,如 abc.net。
- ◆ IP 地址:可以指定完整的 IP 地址或部分 IP 地址。
- ◆ 网络/子网掩码:如 192.168.1.0/255.255.255.0。
- ◆ CIDR 规范:如 192.168.1.0/24。

### 2) 举例说明

```
Order Allow, Deny
Allow from all
Deny from www.abc.com
```

意为接受所有的人访问 Apache 服务器,但不允许来自 www.abc.com 的任何访问。

```
Order Deny, Allow
Deny from all
Allow from abc.com.cn
```

意为不接受所有人访问,但允许 abc.com.cn 网站的来访。

# MySQL 安装和使用

MySQL 是一个开放源码的多用户、多线程 SQL 数据库服务器软件。开发者为瑞典 MySQL AB 公司。它能让你通过编程语言如 PHP 来存储或取回数据。你可以花最少的努力快速而高效地存储多种类型的数据,如布尔类型、文本类型、整数类型、图像类型、二进制数据和 BLOB 数据。使用数据库对于创建动态网站是十分重要的。

## 第1节 下载 MySQL

MySQL 的安装可以从源代码或者预编译的二进制文件开始。如果从源代码开始,虽然控制权更大,但所花时间也较长。安装预编译的二进制文件则非常方便,而且还可以找到现成的用于许多操作系统的安装文件。

不管选择源代码还是二进制文件,都应该访问 MySQL 主页:<http://www.mysql.com> 了解相关资讯,查阅有关最新开发情况的新闻,找到离自己最近的一个镜像站点。所有的 MySQL 文件都通过遍布全球的许多服务器提供。选择离自己最近的站点有助于缓解网络负担,同时缩短下载时间。本教材推荐 <http://mirror.lupaworld.com/mysql>。

## 第2节 安装 MySQL

### 1. 准备安装

在开始安装 MySQL 之前,必须要明白的几个问题:

#### 1) 要安装哪个版本

这通常需要在最新的稳定版本和最新的开发版本之间、安装在哪个类型的服务器上做选择。一般我们推荐使用最新的稳定版本,当前的最新的稳定版本是 MySQL 5.0.X。

#### 2) 准备以超级用户 (root) 还是以其他用户的身份安装 MySQL

MySQL 不需要 root 权限就可以运行,但是以 root 身份安装可以为系统中的每位用户提



供一个副本。如果没有 root 访问权,你就必须把它安装在自己的主目录下。但是即使以 root 用户安装,也应该以不同用户的身份运行。这样,通过将数据文件设置为只能由特定用户读取,就可以使数据库的所有数据都受到保护。此外,如果数据库的安全受到了威胁,攻击者也只能访问 MySQL 特定的用户账号,而该账号没有超出数据库的权限。

## 2. Linux 平台安装

MySQL 适用于所有 Unix/Linux 平台。这里仅介绍在 Linux 下安装二进制和源代码版本的一般步骤。这也可以用来指导别的操作系统下的安装。

### 1) 二进制 (Tarball) 版本

例如选择在 Linux 服务器上安装 MySQL 5.0.24 请到 LUPA 镜像下载:

<http://mirror.lupaworld.com/mysql/Downloads/>。

建议为 MySQL 管理创建一个用户和组,由该用户运行 MySQL 服务器并执行管理任务,但也可以用超级用户。

第一步是创建一个用户运行 MySQL 服务器。在 linux 下可以用 useradd 和 groupadd 实用工具来完成。一般情况下,你需要是 root 用户才能成功地完成这些任务,所以在做其他事情之前,利用 su 命令成为 root 用户。

```
[root@ lupa/]# su
[root@ lupa/]# groupadd mysql //创建 mysql 用户;
[root@ lupa/]# useradd-g mysql mysql //创建 mysql 组;
```

选择 MySQL 软件要安装的位置,并将当前目录转换到该目录。在这个例子中,我们将安装到 /usr/local/, 这是 MySQL 软件的标准安装位置。当然也可以安装到任何你喜欢的目录。

```
[root@ lupa/]# cd/usr/local
```

现在解开软件包。

```
[root@ lupa local]# tar zxvf/tmp/mysql-standard-5.0.24-linux-i686.tar.gz
```

现在应该查看一下新目录。

```
[root@ lupa local]# ls-ld.mysql *
```

下一步是创建一个符号链接,以便安装能够指向 /usr/local/mysql。

```
[root@ lupa local]# ln-s mysql-standard-5.0.24-linux-i686 mysql //创建软连接
[root@ lupa local]# ls-ld mysql *
```

现在到 MySQL 目录查看一下。

```
[root@ lupa local]# cd mysql
[root@ lupa mysql]# ls -l
```

虽然软件已经安装好,但还有几项配置任务要完成。运行 `scripts/mysql_install_db` 创建 MySQL 许可表。

```
[root@ lupa mysql]# scripts/mysql_install_db//创建 mysql 表;
Preparing db table
Preparing host table
Preparing user table
Preparing func table
Preparing tables_priv table
Preparing columns_priv table
Installing all prepared tables
060731 14:13:14. /bin/mysql:shutdown Complete
```

设置二进制文件所有权,从而使之归 root 所有,并属于在前面创建的 MySQL 管理员组。

```
[root@ lupa/]# chown-R root/usr/local/mysql
[root@ lupa/]# chgrp-R mysql/usr/local/mysql
```

将数据目录所有权设置为先前创建的 MySQL 管理用户。

```
[root@ lupa/]# chown-R mysql/usr/local/mysql/data
```

MySQL 现在已经安装完成并准备运行。启动服务器并运行 `safe_mysqld`。

```
[root@ lupa mysql]# ./bin/safe_mysql-user = mysql &
```

## 2) 二进制 (RPM) 版本

在 Linux 系统下安装 MySQL,推荐通过 RedHad 包管理程序。

完整的 MySQL RPM 版本的文件:

文件名	描述
MySQL-server-version. i386. rpm	MySQL 服务端软件
MySQL-client-version. i386. rpm	MySQL 客户软件
MySQL-bench-version. i386. rpm	MySQL 测试和基准程序;需 perl 和 mysql rpm
MySQL-devel-version. i386. rpm	编写其他 MySQL 客户的函数库和包含文件
MySQL-share-version. i386. rpm	MySQL 客户共享函数库

安装 RPM 版本的过程比较方便。首先,获取要安装的 RPM。至少要安装服务器和客户软件。其次,使用 RPM 实用工具安装程序包。

在这个例子中使用的软件,你可以到 <http://mirror.lupaworld.com/mysql/downloads/>

mysql/5.0.html 下载 RPM 文件到/tmp 下。

执行下面的命令序列进行安装：

```
[root@ lupa/]# rpm-i/tmp/MySQL-server-standard-5.0.24-0-i386.rpm
[root@ lupa/]# rpm-i/tmp/MySQL-client-standard-5.0.24-0-i386.rpm
[root@ lupa/]# rpm-i/tmp/MySQL-bench-standard-5.0.24-0-i386.rpm
[root@ lupa/]# rpm-i/tmp/MySQL-devel-standard-5.0.24-0-i386.rpm
[root@ lupa/]# rpm-i/tmp/MySQL-share-compat-5.0.24-0-i386.rpm
```

RPM 将在/etc/rc.d 中创建合适的项,从而在系统引导和关闭时自动启动和停止服务器。RPM 也可启动 MySQL 服务器,所以当 RPM 安装完成后,就可以使用 MySQL 了。

RPM 版本将文件放置在不同于 tarball 版本的位置。检查 RPM 以确定文件存放的位置,在此使用 RPM 查询选择。

```
[root@ lupa/]# rpm-gpl MySQL-version.i386.rpm
```

如果忽略 RPM 文件名,要确定安装的目录,可以查询 RPM 数据库。

```
[root@ lupa/]# rpm-ql MySQL-version
```

还要注意 RPM 将数据放在/var/lib/data,而不是/usr/local/mysql/data 下。

### 3) 源代码安装

建议你使用源代码安装,源代码版本安装不同于二进制版本。因为从源代码编译是个复杂的过程,有许多可能出现的错误,具体看你的操作系统、所需配置以及工具集等。

对于我们的例子,假定 mysql-5.0.24.tar.gz 已经下载到/root 下。

与二进制安装一样,第一次是创建运行 MySQL 服务器的用户。

```
[root@ lupa/]# groupadd mysql //建立用户组;
[root@ lupa/]# useradd-g mysql mysql //建立用户;
```

解开源文件包。

```
[root@ lupa ~]# tar zxvf mysql-5.0.24.tar.gz
```

进入新创建的 MySQL 目录。

```
[root@ lupa ~]# cd mysql-5.0.24
```

现在使用 configure 脚本来配置所建立的 MySQL。在这个例子中,我们使用了 prefix 选项,从而将安装位置设置为/usr/local/mysql。

```
[root@ lupa mysql-5.0.24]# ./configure--prefix=/usr/local/mysql
```

configure 实用工具提供了许多选项,要得到可用的选项的更多帮助,请运行:

```
[root@ lupa mysql-5.0.24]#. /configure-help
```

完成就可以构建二进制文件。

```
[root@ lupa mysql-5.0.24]# make //编译需要一段时间;
```

如果一切顺利,你现在就得到了 MySQL 的二进制版本,最后需要做的事情是安装它。

```
[root@ lupa mysql-5.0.24]# make install
```

现在软件安装好,还要几项配置任务要完成,运行 `mysql_install_db` 来创建 MySQL 许可表。

```
[root@ lupa mysql-5.0.24]# cp support-files/my-medium.cnf/etc/my.cnf
```

```
[root@ lupa mysql-5.0.24]# cd/usr/local/mysql
```

```
[root@ lupa mysql]# bin/mysql_install_db--user = mysql
```

```
[root@ lupa mysql]# chown-R root. //注意后面有个.;
```

```
[root@ lupa mysql]# chown-R mysql var
```

```
[root@ lupa mysql]# chgrp-R mysql. //注意后面有个.;
```

启动服务需运行 `safe mysqld`。

```
[root@ lupa mysql]# ./bin/safe_mysqld--user = mysql &
```

停止服务需运行 `mysqladmin`。

```
[root@ lupa mysql]# ./bin/mysqladmin shutdown
```

如果需要开机启动 MySQL,则进行如下操作:

编辑文件 `rc.local`。

```
[root@ lupa ~]# vi/etc/rc.local
```

在配置文件里加入下列一行。

```
/bin/sh-c '/usr/local/mysql/; ./bin/mysqld_safe &'
```

保存退出即可。

### 第3节 配置 MySQL

MySQL 的配置需要 MySQL 服务器进程、`mysqld` 及其几个客户进程,如 MySQL 命令实用程序。确切地说,要使用命令行选项、配置文件和环境变量的组合来进行配置。几乎任何配置项都用这三种机制来进行管理。

因为可以使用多种方式定义选项,所以 MySQL 提供了一个定义如何解决冲突的内置优

选顺序:命令行选项、配置选项、环境变量选项。处理 MySQL 选项最简单、最常用的方法就是使用配置文件。配置文件可以让你将所有选项写入一个文件中,这样不必每次都运行命令或日志来将它们指定到一台机器中。

## 1. 配置文件定位

在 Linux 系统中,决定 MySQL 服务器运行特性的 MySQL 主配置文件可以存放于多个位置。MySQL 服务器在启动时会到以下四个位置去查找主配置文件。通常主配置文件名为 `my.cnf`。

### 1) `/etc/my.cnf` 文件

MySQL 查找的第一个位置是全局选项文件。

### 2) `DATADIR/my.cnf` 文件

`DATADIR` 是 MySQL 服务器保存其数据文件的目录。这个配置文件中只应用于给定服务器的配置参数。

### 3) 通过 `--defaults-extra-file = filename` 命令行选项指定的位置

这个命令行选项使 MySQL 服务器或客户实用程序在任意指定位置查找配置文件。

### 4) `$ HOME/.my.cnf` 文件

`$ HOME` 是保存当前用户主目录的环境变量。用户主目录下的配置文件用来为各用户保存其特定的选项,即大多数客户选项的位置。

## 2. 配置文件内容

下面是一个配置文件的样例。MySQL 配置文件:

```
[client]
# password = your_password
port = 3306
socket = /tmp/mysql.sock
# Here follows entries for some specific programs
# The MySQL server
[mysqld]
port = 3306
socket = /tmp/mysql.sock
skip-locking
key_buffer = 16M
```



```
max_allowed_packet = 1M
table_cache = 64
sort_buffer_size = 512K
net_buffer_length = 8K
read_buffer_size = 256K
read_rnd_buffer_size = 512K
myisam_sort_buffer_size = 8M
# Don't listen on a TCP/IP port at all. This can be a security enhancement,
# skip-networking
# Replication Master Server ( default )
# binary logging is required for replication
log-bin = mysql-bin
# required unique id between 1 and 2^32-1
# defaults to 1 if master-host is not set
# but will not function as a master if omitted
server-id = 1
# Replication Slave ( comment out master section to use this )
# To configure this host as a replication slave, you can choose between
# two methods:
# 1) Use the CHANGE MASTER TO command ( fully described in our manual ) -
# the syntax is:
# CHANGE MASTER TO MASTER_HOST = < host > , MASTER_PORT = < port > ,
# MASTER_USER = < user > , MASTER_PASSWORD = < password > ;
# where you replace < host > , < user > , < password > by quoted strings and
# < port > by the master's port number ( 3306 by default ).
# Example:
# CHANGE MASTER TO MASTER_HOST = ' 125.564.12.1 ' , MASTER_PORT = 3306 ,
# MASTER_USER = ' joe ' , MASTER_PASSWORD = ' secret ' ;
# OR
# 2) Set the variables below. However, in case you choose this method, then
# start replication for the first time ( even unsuccessfully, for example
# if you mistyped the password in master-password and the slave fails to
# connect ), the slave will create a master.info file, and any later
# change in this file to the variables' values below will be ignored and
# overridden by the content of the master.info file, unless you shutdown
# the slave server, delete master.info and restart the slaver server.
# For that reason, you may want to leave the lines below untouched
# ( commented ) and instead use CHANGE MASTER TO ( see above )
```

```
# required unique id between 2 and 2^32-1
# ( and different from the master)
# defaults to 2 if master-host is set
# but will not function as a slave if omitted
#server-id = 2
# The replication master for this slave-required
#master-host = <hostname>
# The username the slave will use for authentication when connecting
# to the master-required
#master-user = <username>
# The password the slave will authenticate with when connecting to
# the master-required
#master-password = <password>
#
# The port the master is listening on.
# optional-defaults to 3306
#master-port = <port>
#
# binary logging-not required for slaves, but recommended
#log-bin = mysql-bin
# Point the following paths to different dedicated disks
#tmpdir = /tmp/
#log-update = /path-to-dedicated-directory/hostname
# Uncomment the following if you are using BDB tables
#bdb_cache_size = 4M
#bdb_max_lock = 10000
# Uncomment the following if you are using InnoDB tables
#innodb_data_home_dir = /usr/local/mysql/var/
#innodb_data_file_path = ibdata1:10M:autoextend
#innodb_log_group_home_dir = /usr/local/mysql/var/
#innodb_log_arch_dir = /usr/local/mysql/var/
# You can set .._buffer_pool_size up to 50-80 %
# of RAM but beware of setting memory usage too high
#innodb_buffer_pool_size = 16M
#innodb_additional_mem_pool_size = 2M
# Set .._log_file_size to 25 % of buffer pool size
#innodb_log_file_size = 5M
#innodb_log_buffer_size = 8M
```



```
#innodb_flush_log_at_trx_commit = 1
#innodb_lock_wait_timeout = 50
[mysqldump]
quick
max_allowed_packet = 16M
[mysql]
no-auto-rehash
# Remove the next comment character if you are not familiar with SQL
#safe-updates
[isamchk]
key_buffer = 20M
sort_buffer_size = 20M
read_buffer = 2M
write_buffer = 2M
[myisamchk]
key_buffer = 20M
sort_buffer_size = 20M
read_buffer = 2M
write_buffer = 2M
[mysqlhotcopy]
interactive-timeout
```

配置文件的一些参数的详解：

### 1) back\_log

要求 MySQL 具有的连接数量。当主要 MySQL 线程在一个很短时间内得到非常多的连接请求,它就起作用,然后主线程检查连接并且启动一个新线程。

back\_log 值指在 MySQL 暂时停止回答新请求之前的短时间内,允许有多少个请求可以被存在堆栈中。如果期望在一个短时间内会有很多连接,则需要增加它的值。注意,back\_log 的值不能高于操作系统的最大值。默认数值是 50。

### 2) interactive\_timeout

关闭服务器之前在一个交互连接上等待行动的秒数。一个交互的客户被定义为对 mysql\_real\_connect() 使用 CLIENT\_INTERACTIVE 选项的客户。默认数值是 28800。

### 3) key\_buffer\_size

索引块为缓冲,并且被所有线程共享。key\_buffer\_size 是用于索引块的缓冲区大小,增加它可得到更好处理的索引(对所有读和多重写)。如果该值太大,系统将开始换页并且变慢。默认数值是 8388600(8M)。

#### 4) max\_connections

允许同时访问的数量,即 mysqld 要求的文件描述符的数量。若访问量超过设定值,我们将会看到 Too many connections 错误。默认数值是 100。

#### 5) record\_buffer

每个进行顺序扫描的线程,都为其扫描的每张表分配一个缓冲区。默认数值是 131072 (128K),我把它改为 16773120 (16M)。

#### 6) sort\_buffer

为每个需要进行排序的线程分配一个缓冲区,增加这个值可以加速 ORDER BY 或 GROUP BY 操作。默认数值是 2097144 (2M)。

#### 7) table\_cache

为所有线程打开表的数量。增加该值能增加 mysqld 要求的文件描述符的数量。MySQL 对每个唯一打开的表需要 2 个文件描述符。默认数值是 64。

#### 8) thread\_cache\_size

可以复用的保存在线程中的数量。如果有的话,新的线程会从缓存中取得,当断开连接的时候如果有空间,则客户的线程保留在缓存中。如果有很多新的线程,为了提高性能的话可以改变这个变量。通过比较 Connections 和 Threads\_created 状态的数据,可以看到这个变量的作用。

#### 9) MySQL 的搜索功能

用 MySQL 进行搜索,要求能不分大小写,又能用中文进行搜索。需要启动 mysqld 时指定 --default-character-set = gb2312。

#### 10) wait\_timeout

服务器在关闭它之前在一个连接上等待行动的秒数。默认数值是 28800。

### 3. 配置 MySQL 随机启动

因为 MySQL 服务器是服务进程,所以应当在机器启动时自动启动,并在计算机关闭时完全关闭。如何达到这些目标很大程度上依赖于操作系统。下面写一个 Linux 系统下的样例:

Mysql.server 脚本支持 MySQL 在 linux 系统上的启动和关闭,在安装目录下的 support-files 下可以找到这脚本,它通常在 /usr/local/mysql/support-files 下。

注意:如果是使用 RPM 包在 Linux 上安装 MySQL,那么 mysql.server 可能已经安装到机器上,如存在这个文件 /etc/rc.d/init.d/mysql,则说明 MySQL 可以自动启动和关闭了。

将 mysql-server 安装到 Linux 系统的过程为：

```
[root@ lupa mysql-5.0.24]# cp support-files/mysqld.server/etc/rc.d/init.d
[root@ lupa mysql-5.0.24]# ln -s /etc/rc.d/init.d/mysql.server/etc/rc.d/rc3.d/
s99mysql
[root@ lupa mysql-5.0.24]# ln -s /etc/rc.d/init.d/mysql.server/etc/rc.d/rc0.d/
s01mysql
```

第一行将脚本复制到初始脚本目录/etc/rc.d/init.d。第二条命令创建了到该脚本的链接,使得 Linux 在系统进入运行级别 3 时执行该脚本。最后一行为运行级别 0 创建 mysql.server 链接。运行级别 0 执行系统终止。当系统关闭时执行/etc/rc.d/rc0.d 脚本。

## 4. 增加安全性

MySQL 数据库的安全配置必须从两个方面入手,系统内部安全和外部网络安全。

### 1) 系统内部安全

MySQL 的授权表给数据库的访问提供了灵活的权限控制,但是如果本地用户拥有对库文件的读权限的话,攻击者只需把数据库目录打包,然后拷到自己本机的数据目录下就能访问窃取的数据。所以 MySQL 所在的主机的安全性是最首要的问题,如果主机不安全,被攻击者控制,那么 MySQL 的安全性也无从谈起。其次就是数据目录和数据文件的安全性,也就是权限设置问题。

数据目录的属性是 700,这样比较好,只有启动数据库的用户可以读写数据库文件,保证了本地数据文件的安全。

如果启动 MySQL 数据库的用户是 mysql,那么像如下的目录和文件的是安全的,请注意数据目录及下面的属性:

```
[root@ lupa/]# ls -l /usr/local/mysql
total 40
drwxrwxr-x 2 root root 4096 Feb 27 20:07 bin
drwxrwxr-x 3 root root 4096 Feb 27 20:07 include
drwxrwxr-x 2 root root 4096 Feb 27 20:07 info
drwxrwxr-x 3 root root 4096 Feb 27 20:07 lib
drwxrwxr-x 2 root root 4096 Feb 27 20:07 libexec
drwxrwxr-x 3 root root 4096 Feb 27 20:07 man
drwxrwxr-x 6 root root 4096 Feb 27 20:07 mysql-test
drwxrwxr-x 3 root root 4096 Feb 27 20:07 share
drwxrwxr-x 7 root root 4096 Feb 27 20:07 sql-bench
drwx-----4 mysql mysql 4096 Feb 27 20:07 var
[root@ lupa/]# ls -l /usr/local/mysql/var
```

```
total 8
drwx-----2 mysql mysql 4096 Feb 27 20:08 mysql
drwx-----2 mysql mysql 4096 Feb 27 20:08 test
[root@ lupa/]# ls -l /usr/local/mysql/var/mysql
total 104
-rw-----1 mysql mysql 0 Feb 27 20:08 columns_priv.MYD
-rw-----1 mysql mysql 1024 Feb 27 20:08 columns_priv.MYI
-rw-----1 mysql mysql 8778 Feb 27 20:08 columns_priv.frm
-rw-----1 mysql mysql 302 Feb 27 20:08 db.MYD
-rw-----1 mysql mysql 3072 Feb 27 20:08 db.MYI
-rw-----1 mysql mysql 8982 Feb 27 20:08 db.frm
-rw-----1 mysql mysql 0 Feb 27 20:08 func.MYD
-rw-----1 mysql mysql 1024 Feb 27 20:08 func.MYI
-rw-----1 mysql mysql 8641 Feb 27 20:08 func.frm
-rw-----1 mysql mysql 0 Feb 27 20:08 host.MYD
-rw-----1 mysql mysql 1024 Feb 27 20:08 host.MYI
-rw-----1 mysql mysql 8958 Feb 27 20:08 host.frm
-rw-----1 mysql mysql 0 Feb 27 20:08 tables_priv.MYD
-rw-----1 mysql mysql 1024 Feb 27 20:08 tables_priv.MYI
-rw-----1 mysql mysql 8877 Feb 27 20:08 tables_priv.frm
-rw-----1 mysql mysql 428 Feb 27 20:08 user.MYD
-rw-----1 mysql mysql 2048 Feb 27 20:08 user.MYI
-rw-----1 mysql mysql 9148 Feb 27 20:08 user.frm
```

用 root 用户启动远程服务一直是安全大忌,因为如果服务程序出现问题,远程攻击者极有可能获得主机的完全控制权。MySQL 从 3.23.15 版本开始时作了小小的改动,默认安装后服务器要用 mysql 用户来启动,不允许 root 用户启动。如果一定要用 root 用户来启动,必须加上 --user = root 的参数 (./safe\_mysqld--user = root &)。因为 MySQL 中有 LOAD DATA INFILE 和 SELECT……INTO OUTFILE 的 SQL 语句,如果是 root 用户启动了 MySQL 服务器,那么,数据库用户就拥有了 root 用户的写权限。不过 MySQL 还是做了一些限制的,比如 LOAD DATA INFILE 只能读全局可读的文件,SELECT……INTO OUTFILE 不能覆盖已经存在的文件。

本地的日志文件也不能忽视,包括 Shell 的日志和 MySQL 自己的日志。有些用户在本地登陆或备份数据库的时候为了图方便,有时会在命令行参数里直接带了数据库的密码,如:

```
[root@ lupa/]# /usr/local/mysql/bin/mysqldump -uroot -ptest test > test.sql
[root@ lupa/]# /usr/local/mysql/bin/mysql -uroot -ptest
```

这些命令会被 shell 记录在历史文件里,比如 bash 会写入用户目录的 .bash\_history 文件,如果这些文件不慎被读,那么数据库的密码就会泄漏。用户登陆数据库后执行的 SQL

命令也会被 MySQL 记录在用户目录的 .mysql\_history 文件里。如果数据库用户用 SQL 语句修改了数据库密码,也会因 .mysql\_history 文件而泄漏。所以我们在 Shell 登陆及备份的时候不要在 -p 后直接加密码,而是在提示后再输入数据库密码。

另外这两个文件我们也应该不让它记录我们的操作,以防万一。

```
[root@ lupa/]# cd/usr/local/mysql/bin/mysql
[root@ lupa mysql]# rm .bash_history.mysql_history
[root@ lupa mysql]# ln-s/dev/null.bash_history
[root@ lupa mysql]# ln-s/dev/null.mysql_history
```

这两条命令把这文件链接到 /dev/null,那么我们的操作就不会被记录到这两个文件里了。

## 2) 外部网络安全

MySQL 数据库安装好以后, user 表是这样的:

Host	User	Password	Select_priv	Grant_priv
localhost	root		Y	Y
lupa	root		Y	Y

其中 lupa 是试验的机器名,所以实际上 MySQL 默认只允许本机才能连接数据库。但是缺省 root 用户口令是空,所以当务之急是给 root 用户加上口令。给数据库用户加口令有三种方法:

- 在 shell 提示符下用 mysqladmin 命令来改 root 用户口令:

```
[root@ lupa mysql]# mysqladmin -u root password lupa
```

这样,MySQL 数据库 root 用户的口令就被改成 lupa 了。

- 用 set password 修改口令:

```
mysql> set password for root@localhost = password('lupa');
```

- 直接修改 user 表的 root 用户口令:

```
mysql> use mysql;
mysql> update user set password = password('test') where user = 'root';
mysql> flush privileges;
```

user 表还有空的匿名用户,虽然它在 Linux 平台下没什么权限,但为了安全起见我们应该删除它:

```
mysql> delete from user where user = '';
```

Windows 版本 MySQL 的 user 表有很大不同,我们看到 Host 字段除了 localhost 还有 %:

```
mysql> delete from user where host = '%';
```



MySQL 的 5 个授权表: user, db, host, tables\_priv 和 columns\_priv 提供非常灵活的安全机制,从 MySQL 3.22.11 开始引入了两条语句 GRANT 和 REVOKE 来创建和删除用户权限,可以方便地限制哪个用户可以连接服务器、从哪里连接以及连接后可以做什么操作。

授权表使用举例:

grant 用于给增加用户和创建权限, revoke 用于删除用户权限。

下面是一些用 grant 增加用户和创建权限的例子:

```
mysql > grant all privileges on *.* to lupa@localhost identified by 'lupa' with grant option;
```

这句增加一个本地具有所有权限的 lupa 用户(超级用户),密码是 lupa。ON 这句中的 \*.\* 意味着“所有数据库、所有表”。with grant option 表示它具有 grant 权限。

下面是用 revoke 删除用户权限的例子:

```
mysql > revoke all on *.* from lupa@localhost;
mysql > delete from user where user = 'test1';
mysql > flush privileges;
```

这些只是 MySQL 授权表的简单使用,更多详细的资料请见 MySQL 提供的手册。

### 3) 一些小技巧

- 如果不慎忘记了 MySQL 的 root 密码,我们可以在启动 MySQL 服务器时加上参数--skip-grant-tables 来跳过授权表的验证(./safe\_mysqld--skip-grant-tables &),这样我们就可以直接登陆 MySQL 服务器,然后再修改 root 用户的口令,重启 MySQL 就可以用新口令登陆了。
- 启动 MySQL 服务器时加上--skip-show-database 使一般数据库用户不能浏览其他数据库。
- 启动 MySQL 服务器时加上--log-slow-queries[ = file] 参数,这样 mysqld 会把 SQL 命令执行时间超过 long\_query\_time 的写入 file 文件。如果没有指定 = file,mysqld 默认会写到数据目录下的 hostname-slow.log。如果只指定了 filename,没有指定路径,那么 mysqld 也会把 filename 写到数据目录下。我们通过这个日志文件可以找出执行时间超长的查询语句,然后尽可能的优化它减轻 MySQL 服务器的负担。
- 如果我们只需本机使用 MySQL 服务,那么我们还可以加上--skip-networking 启动参数使 MySQL 不监听任何 TCP/IP 连接,增加安全性。

## 第 4 节 MySQL 管理

### 1. 使用命令工具

标准 MySQL 安装中自带的命令实用程序,这些实用程序大多是二进制可执行文件,有

的则是外壳脚本。

### 1) myisampack

这个实用程序创建只读的压缩的数据表,myisamchk 这样调用:

```
[root@ lupa mysql]# myisamchk [ options ] tbl_name
```

如果你在一个数据库目录里,你可以这样检查所有数据库表:

```
[root@ lupa mysql]# myisamchk /path/to/database_dir/ * .MYI
```

### 2) MySQL

对 MySQL 用户而言,MySQL 命令行实用程序也许是最重要的可执行程序。它允许交互或在批处理模式下查询数据库。

命令为:

```
[root@ lupa mysql]# mysql db_name
```

连接 MySQL

格式:mysql-h 主机地址-u 用户名-p 用户密码

假设远程主机的 IP 为:110.110.110.110,用户名为 root,密码为 abcd123。则键入以下命令:

```
[root@ lupa mysql]# mysql -h 110.110.110.110-u root -pabcd123 .
```

### 3) mysqladmin

用于执行管理性操作。语法是:

```
[root@ lupa mysql]# mysqladmin [ OPTIONS ] command [ command-option ] command...
```

通过执行 mysqladmin--help,可以得到 mysqladmin 版本所支持的一个选项列表。

目前 mysqladmin 支持下列命令:

命 令	说 明
create databasename	创建一个新数据库
Drop databasename	删除一个数据库及其所有表
extended-status	给出服务器的一个扩展状态消息
Flush-hosts	清空所有缓存的主机
Flush-logs	清空所有日志
Flush-tables	清空所有表
Flush-privileges	再次装载授权表(同 reload)
kill id,id,...	结束 mysql 线程



命 令	说 明
password	新口令,将老口令改为新口令
Ping	检查 mysqld 是否活着
processlist	显示服务器中活跃线程列表
reload	重载授权表
refresh	清空所有表并关闭和打开日志文件
shutdown	关掉服务器
status	给出服务器的简短状态消息
variables	打印出可用变量
version	得到服务器的版本信息

如:再将 root 的密码改为 newpassword。

```
[root@ lupa mysql]# mysqladmin-uroot-poldpassword password newpassword
```

#### 4) mysqlbinlog

该命令将二进制日志中的项目转换为可读的表单或 SQL 语句,其命令格式:

```
[root@ lupa mysql]# mysqlbinlog[ options ] log_file
```

#### 5) mysqlcheck

mysqlcheck 用于检查及修复数据库或表。

用法:

```
mysqlcheck[ OPTIONS ] database[ tables ]
mysqlcheck[ OPTIONS ] --databases DB1[ DB2 DB3... ]
mysqlcheck[ OPTIONS ] --all-databases
```

常用选项

- a 分析指定的表
- c 检查库或表
- r 修复库和表
- o 优化指定的表
- h mysql 服务主机
- u 用户名
- p 密码
- auto-repair 当检查表有错误自动修复
- e 完全检查或修复指定表,时间最长
- m 检查程序,一般所用时间也最短
- q 最快的检查或修复表的方法,用的时间最少

- F 只检查没有正确关闭的表
- f 如果碰到 sql 错误,强制执行

## 6) mysqldump

mysqldump 实用程序从数据库中抽取数据。

用法:

```
[root@ lupa mysql]# mysqldump[ OPTIONS] database[ tables]
[root@ lupa mysql]# mysqldump[ OPTIONS] --databases DB1[ DB2 DB3... ]
[root@ lupa mysql]# mysqldump[ OPTIONS] --all-databases
```

如:

```
[root@ lupa mysql]# mysqldump--opt school > school. sql
```

注释:将数据库 school 备份到 school. sql 文件, school. sql 是一个文本文件,文件名任取,打开看看你会有新发现。

## 7) mysqlhotcopy

mysqlhotcopy 用于为活动的数据库执行一个安全备份。

用法:

```
[root@ lupa mysql]# mysqlhotcopy[ options] db_name/path/to/new/directory
```

如:

```
[root@ lupa mysql]# mysqlhotcopy--checkpoint dbinfo. checkpoint--addtodest db_douzi_
org/var/db_backup
```

--checkpoint dbinfo. checkpoint 这个是指定存放操作记录的数据库/表

--addtodest 增量备份,新的备份自动覆盖原来的

db\_douzi\_org 要备份的数据库名

/var/db\_backup 备份目录

# 2. 建立和管理数据库

## 1) 连接服务器

```
[root@ lupa mysql]# mysql-hlocalhost-uroot-p
```

输入数据库的密码后,会看到以下界面:

```
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 5919160 to server version: 4.0.24-standard-log  
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.  
mysql >
```

## 2) 显示命令

- 显示数据库列表:

```
show databases;
```

- 显示库中的数据表:

```
use mysql  
show tables;
```

- 显示数据表的结构:

```
describe 表名;
```

- 建库:

```
create database 库名;
```

- 建表:

```
use 库名;  
create table 表名(字段设定列表);
```

- 删库和删表:

```
drop database 库名;  
drop table 表名;
```

- 将表中记录清空:

```
delete from 表名;
```

- 显示表中的记录:

```
select * from 表名
```

## 3. 建立和管理用户

MySQL 3.22.11 开始引入两条语句使得这项工作更容易做: GRANT 语句创建 MySQL 用户并指定其权限, 而 REVOKE 语句删除权限。两条语句扮演了 MySQL 数据库的前端角

色,并提供与直接操作这些表的内容不同的另一种方法。

GRANT 语句的语法看上去像这样:

```
GRANT privileges ( columns ) ON what TO user IDENTIFIED BY "password" WITH
GRANT OPTION
```

要使用该语句,你需要填写下列部分:

### 1) Privileges

授予用户的权限,下表列出可用于 GRANT 语句的权限指定符:

权限指定符	权限允许的操作
ALTER	修改表和索引
CREATE	创建数据库和表
DELETE	删除表中已有的记录
DROP	抛弃(删除)数据库和表
INDEX	创建或抛弃索引
INSERT	向表中插入新行
REFERENCE	未用
SELECT	检索表中的记录
UPDATE	修改现存表记录
FILE	读或写服务器上的文件
PROCESS	查看服务器中执行的线程信息或杀死线程
RELOAD	重载授权表或清空日志、主机缓存或表缓存
SHUTDOWN	关闭服务器
ALL	所有;ALL PRIVILEGES 同义词
USAGE	特殊的“无权限”权限

### 2) columns

权限运用的列,它是可选的,并且只能设置列特定的权限。如果命令中有多于一个列,应该用逗号分开它们。

### 3) what

权限运用的级别。权限可以是全局的(适用于所有数据库和所有表)、特定数据库(适用于一个数据库中的所有表)或特定表的。可以通过指定一个 columns 字段的权限是列特定的。

### 4) user

权限授予的用户,它由一个用户名和主机名组成。在 MySQL 中,你不仅指定谁能连接,

还可指定从哪里连接。这允许你让两个同名用户从不同地方连接。MySQL 让你区分他们, 并彼此独立地赋予权限。

MySQL 中的一个用户名就是你连接服务器时指定的用户名, 该名字不必与你的系统登录名或 Windows 名联系起来。缺省地, 如果不明确指定一个名字, 客户程序将使用登录名作为 MySQL 用户名。这只是一个约定。你可以在授权表中将该名字改为 nobody, 然后以 nobody 连接执行需要超级用户权限的操作。

## 5) password

赋予用户的口令, 它是可选的。如果对新用户没有指定 IDENTIFIED BY 子句, 该用户不赋给口令(不安全)。对现有用户, 任何你指定的口令将代替老口令。如果你不指定口令, 老口令保持不变。当你用 IDENTIFIED BY 时, 口令字符串用改用口令的字面含义, GRANT 将为你编码口令, 不要像 SET PASSWORD 那样使用 password() 函数。

WITH GRANT OPTION 子句是可选的。如果你包含它, 用户可以授予权限通过 GRANT 语句授权给其他用户。你可以用该子句给予其他用户授权的能力。

用户名、口令、数据库和表名在授权表记录中是大小写敏感的, 主机名和列名不是。

一般地, 你可以通过询问几个简单的问题来识别 GRANT 语句的种类:

- 谁能连接, 从那儿连接?
- 用户应该有什么级别的权限, 他们适用于什么?
- 用户应该允许管理权限吗?

下面就讨论一些例子。

- 谁能连接, 从那儿连接?

你可以允许一个用户从特定的或一系列主机连接。有一个极端, 如果你知道将只从一个主机连接, 可以将权限局限于单个主机。

```
GRANT ALL ON samp_db. * TO test@localhost IDENTIFIED BY“ruby”
GRANT ALL ON samp_db. * TO fred@res.mars.com IDENTIFIED BY“quartz”
```

(samp\_db. \* 意思是 samp\_db 数据库的所有表)。你也可以允许他从无论哪里连接:

```
GRANT ALL ON samp_db. * TO max@% IDENTIFIED BY“diamond”
```

“%”字符起通配符作用, 与 LIKE 模式匹配的含义相同。在上述语句中, 它意味着“任何主机”。所以 max 和 max@% 等价。这是建立用户最简单的方法, 但也是最不安全的。可以允许一个用户从另一个受限的主机集合访问。例如, 要允许 test1 从 test.net 域的任何主机连接, 用一个%.test.net 主机指定符:

```
GRANT ALL ON samp_db. * TO test1@.test.net IDENTIFIED BY“quartz”;
```

如果你喜欢, 用户标识符的主机部分可以用 IP 地址而不是一个主机名来给定。你可以指定一个 IP 地址或一个包含模式字符的地址, 而且, 从 MySQL 3.23, 你还可以指定具有指出用于网络号的位数的网络掩码的 IP 号:

```
GRANT ALL ON samp_db. * TO test@ 192. 168. 128. 3 IDENTIFIED BY "ruby"//
GRANT ALL ON samp_db. * TO test2@ 192. 168. 128. % IDENTIFIED BY "quartz"//
GRANT ALL ON samp_db. * TO test3@ 192. 168. 128. 0/17 IDENTIFIED BY "ruby"
```

第一个例子指出用户能从其连接的特定主机,第二个指定对于 C 类子网 192. 168. 128 的 IP 模式,而第三条语句中,192. 168. 128. 0/17 指定一个 17 位网络号并匹配具有 192. 168. 128 头 17 位的 IP 地址。

如果 MySQL 抱怨你指定的用户值,你可能需要使用引号(只将用户名和主机名部分分开加引号)。

```
GRANT ALL ON samp_db. president TO "test"@ "test. net"
```

● 用户应该有什么级别的权限和它们应该适用于什么?

你可以授权不同级别的权限,全局权限是最强大的,因为它们适用于任何数据库。要使 test 成为可做任何事情的超级用户,包括能授权给其他用户,发出下列语句:

```
GRANT ALL ON * . * TO test@ localhost IDENTIFIED BY "coffee" WITH GRANT OP-
TION
```

ON 子句中的 \* . \* 意味着“所有数据库、所有表”。从安全考虑,我们指定 test 只能从本地连接。限制一个超级用户可以连接的主机通常是明智的,因为它限制了试图破解口令的主机。

有些权限(FILE、PROCESS、RELOAD 和 SHUTDOWN)是管理权限并且只能用“ON \* . \*”全局权限指定符授权。如果你愿意,你可以授权这些权限,而不授权数据库权限。例如,下列语句设置一个 flush 用户,他只能发出 flush 语句。这可能在你需要执行诸如清空日志等的管理脚本中会有用:

```
GRANT RELOAD ON * . * TO flushl@ localhost IDENTIFIED BY "flushpass"
```

数据库级权限适用于一个特定数据库中的所有表,它们可通过使用 ON db\_name. \* 子句授予:

```
GRANT ALL ON samp_db TO galen@ abc. lamp. net IDENTIFIED BY "rock"//GRANT
SELECT ON samp_db TO ro_user@ % IDENTIFIED BY "rock"
```

第一条语句向 galen 授权 samp\_db 数据库中所有表的权限。第二条创建一个严格限制访问的用户 ro\_user(只读用户),只能访问 samp\_db 数据库中的所有表,但只有读取权限,即用户只能发出 SELECT 语句。

你可以列出一系列同时授予的各个权限。例如,如果你想让用户能读取并能修改现有数据库的内容,但不能创建新表或删除表,如下授予这些权限:

```
GRANT SELECT, INSERT, DELETE, UPDATE ON samp_db TO galen@ lamp. net INDE-
TIFIED BY "rock"
```

对于更精确的访问控制,你可以授权在各个表上,或在表的每个列上。当你想给用户隐藏一个表的部分时,或你想让一个用户只能修改特定的列时,列的特定权限非常有用。如:

```
GRANT SELECT ON samp_db. member TO galen@localhost IDENTIFIED BY "rock"  
GRANT UPDATE (expiration) ON samp_db. member TO galen@localhost
```

第一条语句授予 galen 对整个 member 表的读权限并设置了一个口令。第二条语句增加了 UPDATE 权限,且只对 expiration 列。没必要再指定口令,因为第一条语句已经指定了。

如果你想对多个列授予权限,指定一个用逗号分开的列表。例如,对 assistant 用户增加 member 表的地址字段的 UPDATE 权限,使用如下语句,新权限将加到用户已有的权限中:

```
GRANT UPDATE (street,city,state,zip) ON samp_db TO assistant@localhost
```

通常,你不想授予任何比用户确实需要的权限宽的权限。然而,当你想让用户能创建一个临时表以保存中间结果,但又不想让他们在一个他们不应修改内容的数据库中这样做时,发生了要授予在一个数据库上的相对宽松的权限。可以通过建立一个分开的数据库(如 tmp)并授予此数据库上的所有权限来进行。例如,如果想让来自 test.net 域中主机的任何用户使用 tmp 数据库,你可以发出这样的 GRANT 语句:

```
GRANT ALL ON tmp. * TO ""@test.net
```

在你做完之后,用户可以创建并用 tmp.tbl\_name 形式引用 tmp 中的表(在用户指定符中的""创建一个匿名用户,任何用户均匹配空白用户名)。

#### ● 用户应该被允许管理权限吗?

你可以允许一个数据库的拥有者通过授予数据库上的所有拥有者权限来控制数据库的访问。在授权时,指定 WITH GRANT OPTION。例如:如果你想让 alicia 能从 big.corp.com 域的任何主机连接并具有 sales 数据库中所有表的管理员权限,你可以用如下 GRANT 语句:

```
GRANT ALL ON sales. * TO alicia@%.big.corp.com IDENTIFIED BY "applejuice"  
WITH GRANT OPTION
```

在效果上 WITH GRANT OPTION 子句允许你把访问授权的权利授予另一个用户。要注意,拥有 GRANT 权限的两个用户可以彼此授权。如果你只给予了第一个用户 SELECT 权限,而另一个用户有 GRANT 加上 SELECT 权限,那么第二个用户就比第一个用户更强大。

## 6) 撤权并删除用户

要取消一个用户的权限,使用 REVOKE 语句。REVOKE 的语法非常类似于 GRANT 语句,除了用 FROM 取代 TO 并且没有 IDENTIFIED BY 和 WITH GRANT OPTION 子句:

```
REVOKE privileges (columns) ON what FROM user
```

user 部分必须匹配原来 GRANT 语句中的想撤权的用户的 user 部分。privileges 部分不需匹配,你可以用 GRANT 语句授权,然后用 REVOKE 语句只撤销部分权限。

REVOKE 语句只删除权限,而不删除用户。即使撤销了所有权限,在 user 表中的用户



记录依然保留,这意味着用户仍然可以连接服务器。要完全删除一个用户,必须用一条 DELETE 语句明确从 user 表中删除用户记录:

```
mysql > DELETE FROM user WHERE user = "user_name" and host = "host name";  
mysql > FLUSH PRIVILEGES;
```

DELETE 语句删除用户记录,而 FLUSH 语句告诉服务器重载授权表。(当你使用 GRANT 和 REVOKE 语句时,表自动重载,而直接修改授权表时则不是)

## 4. 恢复数据

### 1) 备份

一个好的备份策略对于管理员来说非常重要。如果系统曾经崩溃过,希望能在数据损失尽量少的情况下恢复数据,那么做好备份将是很有益的。同样,如果偶然删除或者损坏了数据库,可以随时提供备份。

有些重要问题在备份的时候需要考虑:

- 可能的话将备份存储在一个设备中而不是数据库中。如果磁盘崩溃了,那么你将会很庆幸在别处还有一个备份。如果有二进制日志,则应存储到备份所在的设备上。
- 确保有足够的磁盘空间来完成备份。
- 除备份以外,还要使用二进制日志,这样可以在最小的数据损失下恢复数据库。如果选择不是用二进制日志,你将能把数据库恢复到最后备份的状态。相应于具体的应用,有时没有二进制备份可能是无用的。
- 保存一定数量的归档备份。
- 在紧急情况发生前测试备份。

下面介绍两个备份的 MySQL 的实用程序:

#### ◆ mysqldump

要用 mysqldump 备份数据库,建议你使用 -opt 选项。这个选项将提供最快速的数据库转储。

命令如下:

```
[root@lupa bin]# mysqldump -opt test > /usr/backups/testdb
```

这个命令将测试数据库转储到文件 /usr/backups/testdb 中。如果正在使用二进制日志,那么还要指定 -flush-logs,这样二进制日志将得到一个备份检查点:

```
[root@lupa mysql]# mysqldump-flush-logs -opt test > /usr/backups/testdb
```

#### ◆ mysqlhotcopy

mysqlhotcopy 是一个 perl 脚本,使用 LOCK TABLES、FLUSH TABLES 和 UNIX cp 的组合来进行数据库快速备份。

命令如下:

```
[root@ lupa mysql]# mysqlhotcopy test/usr/backups
```

这个命令在/usr/backups 目录下创建一个新的目录,将测试数据库的所有数据文件复制到其中。

## 2) 恢复

一般情况下需要两样东西来执行数据库恢复:备份文件和二进制日志。

### ● mysqldump 恢复

下面的命令是重新加载数据库:

```
[root@ lupa mysql]# cat test.dump | mysql
```

这条命令将运行由 mysqldump 生成的 SQL 命令,将数据库带回到最后一次备份时的状态。

如果你只想恢复一个数据库,那么使用-one-database mysql 选项来过滤用于其他数据库的 SQL 命令。

命令如下:

```
[root@ lupa mysql]# mysqlbinlog host-bin.XXX | mysql-one-database = testdb
```

### ● mysqlhotcopy 恢复

从 mysqlhotcopy 备份恢复,可以通过在服务器不运行时将数据库文件从备份位置复制到 mysql 数据位置来重新装载数据库。假设数据库备份下/var/backup/test 下,而 mysql 数据位置在/usr/local/mysql/data 下,则执行:

```
[root@ lupa mysql]# cp -R /var/backup/test/usr/local/mysql/data
```

这个命令将数据库带回最后备份的状态。

## 5. phpMyAdmin

phpMyAdmin 是一套以基于 php 开发,针对 MySQL 数据库系统的 Web 界面管理系统。它可以很方便地以图形化界面,来对 MySQL 数据库的数据表做增删的动作,也可以做数据库本身的增删管理;另外一个好处是,你也可以通过这个界面来学习 SQL 正确的语法。

我们可以通过站点 <http://www.phpmyadmin.net> 和 <http://tech.lupaworld.com/lamp> 下载最新版本的 phpMyAdmin。我们选择 2.8 系列,这里要注意的是 2.8 版本与早期版本在安装方面有所有不同。安装 phpMyAdmin 有多种方法,其一是直接改配置文件内容,其二是通过安装文件生成脚本文件。本书介绍是第一种方法,主要是因为简单快速。

假设这里的版本是 phpMyAdmin2.8.2.4.tar.gz,对软件解压:

```
[root@ lupa root]# tar zxvf phpMyAdmin2.8.2.4.tar.gz
```

解压后得到目录 phpMyAdmin2.8.2.4

```
[root@ lupa root]# mv phpMyAdmin2.8.2.4/usr/loca/apache/htdocs/ //移动文件
[root@ lupa root]# mv phpMyAdmin2.8.2.4 pma //重命名为 pma
[root@ lupa root]# cd /usr/loca/apache/htdocs/pma //进入 pma 目录
[root@ lupa pma]# cp libraries/config.default.php config.inc.php //复制文件并更
命。
[root@ lupa pma]# vi config.inc.php //配置文件 config.inc.php
```

在配置文件中,找到以下条目:(其他都默认)

```
$ cfg[ 'blowfish_secret' ] = ' ' //主要作用是加密你的 cookie,可以任意设置
$ cfg[ 'Servers' ][ $ i ][ 'controluser' ] = 'root'; //MySQL 控制用户设置
$ cfg[ 'Servers' ][ $ i ][ 'controlpass' ] = '';
$ cfg[ 'Servers' ][ $ i ][ 'auth_type' ] = 'cookie'; //验证方式
$ cfg[ 'Servers' ][ $ i ][ 'user' ] = 'root'; //MySQL 用户
$ cfg[ 'Servers' ][ $ i ][ 'password' ] = '';
```

现在配置文件都设置好了,打开浏览器,输入: <http://localhost/pma/>, 如图:

phpMyAdmin 可以运行起来了,但默认设置是不安全的,你需要给 phpMyAdmin 所在的目录加上访问认证,可以参考本书第二章 apache 使用和安装中访问认证控制这节来实现。

# PHP 安装和使用

本章我们将介绍 LAMP 的最后一个部分 PHP 的安装和使用。PHP 的配置在 LAMP 中起决定作用,它是建立 Apache 与 MySQL 之间的桥梁。

## 第 1 节 为什么要使用 PHP

PHP 是一种用于创建动态 Web 页面的服务端脚本语言。如同 ASP 和 JSP,用户可以混合使用 PHP 和 HTML 编写 Web 页面,当访问者浏览到该页面时,服务端会首先对页面中的 PHP 命令进行处理,然后把处理后的结果连同 HTML 内容一起传送到访问端的浏览器。但是与 ASP 或 JSP 不同,PHP 是一种源代码开放程序,拥有很好的跨平台兼容性。用户可以在 Windows NT 系统以及许多版本的 Unix 系统上运行 PHP,而且可以将 PHP 作为 Apache 服务器的内置模块或 CGI 程序运行。

除了能够精确地控制 Web 页面的显示内容之外,用户还可以通过使用 PHP 发送 HTTP 报头。可以通过 PHP 设置 cookies,管理用户身份识别,并对用户浏览页面进行重定向。PHP 具有非常强大的数据库支持功能,能够访问几乎目前所有较为流行的数据库系统。此外,PHP 可以与多个外接库集成,为用户提供更多的实用功能,如生成 PDF 文件等。

你可以直接在 Web 页面中输入 PHP 命令代码,因而不需要任何特殊的开发环境。在 Web 页面中,所有 PHP 代码都被放置在“<? php”和“? >”中。此外,用户还可以选择使用诸如 <SCRIPT LANGUAGE = “php” > </SCRIPT> 等的形式。PHP 引擎会自动识别并处理页面中所有位于 PHP 定界符之间的代码。

PHP 脚本语言的语法结构与 C 语言和 Perl 语言的语法风格非常相似。用户在使用变量前不需要对变量进行声明。使用 PHP 创建数组的过程也非常简单。PHP 还具有基本的面向对象组件功能,可以极大的方便用户有效组织和封装自己编写的代码。

## 第 2 节 PHP 版本

PHP 当前最流行的两大版本是 PHP 4 和 PHP 5。本书选用 PHP 5.1.2 为例子。它除了具备 PHP 过去版本的优点外,还融合了 Zend Engine 2.0 技术。引进了私有和保护成员变量,它们可以定义可视化的类属性,PHP 5 还引入了抽象类和方法,数据抽象层(PDO),等等。

PHP 5 相对于过去的版本,对 XML 的支持性更好、更灵活,并且对 java 的兼容性和强大的面向对象功能更加显著。

## 第3节 PHP 开发工具

PHP 开发工具很多,开发人员可以借助开发工具,写出更优秀的 PHP 程序。本书仅向读者介绍 Zend 开发工具。

Zend Studio 是专业开发人员在使用 PHP 整个开发周期中的集成开发环境(IDE),它包括了 PHP 所有必需的开发部件。通过一整套编辑、调试、分析、优化和数据库工具,Zend Studio 加速开发周期,并简化复杂的应用方案。Zend Studio 设计时考虑了商务应用开发的需要,是开发者利用 PHP 创造强有力数据库应用软件的完美开发环境。Zend Studio 具有专业和企业开发者开发、配置、调试和管理关键商务 PHP 应用所需的易使用性、可伸缩性、可靠性和扩展性。

## 第4节 安装 PHP

PHP 是本书最后一个安装项目,LAMP 是否能工作关键是取决于 PHP 的安装和配置,在本节将学到如何在 PHP 安装过程中,根据需要添加定制 PHP 功能。

### 1. 确定配置选项和扩展名

#### 1) 基本配置选项

```
--enable-debug
```

编译时加入调试符号。

```
--with-layout = TYPE
```

设置被安装文件的布局,TYPE 是 PHP(默认)或 GNU。

```
--with-pear = DIR
```

在 DIR(默认为 PREFIX/lib/php)中安装 PEAR。

```
--without-pear
```

不安装 PEAR。

```
--enable-sigchild
```

使用 PHP 自带的 SIGCHLD 处理器。

`--disable-rpath`

禁用在搜索路径中传递其他运行库。

`--enable-libgcc`

启用 libgcc 的精确链接。

`--enable-php-streams`

包含试验性的 PHP 流。不要使用这个选项,除非要测试代码。

`--with-zlib-dir[ = DIR]`

定义 zlib 的安装目录。

`--with-tsrml-pthreads`

使用 POSIX 线程(默认)。

`--enable-shared[ = PKGS]`

编译共享库[默认 = 是]。

`--enable-static[ = PKGS]`

编译静态库[默认 = 是]。

`--enable-fast-install[ = PKGS]`

为快速安装优化[默认 = 是]。

`--with-gnu-ld`

假设 C 编译器使用 GNU ld[默认 = 否]。

`--disable-libtool-lock`

阻止锁(可能破坏平行编译)。

`--with-pic`

尝试仅使用 PIC/非 PIC 对象[默认 = 都使用]。

`--enable-memory-limit`

编译内存限制支持功能。

`--disable-url-fopen-wrapper`

禁用 URL 形式的 fopen 包裹器,该包裹器允许通过 HTTP 或者 FTP 访问文件。

`--enable-versioning`

仅导出必需的符号,查看 INSTALL 文件以获得更多信息。

## 2) PHP 选项

`--enable-maintainer-mode`

启用后使得规则和依赖关系(和有时的混乱)临时对安装程序不再有用。

`--with-config-file-path = PATH`

设置 php.ini 的搜索路径,默认为 PREFIX/lib。

`--enable-safe-mode`

默认启用安全模式。

`--with-exec-dir[ = DIR ]`

在安全模式时仅允许在 DIR 目录中执行。默认目录为 /usr/local/php/bin。

`--enable-magic-quotes`

默认启用 magic quotes。

`--disable-short-tags`

默认仅用短形式的开始标签 <?。

## 3) SAPI 选项

下面列出 PHP 可用的 SAPI(服务器应用编程接口)。

`--with-aolserver = DIR`

指定 AOLserver 的安装路径。

`--with-apxs[ = FILE ]`

编译共享的 Apache 模块。FILE 是可选的 Apache apxs 工具的路径,默认指向 apxs。请确认指定的 apxs 已经安装在服务器中,并且它不是 Apache 源码包中的那个 apxs。

`--with-apache[ = DIR ]`

编译静态 Apache 模块。DIR 是 Apache 编译目录的顶层,默认为 /usr/local/apache。

`--with-mod_charset`

启用 mod\_charset 的转换表。



```
--with-apxs2[ = FILE]
```

编译共享的 Apache 2.0 模块。FILE 是可选的 Apache apxs 工具的路径,默认指向 apxs。

```
--with-caudium = DIR
```

为使用 Caudium 编译 PHP 为一个 Pike 模块。DIR 是 Caudium 服务器目录,默认为 /usr/local/caudium/server。

```
--disable-cli
```

禁止编译 PHP 的命令行版本(CLI)(使用它将同时强制使用--without-pear 选项)。

```
--enable-embed[ = TYPE]
```

启用潜入的 SAPI 编译。TYPE 或者为 shared 或者为 static,默认为 shared。

```
--with-isapi = DIR
```

为 Zeus 服务器以 ISAPI 模块方式编译 PHP。

```
--with-nsapi = DIR
```

指定 Netscape/iPlanet/SunONE 的安装目录。

```
--with-pi3web = DIR
```

为 Pi3Web 服务器编译 PHP 模块。

```
--with-roxen = DIR
```

以 Pike 模块方式编译 PHP,DIR 是 Roxen 的根目录,默认为 /usr/local/roxen/server。

```
--enable-roxen-zts
```

使用 Zend 线程安全编译 Roxen 模块。

```
--with-servlet[ = DIR]
```

包含 servlet 支持。DIR 是 JSDK 的安装目录。此 SAPI 要求 java 扩展必须作为共享模块编译到 PHP 中。

```
--with-thttpd = SRCDIR
```

编译 PHP 为 thttpd 模块。

```
--with-tux = MODULEDIR
```

编译 PHP 为 TUX 模块(仅在 Linux 下有效)。

```
--with-webjames = SRCDIR
```

编译 PHP 为 WebJames 模块(仅在 RISC 的操作系统中有效)。

```
--disable-cgi
```

禁止编译 CGI 版本的 PHP。

```
--enable-force-cgi-redirect
```

启用内部服务器重定向的安全检测。如果您在 Apache 下使用 CGI 版本的 PHP,请启用该选项。

```
--enable-discard-path
```

如果启用该选项,PHPCGI 目录可以安全地放在 Web 目录树的外面,人们无法避开.htaccess 的安全限制。

```
--enable-fastcgi
```

如果启用,CGI 模块将被编译为支持 FastCGI。

```
--disable-path-info-check
```

如果该选项被禁用,路径将不能工作,例如/info.php/test? a = b

## 2. 下载和准备工作

获得最新的 PHP 系统(本书采用 PHP 5.1.2,你可以根据自己的偏好,下载不同的版本),官方下载地址:<http://us3.php.net/downloads.php>。我们也推荐速度较快的 LUPA 社区,下载地址:<http://tech.lupaworld.com/lamp/>下载并解压缩,假如我们下载到/root 目录下:

```
[root@ lupa/]# cd root //进入 root 目录;
```

```
[root@ lupa ~]# tar-zxvf php-5.1.2.tar.gz //解压安装包;
```

解压后会生成 PHP 5.1.2 目录。接下来建立安装目录,这里我们把 PHP 安装到/usr/local/。

```
[root@ lupa/]# mkdir/usr/local/php //建立安装目录;
```

## 3. 编译安装 PHP

进入解开的文件目录,按顺序执行下列操作:

```
[root@ lupa ~]# cd php5.1.2
[root@ lupa php5.1.2]# ./configure--prefix=/usr/local/php\ //指定安装目录;
--with-apxs2=/usr/local/Apache/bin/apxs\ //加入 apxs 支持;
--with-mysql=/usr/local/mysql\ //支持 MySQL;
```

以上配置选项仅仅实现了共享的 Apache 模块,MySQL 数据库支持,更多的功能可以通过添加参数来实现。

接下来编译安装,执行:

```
[root@ lupa php5.1.2]# make
[root@ lupa php5.1.2]# make install
```

最后拷贝 php.ini 配置文件。这里的 php.ini-dist 文件在解压后的文件包中。

```
[root@ lupa php5.1.2]# cp php.ini-dist/usr/local/php/etc/php.ini
```

## 4. 配置 Apache 和 PHP

编辑 httpd.conf 文件:

```
[root@ lupa/]# vi/usr/local/Apache/conf/httpd.conf
```

在配置单里增加

```
AddType application/x-httpd-php.php.php3 //添加 Apache 支持和解释的 php 后缀名;
AddType application/x-httpd-php-source.phps //同上;
```

并找到下面一行

```
DirectoryIndex index.html index.html.var
```

改为

```
DirectoryIndex index.html index.html.var index.php //添加默认访问类型;
```

支持 PHP 格式。

当一切都配置好之后重启 Apache:

```
[root@ lupa/]# /usr/local/Apache/bin/apachectl restart
```

如果使用带有 mod\_ssl 的 Apache,需要执行:

```
[root@ lupa/]# /usr/local/Apache/bin/apachectl startssl
```


至此,PHP 安装完成。

## 5. 运行 PHP

测试运行 PHP, 首先写一个简单的 PHP 脚本检测安装是否成功。

```
<?
phpinfo();
? >
```

将其保存到 `/usr/local/Apache/htdocs/`, 命名为 `index.php` 文件。访问 `http://localhost` 如果看到如下页面, 恭喜你 PHP 安装成功。

PHP Version 5.1.2	
	
System	Linux localhost.localdomain 2.6.15-1.2054_FC5 #1 Tue Mar 14 15:48:33 EST 2006 i686
Build Date	Aug 25 2006 21:31:00
Configure Command	./configure '--prefix=/usr/local/php' '--with-apxs2=/usr/local/apache/bin/apxs' '--with-mysql=/usr/local/mysql'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/php/lib
PHP API	20041225
PHP Extension	20050922
Zend Extension	220051025
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
IPv6 Support	enabled
Registered PHP Streams	php, file, http, ftp
Registered Stream Socket Transports	tcp, udp, unix, udg
Registered Stream Filters	string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, convert.iconv.*

## 第 5 节 PHP 语法

如果你是一位程序员或者是一个懂一点 C 语言的人, 初次接触 PHP, 你一定会感到很亲切, 它的语法风格接近 C 等基本语言。要用 PHP 写个网页, 只需要一个简单的文字编辑器, 像 Linux 的 vi 工具、Windows 的记事本。而且只要有程序设计基础, 不论是 C、C++、Pascal、JAVA, 学 PHP 就会很快, 因为它和其他语言的语法或是结构差不多。

### 1. Hello world

学习一个程序语言最基本的就是显示 Hello world, 也就是做最基本的输出, 如果不会输

出,不管程序执行了什么,你也得不到任何结果。比较一下 C 语言和 PHP 语言的输出方式:

### 1) C 语言的输出

```
#include <stdio.h>
void main()
{
    printf("Hello World \n");
}
```

### 2) PHP 语言的输出

```
<? php
echo "Hello World \n";
? >
```

## 2. 常量与变量

### 1) 常量

PHP 预先定义了几个常量,并提供了一种机制在运行时自己定义。常量和变量基本上是一样的,不同的是:常量必须用 DEFINE 函数定义。常量一旦定义好,就不能被重新定义。PHP 中预先定义好的常量:

\_\_FILE\_\_

当前正在处理的脚本文件名。如果使用在一个被包含的文件中,那么它的值就是这个被包含的文件,而不是包含它的文件名。

\_\_LINE\_\_

正在处理的文件的当前行数。

PHP\_VERSION

表示 PHP 的当前版本。

PHP\_OS

PHP 所在的操作系统名字。

TRUE      FALSE

真值      假值

当然你可以用 DEFINE 函数定义更多的常量。举例,定义常量 LUPA:

```
<? php
define("LUPA", "Hello World. ");
echo LUPA; //输出"Hello World."
? >
```

在 PHP 里面有些预设好的常量,像是基本的 True 以及 False 各代表 1 和 0,另外 PHP\_OS 可以显示出服务器的系统。要注意的是常量不能在任何引号中使用。

```
<? php
define("LUPA", "Open Source");
echo "LUPA";
//显示 LUPA
echo "I Love". LUPA;
//显示 I Love Open Source
? >
```

## 2) 变量

PHP 的变量是由符号( \$ )加上变量名称组成。变量名称第一个字不得为数字,而且变量名称中也不得包含符号。

正确的变数: \$ a、\$ aa、\$ a1、\$ a\_2、\$ \_aaa

不正确的变数: \$ 1a、\$ aa/s、\$ a \$ a、\$ 1\%

变量使用范例:

```
<? php
$ a1 = "test1";
echo " $ a1";
$ a2 = "test2";
echo " $ a2";
? >
```

在使用 echo 时,变量内容可以用""或是' '或是不使用引号,但为了避免弄混,初学者尽量养成使用""的习惯。

- "":引号中间的数据可以包含字符串、数字或是变量。
- ' ':引号中间只能包含字符串及数字。
- 不使用引号:只能纯字符串、数字或是纯变量。

```
<? php
$ name = "Open Source";
//例 1:
echo "I Love $ name";
//显示 I Love Open Source
//例 2:
echo 'I Love $ name';
//显示 I Love $ name
//例 3(混合使用):
echo "I Love". $ name; //显示 I Love Open Source
? >
```



### 关于全局变量与局部变量

学过 C/JAVA/C++ 之类的朋友一定了解 `main()` 内声明的变量所达的区域。PHP 为解释性语言而不是编译性语言,我们也知道 PHP 不存在 `main()` 主体,怎么声明呢? 其实 PHP 页面本身就是一个 `main()`,只要是在页面内而非在函数内声明的变量我们都称其为全局变量。例如:

```
<? php
    $ int_a = 0; //初始全局变量 $ int_a 将其赋值为 0;
    echo "全局变量:". $ int_a; //打印全局变量;
    function child_a() //局部变量的使用;
    {
        $ int_a = 0; //这里是局部变量,虽然与全局变量名相同但此变量仅在 child_a
    内可见;
        echo "From child_a:". $ int_a;
    }
    function child_b() { //调用全局变量;
        global $ int_a; //声明调用全局变量;
        echo "From child_b:". $ int_a;
    }
? >
```

从上面的一个例子,我们可以很简单明了的看明白 PHP 中全局与局部变量之间的差距。

## 3. 运算符

运算符为程序中用于执行计算动作的符号。PHP 的运算符跟 C 很类似,大致分为以下几类:

- ◆ 算术运算符
- ◆ 比较运算符
- ◆ 逻辑运算符
- ◆ 字符串运算符

各类运算符的说明请参考接下来各小节的说明,主要介绍常用运算符。

### 1) 算术运算符

算术运算符主要由加、减、乘、除及余数组成,如下表:

运算符符号	名称	使用语法范例	当 A = 2, B = 1 时的计算结果
+	加	A + B	3
-	减	A - B	1
*	乘	A * B	2
/	除	A / B	2
%	余数	A % B	0

举例说明:

```
<? php
$ x = 10 + 11;
echo $ x; //输出 21;
$ y = 13%3;
echo $ y; //输出 1;
? >
```

## 2) 比较运算符

比较运算符由等于、大于、小于、大于等于、小于等于、不等于组成,如下表:

运算符符号	名称	使用语法范例	当 A = 1, B = 2 时的计算结果
=	等于	A = B	假
>	大于	A > B	假
<	小于	A < B	真
>=	大于等于	A >= B	真
<=	小于等于	A <= B	假
!=, <>	不等于	A <> B, A != B	真

举例说明:

```
<? php
$ A = 3;
if ( $ A == 3 ) echo "真的";
if ( $ A >= 5 ) echo "假的";
if ( $ A != 3 ) echo "假的";
? >
```

## 3) 逻辑运算符

逻辑运算符由 AND、OR、XOR、! 等组成,如下表:

运算符符号	名称	使用语法范例	说明
&&, AND	且	A && B, A AND B	当 A、B 均为真时,结果为真
, OR	或	A    B, A OR B	当 A、B 其中之一为真时为真
XOR	互斥	A XOR B	A 与 B 相同时为假,否则为真
!	否	! A, ! F	T 为 F, F 为 T,即颠倒原本意义

下表称为真值表,T 代表真,F 代表假,我们整理当 A、B 在各种情形下,各逻辑运算符的运算结果:

A	B	A ADD B	A OR B	A XOR B	! A
T	T	T	T	F	F
F	T	F	T	T	T
F	F	F	F	F	T
T	F	F	T	T	F

举例说明：

```
<? php
$ x = 10;
$ y = 15;
if ( $ x == 10 && $ y > 0 ) echo "真";
if ( $ x == 10 && $ y < 0 ) echo "假";
if ( $ x >= 10 || $ y == -15 ) echo "真";
if ( $ x == 10 || $ y == 15 ) echo "真";
if ( $ x != 10 || $ y == -15 ) echo "假";
if ( $ x == 10 xor $ y > 0 ) echo "假";
if ( $ x == 10 xor $ y > 16 ) echo "真";
$ z = 9;
if ( $ x == 10 xor $ y > 0 xor $ z == 8 ) echo "假";
if ( $ x == 10 xor $ y > 0 xor $ z == 9 ) echo "真";
? >
```

#### 4) 字符串运算符：

“.”是连接变量与变量或是变量与字符串。举例说明,如下：

```
<? php
$ A = "LUPA";
$ B = $ A . $ A;
echo $ B; //输出 LUPALUPA
? >
```

再举个例子：

```
<? php
$ A = "Hello";
$ num = 3 + 2;
$ num ++;
echo " $ A $ num people!";
? > //输出 Hello 6 people!
```

## 4. 流程控制

流程控制是程序中用于控制或选择某一程序区段执行方式的语法。有了流程控制以后,可以让原先一行一行的程序产生许多变化,能够重复或控制某一段程序是否被执行。在 PHP 中,主要有以下几种流程控制方式:

### 1) if 判断语句

if 是最常用的判断式,只针对单一条件做判断。要是不符合指定的条件,就不执行指定的动作,然后继续执行这个条件式之后的程序。

```
if (2 > 1) {  
    动作 1  
    动作 2  
    ...  
}
```

如果执行的动作只有一个,可以省略 `}`。比如:

```
if (3 > 1) 动作 1
```

### 2) if else 语句

if else 可以针对 if 描述的条件,执行指定的动作,若不符合条件,则执行 else 的动作。

```
if (某条件) {  
    动作 1  
    动作 2  
    ...  
} else {  
    动作 3  
    动作 4  
    ...  
}
```

可以应用在连接数据库时告知使用者是不是联机成功,if 联机成功就进行查询的动作,else 送出联机失败的消息。

### 3) if else if 语句

if else if 是针对多种状况做判断,再决定执行哪一种动作,只要执行到符合的条件就执行,然后忽略之后的其他状况,结束这个判断。

```
if (条件 1) {  
    动作 1  
    动作 2  
    ...  
} else if (条件 2) {  
    动作 3  
    动作 4  
    ...  
} else {  
    动作 5  
    动作 6  
    ...  
}
```

#### 4) switch

switch 也有转换的意思,对指定的变量一个个的比对,switch 跟 if 条件语句不同的地方在于就算找到了符合条件的也会继续对比下去,一旦遇到指定变量,除非对比完毕或是遇到 break 指令,不然会把之后所有的动作执行完。

```
<? php  
$ i = 3;  
switch ( $ i ) {  
case 1:  
    echo "i = 1";  
    break;  
case 2:  
    echo "i = 2";  
    break;  
case 3:  
    echo "i = 3";  
    break;  
case 4:  
    echo "i = 4";  
    break;  
} //结果会输出 i = 3;  
? > //要是没有 break;  
<? php  
$ i = 2;
```



```
switch ( $ i) {  
    case 1:  
        echo "i = 1";  
    case 2:  
        echo "i = 2";  
    case 3:  
        echo "i = 3";  
    case 4:  
        echo "i = 4";  
} //结果会输出 i = 2 i = 3 i = 4;  
? >
```

### 5) for 循环

php 使用 for 循环的方式与其他语言相同,for 里面包含三个部分,起始值、变量范围、变量递增、递减。

```
<? php  
for ( $ i = 0; $ i < = 10; $ i + + ) {  
    echo " $ i";  
} //输出 012345678910;  
? >
```

### 6) while 循环

php 使用 while 循环的方式与其他语言相同,在()内条件成立时,会继续执行{}内的程序代码,适合用来处理数量不确定的数据。

```
<? php  
$ i = 8;  
while ( $ i > 0 ) {  
    echo " $ i";  
    $ i --;  
} //输出 87654321;  
? >
```

要是不确定资料数量,就可以使用下面的范例做输出,不论 \$ data 这个数组多大都适用。

```
<? php  
$ data = array( "A", "B", "C" );  
$ i = 0;
```



```
while ( $ data[ $ i]) {  
    echo $ data[ $ i + + ];  
} //输出 ABC;  
? >
```

## 5. 数组

PHP 的数组使用相比较 C 语言简单得多,使用前不需要定义,所以也不需要定义数组的大小。要定义一个数组,使用 `array()` 函式:

定义数组:

```
<? php  
$ data = array();  
? >
```

但通常是不需要这个步骤的,这个定义的动作只会让程序稍微慢一点,除非要替这个数组一次加入多个元素,否则这个函式可以省略。

在数组加入元素:

```
<? php  
$ data[ ] = 1;  
$ data[ ] = 2;  
$ data[ ] = 3;  
echo $ data[0];  
echo $ data[1];  
echo $ data[2]; //结果输出 123;  
? >
```

也可以这样,让数组中的元素在开始就被定义好:

```
<? php  
$ data = array("1","2","3");  
echo $ data[0];  
echo $ data[1];  
echo $ data[2]; //结果输出 123;  
? >
```

# 优化测试 LAMP

现在我们已经完成了 Linux、Apache、MySQL 和 PHP 的安装。接下来的任务就是要让 LAMP 稳定地运作起来。本章先从创建虚拟主机和使用 MySQL 数据库开始,进行对 LAMP 平台的基础测试。然后重点分析 PHP 缓存器的安装和配置,以便对 LAMP 的性能进一步优化。

## 第1节 安装 Apache 虚拟主机

为展开 LAMP 的优化测试,我们需要确保 Apache 正常运行。为确保测试环境的相对独立性,我们将建立一个虚拟主机来进行测试。创建虚拟主机只需要对 Apache 的配置文件 httpd.conf 进行修改即可。

先创建一个基于域名的虚拟主机,例如 `http://www.lamp.com`。它将直接指向到 `/usr/local/apache/htdocs/lamp/` 目录下。根据以下步骤进行操作:

- 具体的配置细节我们在第三章已经有描述。我们只需要在配置文件中添加如下内容:

```
<VirtualHost *:80>
    DirectoryIndex index.html index.htm
    ServerAdmin webmaster@lamp.com
    DocumentRoot /usr/local/apache/htdocs/lamp/ //定义主目录
    ServerName www.lamp.com //设置域名
    ErrorLog logs/www.lamp.com-error_log
    CustomLog logs/www.lamp.com-access_log combined
</VirtualHost>
```

并确保 `NameVirtualHost *:80` 没有被注释。

- 重新启用 Apache 服务器。
- 为确保本测试顺利进行,我们手动设定 lamp 域名解析到本地。

```
[root@lupa/]# echo "127.0.0.1 www.lamp.com" > /etc/hosts
```

- 然后创建一个 `helloworld.htm`。

```
[root@ lupa/]# echo "helloworld" > /usr/local/apache/htdocs/lamp/index.html
[root@ lupa/]# chmod 777 /usr/local/apache/htdocs/lamp/index.html
```

- 打开浏览器输入 <http://www.lamp.com>, 如果能显示出 helloworld 那恭喜你, 说明虚拟主机创建成功。

## 第 2 节 准备 MySQL 数据库和用户

创建一个 MySQL 数据库 lamp\_database 和一个单独的具备此数据库管理权限的用户, 用户名和密码分别为: lamp\_user、lamp\_passwd。首先登陆 MySQL 数据库。

```
[root@ lupa/]# mysql -u "root" -p //以 root 用户进入 Mysql 数据库
Enter password: * * * * //系统提示要求输入密码, 你必须知道与用户名对应的密码
mysql > create database lamp_database;
mysql > GRANT ALL ON lamp_database. * TO lamp_user@ localhost IDENTIFIED BY 'lamp_passwd';
```

至此, 数据库和用户准备完毕。

## 第 3 节 测试 Apache, PHP 和 MySQL

### 1. 数据库连接

现在编写 PHP 程序, 我们直接采用如下脚本供大家使用:

```
/*
Connect.php
数据库链接脚本
*/
<? php
$ hostname_conn = "localhost"; //指定主机名, 可以是 IP
$ database_conn = "lamp_database"; //要连接的数据库名称
$ username_conn = "lamp_user"; //我们创建的用户
$ password_conn = "lamp_passwd"; //用户密码
$ conn = mysql_pconnect( $ hostname_conn, $ username_conn, $ password_conn) or
die( mysql_error() );
```

```
echo“连接成功”;  
? >
```

将 connect.php 拷贝到 /usr/local/apache/htdocs/lamp/connect.php。

用浏览器访问: <http://www.lamp.com/connect.php> 如出现“连接成功”字样,则说明 LAMP 正常工作了。

## 2. 数据插入

要测试数据库的插入,我们首先需要在 lamp\_database 中建立数据表,如下所示:

```
mysql > use lamp_database;  
mysql > create table test (  
mysql > id tinyint(4) NOT NULL auto_increment,  
mysql > name varchar(10) NOT NULL,  
mysql > KEY ID(id)  
mysql > );
```

现在我们创建了一个包含有 id 和 name 字段的数据表 test。其中 id 是自动增长。然后我们创建以下 PHP 脚本:

```
/*  
insert.php  
数据库插入脚本  
*/  
<? php  
include“connect.php”; //包含数据库链接脚本  
$ name = “example”;  
$ query = “insert into lamp_database (name) values (‘$ name’)”;  
$ result = @ mysql_query(“$ query”, $ conn); //执行  
if (! $ resule)  
    echo“插入数据失败”;  
else  
    echo“执行成功”;  
? >
```

同样用浏览器访问 <http://www.lamp.com/insert.php>。

## 3. 数据提取

现在我们开始将针对已经进行数据插入操作的数据表进行数据读取操作,同样我们先

编写以下脚本：

```
/*
read.php
数据库读取脚本
*/
<? php
include "connect.php"; //包含数据库链接脚本
$ name = "example";
$ query = "select id,name from lamp_database";
$ result = @ mysql_query( " $ query", $ conn); //执行
if( $ num == 0){
    echo "没有找到任何记录";
    exit();
}
while( $ row = mysql_fetch_array( $ result))
{
    echo $ row["id"]. " ". $ row["Name"]. "<br>";
}
? >
```

然后再用浏览器访问 <http://www.lamp.com/read.php>。

到目前为止,针对虚拟主机的 PHP 和 MySQL 功能测试已经完成,现在我们开始考虑系统的运行效率。目前除了可以针对 LAMP 本身进行优化配置之外,我们还可以采用各种的缓存引擎(caching engines)来提高 PHP 的执行速度。为了更好地了解这是如何工作的,我们快速查看一下 PHP 在执行脚本时所经历的过程。与任何脚本语言一样,代码在执行前都会被分析并从可供人读的格式转换为一系列可由机器识别的指令。在 PHP 中,这一过程是由 Zend Engine Parser 执行的,它将一个脚本转换为所谓的操作码。然后这些操作码被传递给执行器,它解译这些指令并执行所要求的操作。但是,大部分应用程序常常加载不同的 PHP 脚本库、类等,从而对脚本的所有外部组件及其子组件的编译和执行的过程会重复进行。

因此我们将采用共享缓存处理操作码,以尽量减少重复性的工作。接下来逐个介绍目前应用广泛的五种方案。

## 第 4 节 Alternative PHP Cache

APC(Alternative PHP Cache)的简称,是来自 Community Connect 的一个免费缓存模块。这个产品已经具有足够的稳定性供正式场合使用,而且它看起来也能在很大程度上提高响应请求的速度。APC 将操作码放入共享内存中来等待下一个访问连接,因此大大减少了对



硬盘访问的开销,从而提高了 PHP 的访问速度。

APC 可以以 DSO(动态共享组件)方式安装,也可以作为静态模块编译进 PHP。下文我们重点介绍 DSO 方式,因为这样就不需要你重新编译 PHP,而且卸载也很方便。

## 1. 安装 APC

最新版的 APC 及相应文档,教程和更新日志可以在:<http://tech.lupaworld.com/lamp/> 下载。截至 2006 年 6 月,最新稳定版本号是 3.0.10。假设该软件放置在 /root/soft/ 中,接下来我们将对 APC-3.0.10.tgz 进行操作。

- 解压到 /usr/local 目录。

```
[root@ lupa/]# cd /usr/local
[root@ lupa local]# tar xzvf /root/soft/APC-3.0.10.tgz
```

- 进入源码目录,使用 phpize 检查 PHP 安装的相关参数并创建 configure 文件。

```
[root@ lupa/]# /usr/local/php/bin/phpize
```

- 运行完成后,我们将采用 configure 指定 apxs 和 php-config 的路径并创建 MakeFile 文件。

```
[root@ lupa APC-3.0.10]# ./configure \
--enable-apc \
--enable-apc-mmap \
--with-apis = /usr/local/apache/bin/apxs \
--with-php-config = /usr/local/php/bin/php-config
```

- 然后进行编译和安装。

```
[root@ lupa APC-3.0.10]# make
[root@ lupa APC-3.0.10]# make install
```

- 此时,apc.so 将出现在 PHP 的 extensions 目录中,然后修改 php.ini 文件,增加对 apc.so 的引用。

```
/* php.ini */
.....
extension = "apc.so"
.....
```

- 重新启动 Apache 即完成 APC 的安装。



## 2. 配置 APC

在大多数情况下,默认的 APC 配置已经可以满足要求。假如需要 APC 更好地根据系统需求和硬件情况充分发挥出性能,那我们可以在 php.ini 中使用如下指令:

指令	默认	位置	描述
apc.enabled	1	php.ini httpd.conf .htaccess	是否使用 APC,1 表示使用,0 为否
apc.shm_segments	1	php.ini httpd.conf	是否启用共享内存段大小的自定义功能
apc.shm_size	30	php.ini httpd.conf	定义共享内存段的大小,单位是 MB
apc.optimization	0	php.ini httpd.conf .htaccess	一个实验性的配置,是否采用 APC 内置的优化功能,0 为否
apc.ttl	0	php.ini httpd.conf	定义一个新的缓存覆盖的时间间隔,默认为 0
apc.gc_ttl	3600	php.ini httpd.conf	定义一个缓存在垃圾回收列表中停留的时间,0 则表示不使用此功能。
apc.cache_by_default	on	php.ini httpd.conf	是否采用缓存文件的控制功能,如启用则可在 apc.filters 中定义哪些文件可以被缓存。
apc.filters	“ ”	php.ini httpd.conf	允许利用正则表达式来定义哪些文件可以被缓存。
apc.slam_defense	0	php.ini httpd.conf	允许使用 0 至 100 之间的数值,当多个进程在同一时间缓存同一文件时,此数值定义各个进程的一个竞争条件值。

## 3. 删除 APC

删除 APC 只需在 php.ini 中将 apc.so 注释即可。

```
/* php.ini */
.....
;extension = "apc.so"
.....
```

## 第5节 eAccelerator

eAccelerator 是一个和 APC 类似的对操作码进行处理的缓存引擎。它是 PHP4 中的 Turck MMCache 发展出来的一个分支程序。官方站点地址 <http://eaccelerator.net/>。目前最新稳定版本是 0.9.4。我们可以去站点

<http://tech.lupaworld.com/lamp/down.php?id=eaccelerator-0.9.4.tar.bz2> 下载 eaccelerator-0.9.4.tar.bz2, 放在 /root/soft 目录中。然后开始我们安装过程。

### 1. 安装 eAccelerator

- 解压缩。

```
[root@ lupa/]# cd /usr/local
[root@ lupa local]# tar xjvf /root/soft/eaccelerator-0.9.4.tar.bz2
```

- 进入源码目录, 使用 phpize 检查 PHP 安装的相关参数并创建 configure 文件。

```
[root@ lupa/]# /usr/local/php/bin/phpize
```

- 运行完成后, 我们将采用 configure 指定 apxs 和 php-config 的路径并创建 MakeFile 文件。

```
[root@ lupa eaccelerator-0.9.4]# ./configure \
--enable-eaccelerator=shared \
--with-php-config=/usr/local/php/bin/php-config
```

- 然后进行编译和安装。

```
[root@ lupa eaccelerator-0.9.4]# make
[root@ lupa eaccelerator-0.9.4]# make install
```

- 创建缓存目录并修改权限以便 Apache 的用户有权限操作。

```
[root@ lupa/]# mkdir /tmp/eaccelerator
[root@ lupa/]# chmod 0777 /tmp/eaccelerator
```

- 此时, eaccelerator.so 将出现在 PHP 的 extensions 目录中, 然后修改 php.ini 文件, 增加对 eaccelerator.so 的引用。

```
/* php.ini */
.....
extension = "eaccelerator.so"
.....
```

- 重新启动 Apache 即完成 eAccelerator 的安装。

## 2. 配置 eAccelerator

同 APC 一样, eAccelerator 也支持指令参数来控制它的行为。以下是各个指令的用法:

指令	默认	位置	描述
eaccelerator.shm_size	0	php.ini httpd.conf	定义 eAccelerator 所使用的共享内存大小, 0 则表示采用操作系统的默认数值。
eaccelerator.cache_dir	/tmp/eaccelerator	同上	定义缓存内容的存放路径。
eaccelerator.enable	1	php.ini httpd.conf .htaccess	定义是否采用 eAccelerator 的缓存功能。默认为 1, 即采用。
eaccelerator.optimizer	1	同上	采用 eAccelerator 的优化功能。
eaccelerator.debug	0	php.ini httpd.conf	是否启用调试模式, 默认为不启用。
eaccelerator.check_mtime	1	同上	控制 eAccelerator 是否检查此文件是否被更新过。
eaccelerator.compress	1	php.ini httpd.conf .htaccess	是否采用 Apache 的 deflate 模块或者 php 的 gzip 模块来进行输出内容的压缩处理。
eaccelerator.compress_level	9	同上	定义压缩级别, 9 为最大值。

## 3. 删除 eAccelerator

同样, 删除一个扩展只需要在 php.ini 中对此扩展进行注释即可。

```
/* php.ini */  
.....  
;extension = "eaccelerator.so"  
.....
```

当然为了彻底地删除此扩展, 我们可以到 php 的 extensions 目录中将 eaccelerator.so 删除。

## 第 6 节 Zend Optimizer

Zend Optimizer 是由 PHP 核心引擎“Zend”创建者 Zend 技术公司所开的免费 PHP 优化

软件。据 Zend 公司透露使用这个软件某些情况下至少可以提高性能 30% 以上。我们可以在 [http://www.zend.com/products/zend\\_optimizer](http://www.zend.com/products/zend_optimizer) 获得相关介绍信息。对应 PHP5.x 的最新稳定版是 3.0.0。下载地址：<http://tech.lupaworld.com/lamp>。

## 1. 安装 Zend Optimizer

Zend Optimizer 的安装很简单,我们将下载到的 ZendOptimizer-3.0.0-linux-glibc21-i386.tar.gz 放置在 /root/soft/ 目录中。

- 解压缩。

```
[root@lupa/]# cd/usr/local
```

```
[root@lupa local]# tar xzvf /root/soft/ZendOptimizer-3.0.0-linux-glibc21-i386.tar.gz
```

- 然后进入解压目录执行内置的自动安装的向导脚本。

```
[root@lupa/]# install.sh
```

- 根据它的提示进行操作。通常采用默认设置即可。在此过程中该脚本会备份 php.ini 文件,并将创建 php.ini 的符号链接到 Zend Optimizer 的安装目录中。并指定此文件作为 php 的配置文件。

## 2. 配置 Zend Optimizer

下面介绍针对 Zend Optimizer 的各个指令的用法:

指令	默认	位置	描述
zend_optimizer.optimization_level	15	php.ini httpd.conf	定义 Zend Optimizer 的压缩级别。各数值具体含义如下: 0:无压缩;1:级别 1;2:级别 2;4:级别 3;8:级别 4 将各个级别的数值相加得到的总和即可实现各个级别的压缩。15 即代表实现所有级别。
zend_optimizer.enable_loader	1	php.ini httpd.conf	是否采用 Zend Optimizer。

## 3. 删除 Zend Optimizer

首先删除在安装过程中创建的 php.ini 的符号链接。然后将 php.ini 的备份还原,重新启动 Apache 即可。当然也可以彻底删除掉 Zend Optimizer 的安装目录。

## 第 7 节 JPCache

JPCache 是采用 PHP 编写的一个缓存引擎方案,它不同于 APC 或者 eAccelerator。JPCache 不需要采用 DSO 或者编译成静态模块进行安装。它只需要在 PHP 的代码里使用 include 或者 require 方式调用即可。

### 1. 安装 JPCache

JPCache 官方站点地址是 <http://www.jp.cache.com/>。目前最新稳定版本是 jpcache v2。我们从这里下载 <http://tech.lupaworld.com/lamp/> 保存到 /root/soft/ 目录中。

- 解压缩处理。

```
[root@ lupa/]# cd /usr/local
[root@ lupa local]# unzip xzvf /root/soft/jpcache.v2.zip
```

- 配置 includedir。

```
/* jpcache.php */
.....
$ includedir = "/usr/local/jpcache";
.....
```

- 配置 jpcache 的类型,目前支持 file 和 mysql。

```
/* jpcache-config.php */
.....
$ JPCACHE_TYPE = "file"; //我们采用 file 类型
// $ JPCACHE_TYPE = "mysql";
// $ JPCACHE_TYPE = "dbm";--Not yet
.....
```

- 创建缓存目录。

```
[root@ lupa/]# mkdir /tmp/jpcache
[root@ lupa/]# chmod 0777/tmp/jpcache
```

- 测试。我们编写以下 PHP 程序进行测试。

```
/*
tesetjpcache.php
*/
```

```
<? php
require "/usr/local/jpcache/jpcache.php";
echo time();
phpinfo();
? >
```

- 将此页面保存到 /usr/local/apache/htdocs/lamp/ 目录中。
- 用浏览器打开 <http://www.lamp.com/testjpcache.php>。在不断的刷新访问过程中,如果显示的时间保持不变则说明 jpcache 生效了。

## 2. 配置 JPCache

下面介绍针对 Zend Optimizer 的各个指令的用法:

指令	默认	描述
\$ JPCACHE_TYPE	必须定义 (file 或者 mysql)	定义 JPCache 采用文件还是数据库来进行缓存的保存方式。
\$ JPCACHE_TIME	900	定义缓存的保存时间,负数表示禁止缓存,0 则表示立刻更新。
\$ JPCACHE_DEBUG	0	是否采用调试模式,0 为禁用。
\$ JPCACHE_IGNORE_DOMAIN	1	匹配文件时,是否检查主机名,假如服务器上只放置一个站点时则设为 1,表明忽略主机名。
\$ JPCACHE_ON	1	是否采用 JPCache 功能。
\$ JPCACHE_USE_GZIP	1	是否采用压缩输出,1 表示使用。
\$ JPCACHE_POST	1	是否对 POST 数据进行缓存,1 表示使用。
\$ JPCACHE_GZIP_LEVEL	9	定义压缩级别。
\$ JPCACHE_DIR	/tmp/jpcache	设定缓存文件保存路径。

## 3. 删除 JPCache

要取消 JPCache 只需在 PHP 代码中去掉对 jpcache.php 的引用即可。

## 第8节 memcached

和 APC 或者 JPCache 等已经介绍过的以处理输出为主要方式的缓存引擎不同,memcached 主要是缓存程序中的各个对象来实现对访问的加速。memcached 官方站点地址:<http://www.memcached.org/>



:<http://www.danga.com/memcached/>。目前最新稳定版是 memcached-1.1.12.tar.gz。我们可以从这里下载:<http://tech.lupaworld.com/lamp/>保存到/root/soft 目录。

## 1. 安装 memcached

- 解压缩。

```
[root@ lupa/]# cd /usr/local
[root@ lupa/]# tar zxvf /root/soft/memcached-1.1.12.tar.gz
```

- 编译安装。

```
[root@ lupa/]# cd /usr/local/memcached-1.1.12
[root@ lupa memcached]# ./configure
[root@ lupa memcached]# make
[root@ lupa memcached]# make install
```

- 启动 memcached 服务程序。分配给 memcached 128M 内存,在 IP 为 127.0.0.1 的 10101 端口监听。

```
[root@ lupa/]# /usr/local/bin/memcached-d-m 128-l 127.0.0.1-p 10101
```

- 现在 memcached 已经启动成功。我们将使用 memcached 的 PHP 扩展来调用这个功能。先到 <http://pecl.php.net/package/memcached> 下载最新稳定版也可以到 <http://tech.lupaworld.com/lamp/> 去下载。memcached-2.0.4.tgz 保存到/root/soft 目录中。
- 解压缩处理。

```
[root@ lupa/]# cd /usr/local
[root@ lupa local]# tar zxvf /root/soft/memcached-2.0.4.tgz
```

- 编译安装。

```
[root@ lupa/]# cd /usr/local/memcached-2.0.4
[root@ lupa memcached-2.0.4]# /usr/local/php/bin/phpize
[root@ lupa memcached-2.0.4]# ./configure
[root@ lupa memcached-2.0.4]# make
[root@ lupa memcached-2.0.4]# make install
```

- 修改 php.ini 文件,增加对 memcached.so 的支持。

```
/* php.ini */
.....
extension = "memcached.so";
.....
```

- 重新启动 Apache。至此 memcached 配置完毕。

## 2. 使用 memcached

memcached 的使用发放与之前介绍的缓存模块完全不同,它将采用代码嵌入的方式进行调用,以下先介绍各个函数,并以一个小例子来说明。

```
bool memcached::add ( string key, mixed var[ , int flag[ , int expire]] )
//增加一个变量或者对象到缓存,并指定一个序列号。如果相同序列号再次被添加
则返回 FALSE。

bool Memcached::close ( void ) //关闭进入缓存服务器的非持久连接。
bool memcached::connect ( string host[ , int port[ , int timeout]] )
//与缓存服务器建立连接。
int memcached::decrement ( string key[ , int value] )
//可根据序列号来减少已缓存的数值。如果该序列号不存在,则被定义为 1。
bool memcached::delete ( string key[ , int timeout] )
//删除一个已经缓存的对象。根据序列号来指定。
bool memcached::flush ( void )
//将所有已缓存的对象设置超时。此时新的缓存对象将被建立,但内存并不立即被
释放。
string memcached::get ( string key )
string memcached::get ( array keys )
//根据设置的序列号来获取在缓存中的相应对象。
array memcached::getStats ( void )
//获取缓存状态,包括总连接数,已经缓存的对象总数和服务器时间。
string memcached::getVersion ( void ) //获取缓存服务器的版本号。
int memcached::increment ( string key[ , int value] )
//可根据序列号来增加已缓存着的数值。如果该序列号不存在,则被定义为 1。
bool memcached::pconnect ( string host[ , int port[ , int timeout]] ) //建立一个持久
连接。
```

以下是一个简单的实例。

```
<? php
class Circle
{
public $ radius;
public function area()
{
return pi() * pow( $ this->radius, 2);
```

```
}  
}  
$ memc = new memcached();  
$ memc->connect("127.0.0.1", 11211);  
$ c = new Circle();  
$ c->radius = 15;  
$ memc->set("circle1", $ c);  
$ stats = $ memc->getStats();  
$ memc->close();  
print_r( $ stats);  
? >
```

执行后将得到以下结果：

```
Array  
(  
    [pid] => 1379  
    [uptime] => 4148  
    [time] => 1124758570  
    [version] => 1.1.12  
    [rusage_user] => 0.030000  
    [rusage_system] => 0.010000  
    [curr_items] => 1  
    [total_items] => 8  
    [bytes] => 77  
    [curr_connections] => 1  
    [total_connections] => 9  
    [connection_structures] => 2  
    [cmd_get] => 0  
    [cmd_set] => 8  
    [get_hits] => 0  
    [get_misses] => 0  
    [bytes_read] => 605  
    [bytes_written] => 1856  
    [limit_maxbytes] => 134217728  
)
```

### 3. 删除 memcached

正如其他的基于 DSO 方式安装的缓存引擎一样。我们可以迅速的取消对它的使用。包括在 `php.ini` 文件中注释掉对 `memcached.so` 的使用,然后重启 Apache。再结束 `memcached` 的进程,并删除 `memcached` 安装目录。

## 第9节 如何选用缓存引擎

是否可以混用以上介绍的各种缓存引擎来发挥更大的性能? JPCache 和 `memcached` 是可以很好地同时工作,因为他们的工作原理不同。但是 APC, eAccelerator 和 Zend Optimizer 却存在一定的问题。通常建议只使用一种缓存引擎。首先我们需要分析系统需求,根据实际情况来确认所需要的 PHP 缓存引擎。假如程序中包含了大量的对象、频繁地从数据库中读取和存储数据,那么 `memcached` 比较适合这种情况。假如站点中包含了大量的静态页面,基本上不存在动态执行程序那么使用 JPCache 最为合适。假如数据层不是性能瓶颈,但是在 PHP 脚本执行时可能会造成部分影响,那么所有基于操作码级别的缓存引擎都可以采用。假如实在无法确定采用何种缓存引擎,那么使用 APC 是最为安全的解决方案。

# 基于 LAMP 实例配置

本章介绍基于 LAMP 平台的实例配置,通过本章学习,可以很容易地在 LAMP 平台上搭建你喜欢的系统,真正体会到 LAMP 平台的功能。在学习本章之前,你必须掌握 LAMP 平台的安装和配置。如果不熟悉,请先掌握前几章内容。

## 第1节 架设 BLOG 平台

### 1. BLOG 概况

根据维基百科记载:BLOG 起源于 Weblog,意思是网络日志。1997 年由 Jorn Barger 所提出。1999 年,Peter Merholz 首次使用缩略词“BLOG”,成为今天常用的术语。但是,BLOG 真正开始快速发展的转折点,是在 1999 年 6 月,当时 Pitas 开始提供免费的 BLOG 服务,紧接着 8 月,Pyra lab 推出了 blogger.com(2003 年被 Google 收购)。blogger.com 提供了简单易学的说明,以及能通过 FTP 直接将 BLOG 发表在个人网站上的功能,这带给使用者很大的方便。目前已经有了很多 BLOG 服务供应商(BSP),业内人士对其盈利前景,持谨慎乐观态度。Weblog 指的是以网页作为呈现媒介的个人日记,也有人把它称作网页形态的日记。1999 年 Peter Merholz 开始把将 Weblog 念成 We BLOG,因而有了 BLOG 这个说法。此时的 BLOG 不仅可以拿来当名词或 Weblog 的缩写,更成为一个动词。BLOG 是个人或群体以时间顺序所作的一种记录,并且不断更新。BLOG 之间的交流主要是通过回溯引用(Track-Back)和广播、留言、评论的方式来进行的。BLOG 的操作管理用语,也借鉴了大量档案管理用语。一个 BLOG 亦可被视为一个档案,或是卷宗。与传统档案不同的是,BLOG 的作者既是这份档案的创作人,也是其档案管理人。而书写 BLOG 的人,通常称为“blogger”(即 BLOG 作者),既可以是一个人撰写,也可以是一群人写。

BLOG 大量采用了 RSS(Really Simple Syndication 或者 Rich Site Summary 或者 RDF Site Summary)技术,所有的 RSS 文件都必须符合由 W3C 发布的 XML 1.0 规范。对读者来说,可以通过 RSS 订阅一个 BLOG,确知该 BLOG 作者最近的更新。对 BLOG 作者来说,RSS 可以使自己发布的文章易于被计算机程序理解并摘要。对知识管理和创造而言,BLOG 提供了新的形态和途径。对汉语为母语的人而言,BLOG 写作既接续了汉语笔记文学的优秀传统,更充分鼓励了个人表达。从交往形态考察,BLOG 空间设定了积极的读者、作者、编者互动

转换关系,“言者互重,阅者相惜”。

## 2. 开始部署 WordPress

了解了 BLOG 的历史和概念之后,接下来重点就是介绍如何去部署属于自己的 BLOG 系统,推荐最优秀的开放源代码 BLOG 系统 WordPress,基于 GNU 通用公共许可证下授权发布。先让我们认识一下 WordPress:使用 PHP 语言开发的 BLOG 平台,用户可以在支持 PHP 和 MySQL 数据库的服务器上架设自己的 BLOG 平台。你也可以认为 WordPress 就是一个个人信息发布平台。

## 3. 获得 WordPress

获得最新的 WordPress 系统(本书采用 WordPress 2.0.4,你可以根据自己偏好,下载不同的版本),官方下载地址:<http://WordPress.org/latest.tar.gz>。我们也推荐速度较快的 LUPA 社区,下载地址:<http://tech.lupaworld.com/lamp/>下载并解压缩。假设我们下载到/root 目录下:

```
[root@lupa/]# cd root //进入 root 目录;  
[root@lupa ~]# tar zxvf latest.tar.gz //解压安装包;
```

解压后会生成 WordPress 目录,把这个目录移动到可执行环境目录下,如:/usr/local/apache/htdocs。

```
[root@lupa ~]# mv WordPress /usr/local/apache/htdocs //移动到目标目录下;
```

注意,如果你不是 root 用户,可能会遇到权限的问题,因此确保你是 root 用户。

## 4. 创建数据库和用户

你可以使用 phpMyAdmin 或 MySQL 客户端来创建 WordPress 数据库和用户。如果你对此不熟悉,请复习第 3 章内容。

### 1) 使用 phpMyAdmin 来创建数据库

phpMyAdmin 是一个用 PHP 编写出来的用来控制和操作 MySQL 数据库的开源软件,可以通过浏览器完全对数据库进行操作,包括建立、复制、删除、查询数据等等。现在你有了 phpMyAdmin 就可以完全不使用 MySQL 命令,直接使用 phpMyAdmin 来管理 MySQL 的数据库。如果你的 Web 服务器安装了 phpMyAdmin,那么创建 WordPress 数据库和用户是件非常简单的事。本教材使用的版本为 phpMyAdmin 2.8.1。phpMyAdmin 的用户界面根据版本的不同会有所变化。

#### ● 创建数据库

现在为 WordPress 的数据库起个名字,比如:WordPress,填入创建一个新的数据库文本框,然后点击创建。就这么简单,已经完成创建数据库了。如图 6-1:



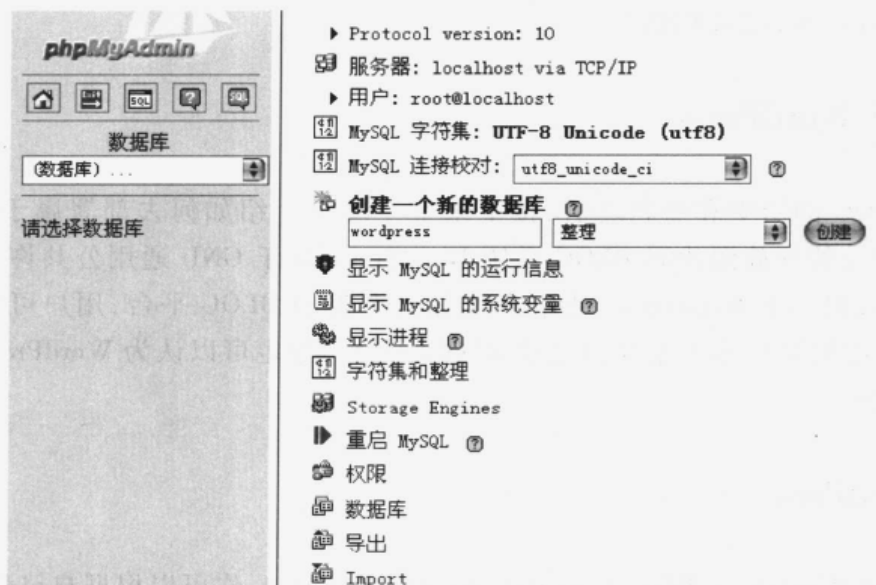



图 6-1

### ● 创建用户

点击左上角的主目录  图标, 返回到首页, 然后点击权限。出现图 6-2。

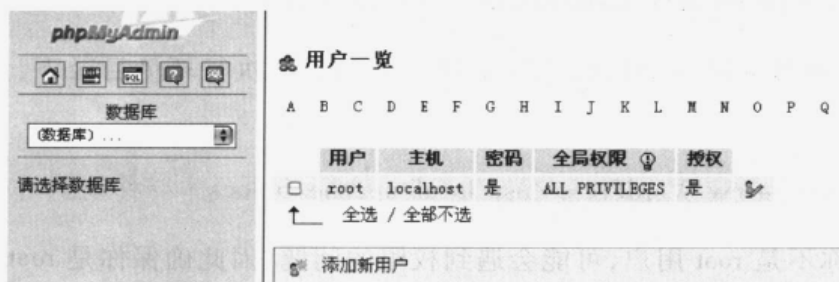


图 6-2

点击图 6-2 中的添加新用户后, 进入图 6-3 图面, 现在为 WordPress 的数据库起一个用户名(比如 lupablog)并将其填入用户名文本框。选择一个你常用的安全密码(大小写字母、数字和符号组合的密码是最理想的), 并将其填入密码框。再次输入同一密码到重新输入密码框。记下刚才选择的用户名和密码, 点击执行。

### ✎ 添加新用户

登入信息

用户名:	使用文本域:	<input type="text" value="lupablog"/>
主机:	任意主机:	<input type="text"/>
密码:	使用文本域:	<input type="password" value="*****"/>
重新输入:		<input type="password" value="*****"/>
Generate Password:	<input type="button" value="Generate"/>	<input type="button" value="Copy"/>

图 6-3

接下来为这个用户添加权限。回到主目录  图标,返回到首页,然后点击权限。如图 6-4,在用户 lupablog 的后面点击编辑权限 .

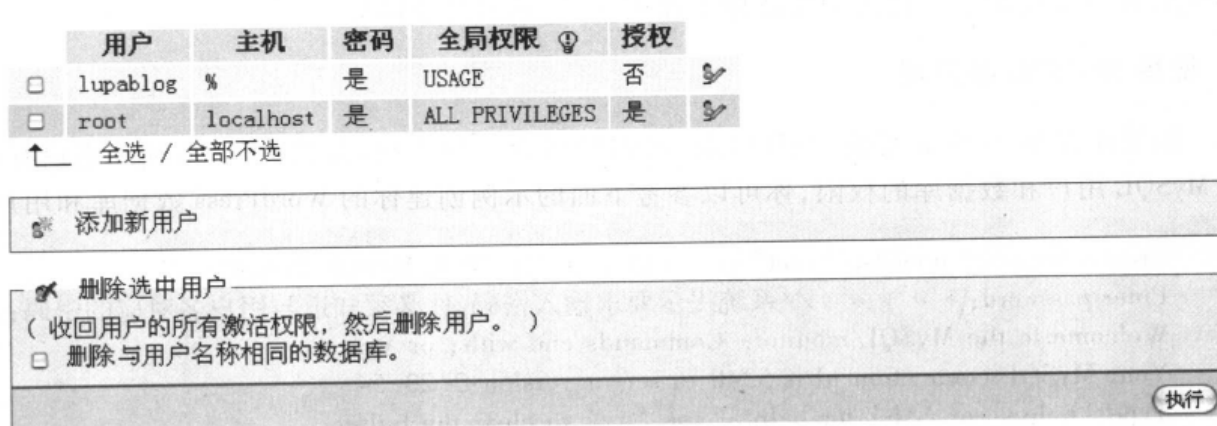


图 6-4

图 6-5 是数据库指定权限画面,在[使用文本域]中点击下拉菜单,找到 wordpress 数据库,点击执行。

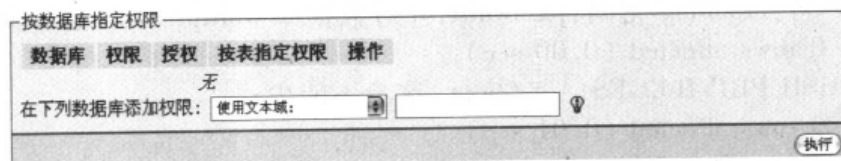


图 6-5

现在对 wordpress 数据库权限进行操作,我们选择全选,点击执行。如图 6-6。

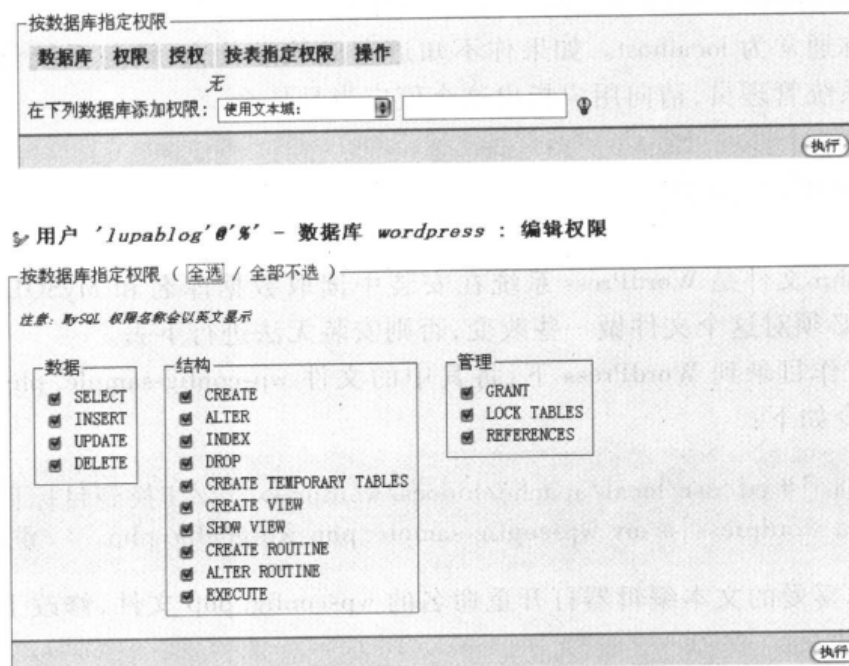


图 6-6

现在为止,你已经建立了用户 lupablog 及其对数据库 wordpress 操作的权限。你可以不用去了解如何使用 MySQL 客户端来创建数据库和用户。但是我们建议初学者先使用 MySQL 客户端用命令方式去创建数据库和用户,以熟悉命令环境。

## 2) 使用 MySQL 客户端

如果你有 Web 服务器的 Shell 权限,习惯使用命令行工具,并且你的 MySQL 用户有创建 MySQL 用户和数据库的权限,你可以参考下面的示例创建你的 WordPress 数据库和用户。

```
[root@lupa/]# mysql-u "root"-p //以 root 用户进入 MySQL 数据库;
Enter password: * * * * //系统提示要求输入密码,你必须知道与用户名对应的密码;
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 5340 to server version:3.23.54
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
//出现以上提示表示成功进入系统;
mysql> CREATE DATABASE "wordpress"; //建立 wordpress 数据库;
Query OK, 1 row affected (0.00 sec) //系统提示;
mysql> GRANT ALL PRIVILEGES ON "wordpress". * TO "lupablog"@ "localhost" \
IDENTIFIED BY "password";
//添加用户名:lupablog 密码:password,并对数据库 wordpress 的全部操作权限。
Query OK, 0 rows affected (0.00 sec)
mysql> FLUSH PRIVILEGES; //执行,这步不能少;
Query OK, 0 rows affected (0.01 sec)
mysql> EXIT //退出系统;
```

根据上面的操作,应该记住以下数据,在安装时会用到这些数据:

- ◆ 数据库:wordpress
- ◆ 用户名:lupablog 密码:password(自己设定)
- ◆ 主机名称通常为 localhost。如果你不知道它应该是什么,请与系统管理员联系。如果你是系统管理员,请向用户指出这个值应当是什么。

## 5. 设置 wp-config. php

wp-config. php 文件是 WordPress 系统在安装中读取数据库名和 MySQL 用户名及密码的脚本文件,你必须对这个文件做一些改变,否则安装无法进行下去。

现在切换工作目录到 WordPress 下,将其中的文件 wp-config-sample. php 重命名为 wp-config. php。命令如下:

```
[root@lupa/]# cd/usr/local/apache/htdocs/wordpress //切换到目标目录;
[root@lupa wordpress]# mv wp-config-sample. php wp-config. php //重命名;
```

接下来用你喜爱的文本编辑器打开重命名的 wp-config. php 文件,修改上面提到的那部分代码,具体做法:

```
[root@lupa wordpress]# vi wp-config. php //用 vi 打开 wp-config. php 文件;
```

你需要修改的部分是：

```
// * * MySQL settings * * //  
define( 'DB_NAME', 'wordpress' ); //数据库名称,必须与你创建的数据库一致;  
define( 'DB_USER', 'lupablog' ); //MySQL 用户名;  
define( 'DB_PASSWORD', 'password' ); //密码;  
define( 'DB_HOST', 'localhost' ); //在 99% 的情况下您无需修改这个值;
```

退出并保存这个文件。

## 6. 执行安装脚本 install.php

打开 Web 浏览器,输入: `http://localhost/WordPress/wp-admin/install.php`, 如果以上几步你都对了,那么你将看到图 6-7:



图 6-7

点击 First Step >> 后,你将看到图 6-8,输入你想要的 BLOG 标题和 Email,点击 Continue to Second Step >>。

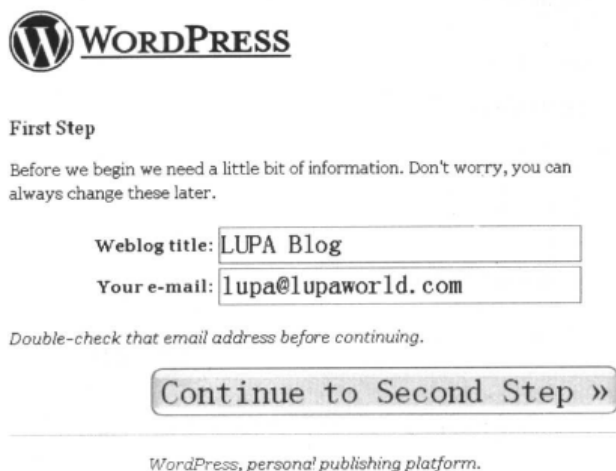


图 6-8

点击 Continue to Second Step 后,系统将建立数据表,这里大概需要 3 ~ 5 秒钟的时间,系统提示安装结束。注意了,现在你看到的是安装结束的画面,图 6-9,请记住系统给你的登入名和密码(密码可以通过后台修改)。



### Second Step

Now we're going to create the database tables and fill them with some default data.

*Finished!*

Now you can log in with the **username** "admin" and **password** "bfe217".

**Note that password** carefully! It is a *random* password that was generated just for you. If you lose it, you will have to delete the tables from the database yourself, and re-install WordPress. So to review:

Username  
admin  
Password  
bfe217  
Login address  
[wp-login.php](#)

Were you expecting more steps? Sorry to disappoint. All done! :)

图 6-9

这时所有的 WordPress 安装结束了。点击 login,进入图 6-10 画面,输入用户名:admin 密码:bfe217,进入后台控制中心,开始你的 BLOG 之旅吧!

A screenshot of the WordPress login interface. At the top is the WordPress logo. Below it, the text 'Username:' is followed by a text input field containing 'admin'. Below that, the text 'Password:' is followed by a text input field containing six asterisks '\*\*\*\*\*'. There is a checkbox labeled 'Remember me' below the password field. To the right of the password field is a button labeled 'Login »'. At the bottom, there are two links: « Back to blog and [Lost your password?](#)

图 6-10

点击 Login 进入 WordPress 的后台,见图 6-11。

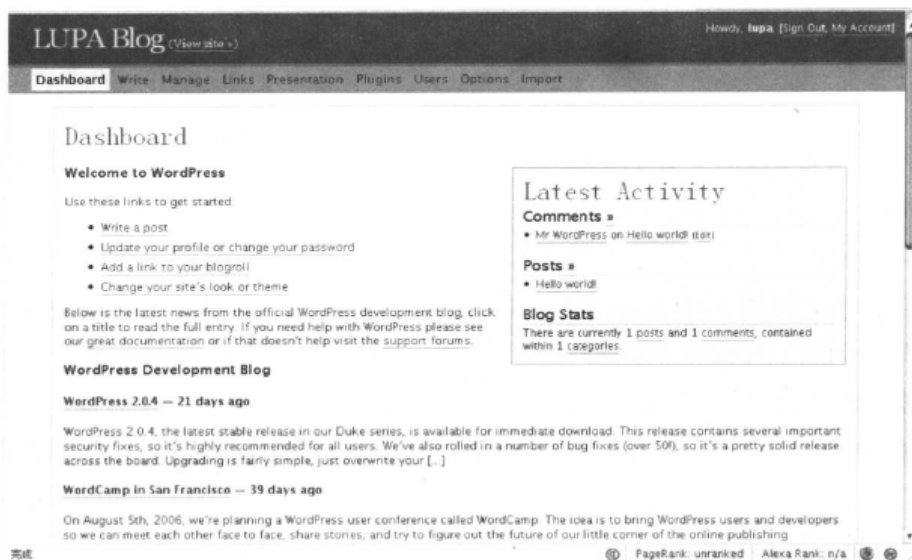


图 6-11

如果不习惯英文界面的读者,可以安装中文语言包。具体做法如下:

- ◆ 在 <http://tech.lupaworld.com/lamp/> 下载中文语言包 zh\_CN.mo。
- ◆ 在 wordpress/wp-includes 目录下建立 languages 文件夹。
- ◆ 将 zh\_CN.mo 文件上传到 wp-includes/languages 文件夹。
- ◆ 修改 WordPress/wp-config.php 文件,将其中的 `define('WPLANG', '');` 修改为: `define('WPLANG', 'zh_CN');`。
- ◆ 刷新一下,就可以看到中文界面。

## 7. 如何使用 WordPress

- ◆ 进入 BLOG 后台管理中心: <http://localhost/WordPress/wp-admin>, 输入正确的用户名和密码就可以登录了。
- ◆ 撰写 BLOG: 点击撰写菜单。WordPress 提供了一个非常好用的可视化编辑器。点击保存, 可把 BLOG 暂时保存起来, 而不在首页显示出来。点击发布, 你的 BLOG 就呈现在首页。
- ◆ 更换 BLOG 主题风格: 你是不是不喜欢你的 BLOG 首页风格, 根据你的自己的偏爱, 可以到: <http://themes.WordPress.net> 去下载主题。把下载来的文件解压缩后, 放到 WordPress/wp-content/themes 目录下。点击外观, 你喜欢的那个主题风格就出现。
- ◆ 添加插件: 你可以在网上找到一些 WordPress 的插件, 比如播放 Flash、视频等等的插件, 下载回来后, 解压缩到 WordPress/plugins 目录下, 到管理中心的插件激活, 就可享受该插件的功能了。

有关 WordPress 的其他功能, 请参考官方手册。本文仅介绍 BLOG 平台的搭建。



## 第2节 架设 Wiki 平台

### 1. Wiki 概况

Wiki 是一种可在网络上多人协同创作的超文本系统,允许任何人能快速轻易的添加、删除、编辑所有的内容,因此特别适合团队合作的写作方式。目前互联网越来越多的网站都提供 WIKI 服务,最著名的是 Wikipedia,中文名称叫维基百科,其所有的文本内容在 GNU 自由文档许可证(FDL)下发布。也就是意味着,维基百科上的内容是自由复制、修改和发布的。

### 2. 开始部署 MediaWik

Wikipedia 基于 MediaWiki 系统架构,接下来将详细介绍 MediaWiki 的安装和使用。只要按本文所说的做下去,就可以轻松架设与 Wikipedia 一模一样的系统平台。在安装之前,我们先了解一下什么是 MediaWiki?

#### ● MediaWiki 简介

MediaWiki 是基于 GPL 发行的 WIKI 程序。它具有很丰富的功能,性能优秀,被维基百科及其他提供维基服务的网站所采用。也是目前全球最流行的维基系统之一。MediaWiki 采用 PHP 语言和 MySQL 数据库,是跨平台软件。目前最稳定的版本是 MediaWiki1.7。

### 3. 获得 MediaWiki

通过以下网址 MediaWiki: <http://tech.lupaworld.com/lamp/> 选择你需要的版本。推荐网址: <http://prdownloads.sourceforge.net/wikipedia/mediawiki-1.7.1.tar.gz?download>。本文以 MediaWiki1.7.1 为例,下载后得到 MediaWiki-1.7.1.tar.gz 文件。假如安装文件在 root 目录下,现在解压并安装。

```
[root@lupa ~]# tar -zxvf mediawiki-1.7.1.tar.gz //解压 mediawiki-1.7.1.tar.gz;
```

把解压后的文件全部移动到可执行环境下,如:/usr/local/apache/htdocs。

```
[root@lupa ~]# mv mediawiki-1.7.1/usr/local/apache/htdocs/ //移动到目标目录下;
```

```
[root@lupa htdocs]# mv mediawiki-1.7.1/wiki //将 mediawiki-1.7.1 重命名 wiki;
```

### 4. 创建数据库和用户

你可以使用 phpMyAdmin 或 MySQL 客户端来创建 WIKI 数据库和用户。具体操作请参考本章第一节架设 BLOG 平台。

建立数据库:wiki

用户:lupawiki

密码:password

你可以试着用 MySQL 客户端来创建,具体命令如下:

```
mysql > CREATE DATABASE "wiki";  
mysql > GRANT ALL PRIVILEGES ON "wiki". * TO "lupawiki" @ "localhost"  
IDENTIFIED BY "password";
```

## 5. MediaWiki 安装与配置

### 1) 增加 config 目录的写权限

```
[root@ lupa htdocs]# cd wiki  
[root@ lupa wiki]# chmod a + w config //增加 config 目录写权限;
```

### 2) 开始 MediaWiki 安装

打开 Web 浏览器,输入: <http://localhost/wiki/> 开始 MediaWiki 安装。如图 6-12。



## MediaWiki 1.7.1

Please setup the wiki first.

图 6-12

### 3) 填写安装数据

点击 Setup the wiki 进入安装画面,以下内容必填,其他默认设置,请参考本书设置:

```
Wiki name:LUPA Wiki //wiki 的名字;  
Language:zh-cn-中文(简体) //选举中文简体;  
Admin username:admin //管理员用户名;
```

```

Password:admin //管理员密码;
Password confirm:admin //重复密码;
Database host:localhost //数据库服务器;
Database name:wiki //数据库名;
DB username:lupawiki //数据库管理用户;
DB password:password //数据库密码;
DB password confirm:password //重复密码;

```

#### 4) 安装完毕

点击 Install MediaWiki 开始安装。10 秒钟左右的时间,你将看到 install successful 提示! 恭喜你成功安装! 如图 6-13。

Installation successful! Move the config/LocalSettings.php file into the parent directory, then follow this link to your wiki.

图 6-13

#### 5) 安装后设置

拷贝 config 目录下的 LocalSettings.php 到上一级目录 wiki 下,之后再删除 config 目录 (千万别忘了做这一步,不然很不安全!)。命令如下:

```

[root@ lupa wiki]# cp config/LocalSettings.php
[root@ lupa wiki]# rm config

```

到此,MediaWiki 就安装好了。如图 6-14。



图 6-14

## 6. 如何使用 MediaWiki

- ◆ 创建一个新账号:你要编辑新建 Wiki 页面前,得先拥有一个账号,如果没有,就创建一个吧。点击用户登录,再 Create an account。
- ◆ 编辑页面:登录之后,看到编辑了吧,点击,就可以编辑该页面了。
- ◆ 新建一个页面:新建页面非常简单,你可以通过 Web 浏览器,访问主页,比如输入: `http://localhost/wiki/index.php/页面名`,这里的“页面名”替换为你想要的名称,比如你想建一个 lupa 的页面,页面名就写 lupa。如果 lupa 页面不存在,则就进入了页面编辑模式,你就可以在这上面书写相关内容。
- ◆ 删除一个页面:在编辑模式下,删除页面的所有内容,该页面就自动删除了。
- ◆ MediaWiki 功能非常强大,内置一些简单的页面编辑代码。要熟练掌握 WIKI 的使用,请参考官方手册。

## 第3节 架设 CRM 平台

客户关系管理(Customer Relationship Management 或简称 CRM),企业活动面向长期的客户关系,以求提升企业成功的管理方式。其主要目的是要协助企业管理销售循环:新客户的招徕、保留旧客户、提供客户服务及进一步提升企业和客户的关系。

在国内中小企业的信息化较为落后,CRM 系统的普及率也相对较低。市场上 CRM 的价格过高,性能和功能无法满足企业的需求,也成了企业应用 CRM 系统较少的直接原因。在企业中部署 CRM 系统,尤其是中小企业,我们推荐用户使用世界上最优秀的开源 CRM 系统 SugarCRM。它完全的开放,用户可以自由获得其源代码,目前它是全世界使用最多的开源 CRM 系统。

### 1. 开始部署 SugarCRM

本节将介绍 SugarCRM 的安装配置和使用,通过本节的学习,你可以快速部署企业 CRM 平台。先让我们认识一下 SugarCRM。

#### ● SugarCRM 简介

SugarCRM 是最成功的开源 CRM 系统。它具有很强的可操作性、灵活的适用性、完善的功能、独特的个性化设计和专业的技术安全保障,是企业实现客户关系管理的理想之选。SugarCRM 系统在企业管理人员、销售人员和客户之间建立了一个立体的管理系统,帮助企业对内和对外实现以“客户为中心”的全方位管理。

### 2. 获得 SugarCRM

通过以下网址获得 SugarCRM: <http://tech.lupaworld.com/lamp/> 选择你需要的版本。推

荐官方网站下载: <http://www.sugarcrm.com>。本文以 SugarCRM 4.2.1 为例,目前这个版本是最稳定的。下载后得到 SugarSuite-4.2.1b.zip 文件,假如文件下载到 root 目录下,解压并安装。

```
[root@ lupa root]# unzip SugarSuite-4.2.1b.zip //当前目录下解压;
```

把解压后的文件全部移动到可执行环境下,如: /usr/local/apache/htdocs。

```
[root@ lupa root]# mv SugarSuite-Full-4.2.1b /usr/local/apache/htdocs/ //移动文件;
```

```
[root@ lupa htdocs]# mv SugarSuite-Full-4.2.1b/crm //重命名为 crm;
```

### 3. 创建数据库和用户

你可以使用 phpMyAdmin 或 MySQL 客户端来创建 CRM 数据库和用户。具体操作请参考本章第一节架设 BLOG 平台。

建立数据库:sugarcrm

用户:lupacrm

密码:password

### 4. SugarCRM 安装与配置

SugarCRM 的安装非常简单,安装方法跟前二节讲的 BLOG 和 WIKI 安装类似,一个好的系统,安装脚本文件做的也相当好。SugarCRM 的安装脚本文件做得非常不错,部署一个 CRM 系统极其简单。

#### 1) 增加相关目录和文件的写权限

默认情况下你需要增加相关目录的写权限,一般官方说明文件都会指出,或者你在安装时,系统会检查到相关目录和文件只读或可写,这里你需要对 cache、custom、data、modules 和 config.php 文件进行权限设置,命令如下:

```
[root@ lupa root]# cd /usr/local/apache/htdocs/crm
```

```
[root@ lupa crm]# chmod -R a + w cache custom data modules config.php
```

//设置 cache custom data modules config.php 文件权限;

#### 2) 开始安装 SugarCRM

打开 Web 浏览器,输入: <http://localhost/crm>,开始 SugarCRM 安装。如图 6-15。

点击 Start 开始安装。第一步就是在安装这个软件前的许可协议,你必须接受 SugarCRM 的协议,才能正常安装。SugarCRM 基于 MPL 许可证发布,MPL 全称是 Mozilla Public License,是经 OSIA 认证的开放源代码软件软件许可证之一。有些刚接受开源软件的朋

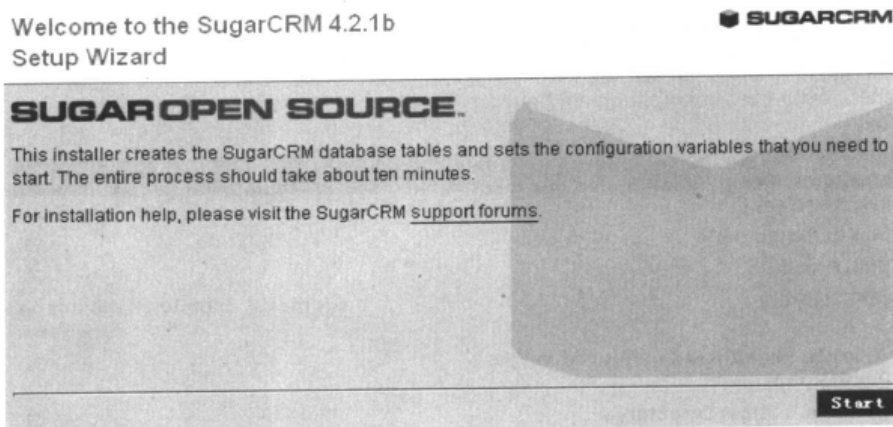


图 6-15

友可能会有所疑问,怎么有那么多种许可证?实际上开源软件许可证有上百种,而经过 OS-IA 认证的,目前还不到 30 种,凡通过了 OSIA 认证的许可证,都体现类似精神,就是源代码开放,自由传播、修改、发布,可能有些细节情况在各种许可证证中有所区别。现在我们继续安装。如图 6-16,选择 I Accept,然后 Next。

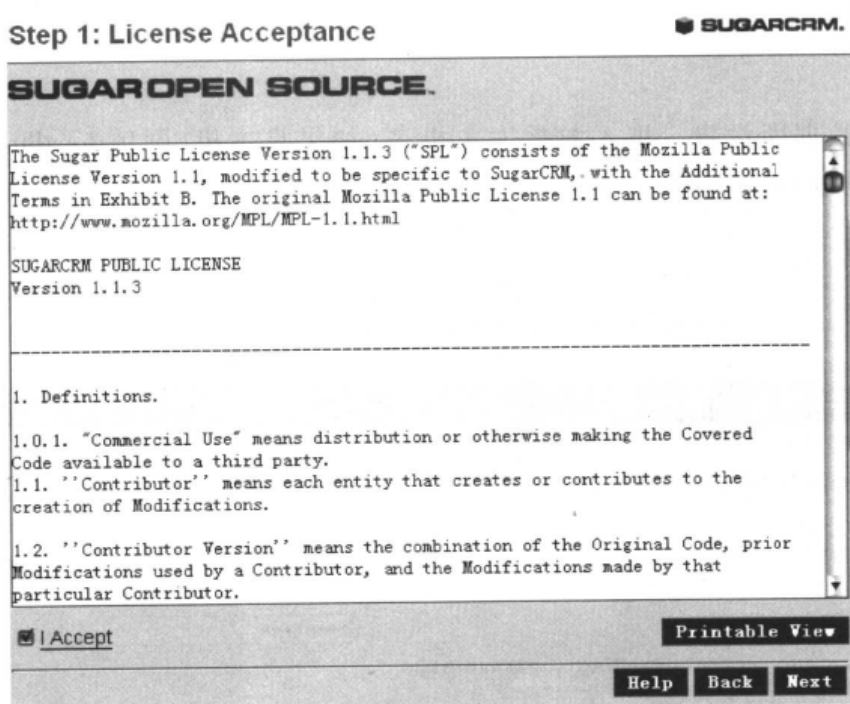


图 6-16

### 3) 系统检测

进入到 System Check Acceptance 的时候,对各个服务进行检测,看 Status,如果有红色,则检查一下相关选项。直到全绿,才能 Next。见图 6-17。



## Step 2: System Check Acceptance

SUGARCRM.

In order for your SugarCRM installation to function properly, please ensure all of the system check items listed below are green. If any are red, please take the necessary steps to fix them.

Component	Status
PHP version	OK (ver 5.1.4)
MySQL Database	OK
XML Parsing	OK
cURL Library	Not found: scheduler will not be functional
Writable SugarCRM Configuration File (config.php)	OK
Writable Custom Directory	OK
Writable Modules Sub-Directories and Files	OK
Writable Data Sub-Directories	OK
Writable Cache Sub-Directories	OK
Writable Session Save Path (G:\Program Files\xampp\tmp)	OK
PHP Safe Mode Turned Off	OK
PHP Allow Call Time Pass Reference Turned On	OK
PHP Memory Limit >= 32M	OK (No Limit)

图 6-17

## 4) 数据库设置

这时,到了数据库设置。填入数据库主机名(如果是本机,则填 localhost),数据库名:sugarcrm;用户:lupacrm;密码:password。如图 6-18 点击 Next,进入下一个配置项。

## Step 3: Database Configuration

SUGARCRM.

Please enter your database configuration information below. If you are unsure of what to fill in, we suggest that you use the default values.

\* Required field

## Database Configuration

* Host Name	<input type="text" value="localhost"/>	
* Database Name	<input type="text" value="sugarcrm"/>	<input type="checkbox"/> Create database
* Database username	<input type="text" value="lupacrm"/>	<input type="checkbox"/> Create user
Database password	<input type="password" value="*****"/>	
Drop and recreate existing Sugar tables?	<input type="checkbox"/>	
Caution: All Sugar data will be erased if this box is checked.		
Populate database with demo data?	<input type="checkbox"/>	
Database account above is a privileged user?	<input checked="" type="checkbox"/>	

Help

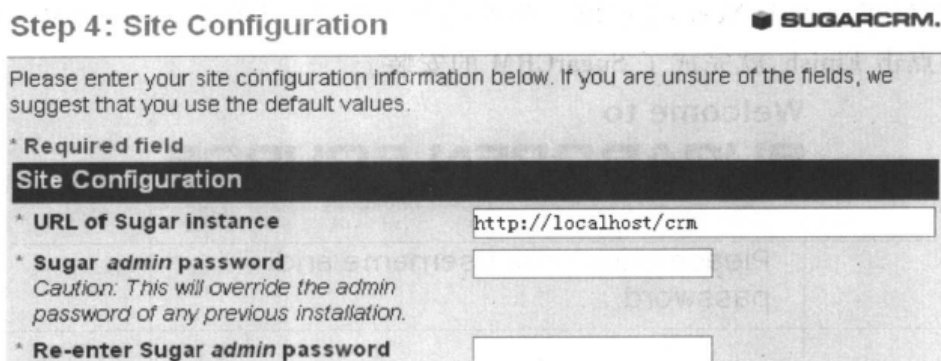
Back

Next

图 6-18

## 5) 站点配置

站点配置。这里输入管理员密码就行了,其他配置选项默认。图 6-19,点击 Next。



**Step 4: Site Configuration** SUGARCRM.

Please enter your site configuration information below. If you are unsure of the fields, we suggest that you use the default values.

\* Required field

**Site Configuration**

\* URL of Sugar instance

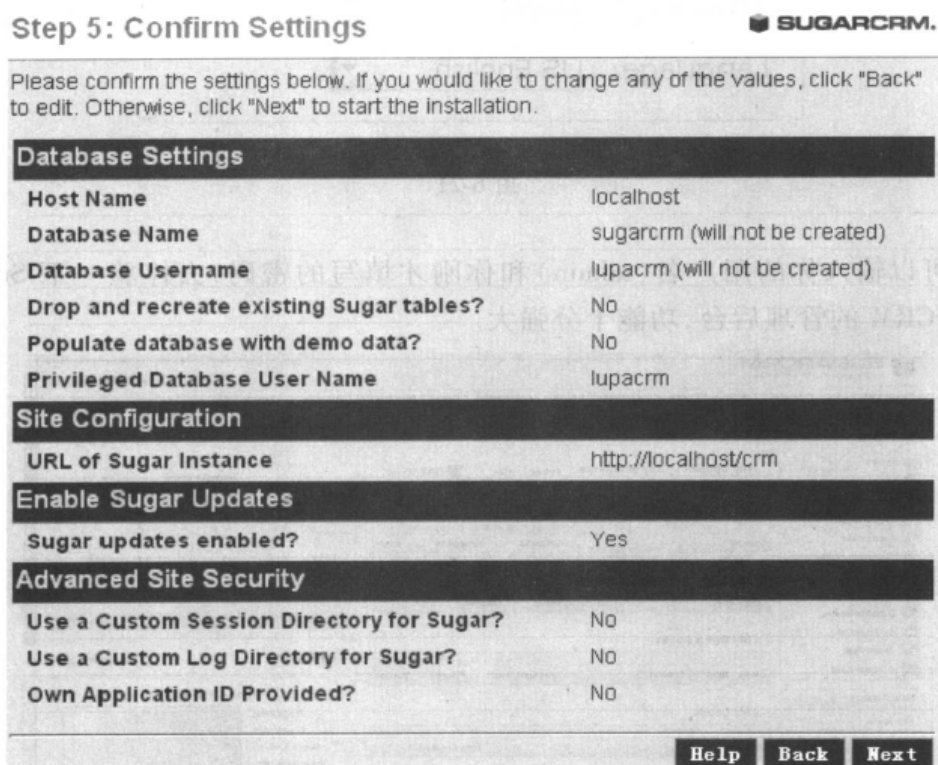
\* Sugar admin password   
Caution: This will override the admin password of any previous installation.

\* Re-enter Sugar admin password

图 6-19

## 6) 确认设置

确认配置信息。正确了吗? 如果设置正确无误,Next。有误,Back 返回修改。如图 6-20。



**Step 5: Confirm Settings** SUGARCRM.

Please confirm the settings below. If you would like to change any of the values, click "Back" to edit. Otherwise, click "Next" to start the installation.

**Database Settings**

Host Name	localhost
Database Name	sugarcrm (will not be created)
Database Username	lupacrm (will not be created)
Drop and recreate existing Sugar tables?	No
Populate database with demo data?	No
Privileged Database User Name	lupacrm

**Site Configuration**

URL of Sugar Instance	http://localhost/crm
-----------------------	----------------------

**Enable Sugar Updates**

Sugar updates enabled?	Yes
------------------------	-----

**Advanced Site Security**

Use a Custom Session Directory for Sugar?	No
Use a Custom Log Directory for Sugar?	No
Own Application ID Provided?	No

[Help](#) [Back](#) [Next](#)

图 6-20

## 7) 安装完成

大概需要 2~5 秒,在页面的底部出现:The setup of Sugar 4.2.1 is now complete。SugarCRM 就安装好了。点击 Next,这里填一些你的个人信息,反馈给 SugarCRM。你也可以选择不填,直接点击 Finish,就完成了 SugarCRM 的安装。

**Welcome to  
SUGAR OPEN SOURCE.**

Please enter your username and password.

User Name:

Password:

Options

Theme:

Language:

图 6-21

现在你可以输入你的用户名(admin)和你刚才填写的密码,去体验一下 SugarCRM,图 6-22 是 SugarCRM 的管理后台,功能十分强大。



图 6-22

## 5. 安装中文语言包

对于英文不好的用户,你可以选择安装简体中文语言包,请按照以下步骤来做:

- 1) 下载简体中文语言包。通过官方网站:<http://www.srgarcrm.com>。
- 2) 拷贝语言包中的 modules 和 include 目录到 SugarCRM 目录/var/www/crm 下。
- 3) 修改配置文件 config.php:

```
'default_charset' => 'iso8859-1', 编码改为 UTF-8
'languages' =>
array(
    'en_us' => 'US English',
    'zh_ch' => '简体中文', //增加一条: 'zh_ch' => '简体中文',
)
```

中文界面,对于中国用户来说,看上去比较熟悉。

## 6. 如何使用 SugarCRM

SugarCRM 的使用是比较容易的,你可以快速建立自己的客户数据,销售情况。现在简单介绍一下,常用功能的使用方法:

- ◆ 新增用户:点击右上角的个人信息,处于左侧一栏的快捷方式中看到新增用户了吧。
- ◆ 系统管理:点击右上角的系统管理,在这里,你可以做很多的工作,比如数据库的备份、系统环境的设置、角色管理等等。不要忘了将系统定期进行数据库备份,以避免出现意外时数据库遭破坏而带来严重的损失。
- ◆ 日程安排:你可以安排电话、会议、任务日程。分别点击相应的菜单栏目。
- ◆ 客户管理:点击客户档案。你可以自己输入客户资料,也可以从外部文件导入(不过这个外部文件是要具有一定格式的)来新增客户。

## 第4节 架设 CMS 系统

### 1. CMS 概况

内容管理系统(Content Management System,简称 CMS),组织和协助共同合作的内容的结果,是指用于管理及方便数字内容的系统。它是支撑电子政务、电信综合门户、企业信息门户、知识管理平台、电子商务平台的基础性软件系统。内容管理系统可以帮助政府、企业或组织灵活、准确、高效、智能地管理信息内容,实现信息的采集、加工、审核、发布、存储、检索、统计、分析、反馈等整个信息生命周期的管理。

## 2. 开始部署 CMS

互联网上出现了很多的 CMS 系统,在功能上也都各有不同。如何去选择适合你自己的 CMS 系统呢?推荐大家使用开源的 CMS 系统,基于 PHP 和 MySQL 的开源 CMS 系统就有上百种,其中 XOOPS、Mambo、PHP-Nuke、Drupal、PostNuke 都是开源 CMS 中的典型代表。这些系统完全可以满足中小企业信息化,企业网站建设的需求。本节将以 Mambo 为例,介绍如何基于 Mambo 搭建企业网站门户。

### ● Mambo 简介

Mambo 中文意思为曼波音乐,在 2004 年 4 月 20 日伦敦举行的 Linux 用户和开发员大会上,Mambo 从众多优秀的开放源码系统中脱颖而出,获得 2004 年度最佳 Linux 开放源码系统奖项,击败和它同场竞技的 kde、firebird sql 以及 egrouppware 等知名度很高的系统。Mambo 是功能非常强大的智能建站系统,结合了 Php-Nuke、XOOPS 的很多优点,使系统更灵活,更强大,还有非常多的插件和模板下载。

## 3. 获得 Mambo

Mambo 发展到现在,版本已经非常多,网络上各种各样的基于 Mambo 系统开发的 CMS 也非常多,我们选择 MamboV4.6 版本。你可以通过:<http://tech.lupaworld.com/lamp/>下载到 MamboV4.6,下载后得到 MamboV4.6RC2.tar.gz 文件,假如文件在/root 目录下,现在开始解压并安装。

```
[root@lupa root]# tar -zxvf MamboV4.6RC2.tar.gz //解压;  
[root@lupa root]# mv MamboV4.6RC2 mambo //重命名;
```

把文件全部移动到可执行环境下,如:/usr/local/apache/htdocs/。

```
[root@lupa root]# mv mambo/usr/local/apache/htdocs/ //把 mambo 移到目标目录下;
```

## 4. Mambo 安装与配置

Mambo 的安装还是比较轻松的,基本上按照系统安装脚本的提示,就可以完成整个系统的安装。现在打开 Web 浏览器,输入:<http://localhost/mambo>。

### 1) 安装前的系统检查

在安装 Mambo 之前,安装脚本文件会对系统平台进行检查,这里有三部分:

- ◆ 检查服务器是否可以运行 Mambo 系统。
- ◆ 一些 PHP 的环境设置。
- ◆ 检测文件和目录的权限。

如果一切都检查好,按 Next。注意如果有些项目显示红色,比如在检测文件和目录权限时,你没有对 Mambo 的相关目录和文件设置权限,它会以红色文字显示。你需要修改每个目录的权限,直到全部为绿色为止。如图 6-23。

修改目录权限的命令为:

```
[root@ lupa mambo]#chmod -R a + w administrator/backups administrator/components
```

这里省略了一些目录,你得把上面标有红色粗体标识字的目录名全都填上去。

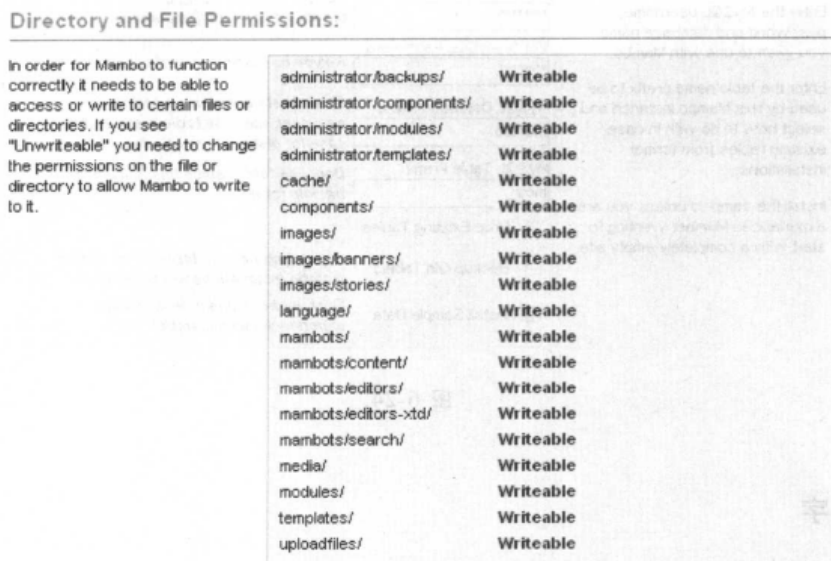


图 6-23

## 2) 许可协议

接受许可协议,你如果使用这个软件,必须接受 GPL 许可协议,因为 Mambo 是基于 GPL 发布的,在同意前打钩,进入下一步。

## 3) 数据库配置

数据库的配置。填写如下数据,其他都默认,如图 6-24,进入下一步。

- Host Name:localhost
- MySQL User Name:lupacms
- MySQL Password:password
- Verify MySQL Password:password
- MySQL Database Name:mambo

数据库名和用户,你可以通过 phpMyAdmin 和 MySQL 客户端来创建。如何创建数据库和用户,请参考本章的第一节架设 BLOB 平台的内容。



Next >>

## step 1

### MySQL database configuration:

Setting up Mambo to run on your server involves 4 simple steps...

Please enter the hostname of the server Mambo is to be installed on.

Enter the MySQL username, password and database name you wish to use with Mambo.

Enter the table name prefix to be used by this Mambo instance and select how to do with in case existing tables from former installations.

Install the samples unless you are experienced Mamboer wanting to start with a completely empty site.

Host Name	<input type="text" value="localhost"/>	<i>This is usually "localhost"</i>
MySQL User Name	<input type="text" value="supacms"/>	<i>Either something as "root" or a username given by the hoster</i>
MySQL Password	<input type="password" value="password"/>	<i>For site security using a password for the mysql account is mandatory</i>
Verify MySQL Password	<input type="password" value="password"/>	<i>Retype password for verification</i>
MySQL Database Name	<input type="text" value="mambo"/>	<i>Some hosts allow only a certain DB name per site. Use table prefix in this case for distinct mambo sites.</i>
MySQL Table Prefix	<input type="text" value="mos_"/>	<i>Don't use "old_" since this is used for backup tables</i>
<input type="checkbox"/> Drop Existing Tables		
<input type="checkbox"/> Backup Old Tables <i>Any existing backup tables from former mambo installations will be replaced</i>		
<input checked="" type="checkbox"/> Install Sample Data <i>Don't uncheck this unless you are experienced with mambo!</i>		

图 6-24

#### 4) 输入站点名字

输入站点名称, 比如 LUPA, 如图 6-25, 进入下一步。

Next >>

## step 2

### Enter the name of your Mambo site:

**SUCCESS!**

Type in the name for your Mambo site. This name is used in email messages so make it something meaningful.

Site name	<input type="text" value="LUPA"/>	<i>e.g. The Home of Mambo</i>
-----------	-----------------------------------	-------------------------------

图 6-25

#### 5) 站点设置

设置站点的网址、绝对路径、权限, 一般情况下就默认。填上管理员的 Email, 如果系统给出的密码太复杂的话, 就改个自己熟悉一点的。进入下一步。

#### 6) 安装完成

Mambo 的安装终于要结束了, 看到提示信息了吧, “为了安全起见, 请删除 installation 目录, 包括其中的所有文件和子目录”(不做这一步, 你还进不了 Mambo 系统)。如图 6-26, 点击 View Site 进入首页, 点击 Administrator 进入管理后台。

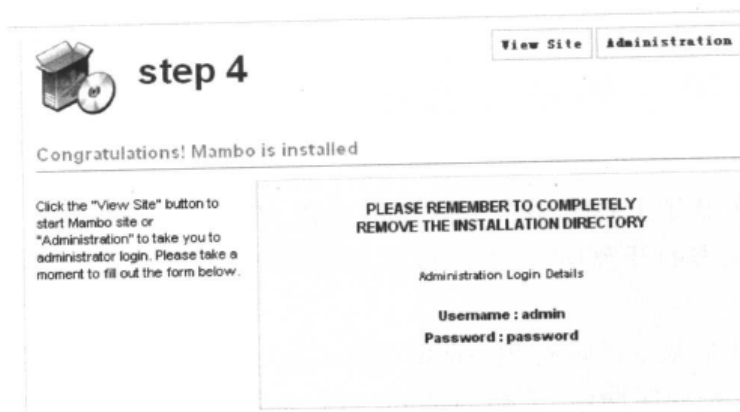


图 6-26

## 7) 进入首页

现在进入首页,如图 6-27。Mambo 的首页采用模版机制,也就是说,只要你懂得如果制作网页,而对 PHP 不懂也可以设计出非常漂亮的主页出来。Mambo 提供了非常多的模版下载。

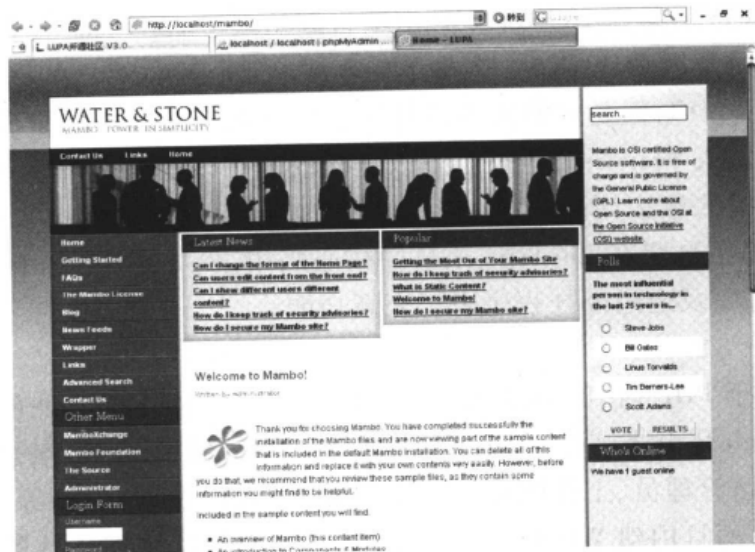


图 6-27

## 5. 如何使用 Mambo

- ◆ 系统管理后台的登录: <http://localhost/mambo/administrator>。
- ◆ 支持中文: 点击菜单 [ site ] -> [ Language manager ] -> [ new ] 选择 Chinese 这样系统就可以支持中文。你发的文章就不会出现乱码,注意系统后台界面还是英文,如果你需要中文,你可以去下载中文语言包,然后安装上去。
- ◆ 新增文章: 点击菜单 [ content ] -> [ all content items ] -> [ new ]。
- ◆ 新增用户: 点击菜单 [ site ] -> [ user manager ]。

Mambo 的使用方法非常简单,一般你熟悉了系统后,马上就可以快速建立自己的网站。

## 第 5 节 LAMP 上的开源项目

本章详细介绍了基于 LAMP 平台上的实例配置,在开源社区中,优秀的项目还非常多,下面推荐几个比较优秀的开源项目,你可以通过本章所学到的知识,去调试你喜欢的系统。

### ● Xoops

Xoops 是一个可扩展、面向对象、简单易用的动态网站内容管理系统,基于 PHP + MySQL 开发。Xoops 是社区网站、公司内部门户、企业网站、weblogs 和更多网站的理想选择。

### ● PHPNUKE

PHPNUKE 是国外著名的开放式 CMS 网站系统,经过多年的完善,目前已经形成包括社区、论坛、日志、电子商务等多功能的大型软件,是国外主流的免费建站系统。

### ● LifeType

LifeType 是个非常优秀的多用户 BLOG 系统,基于 PHP + MySQL 开发,在 GPL 许可证下发布,它的目标是要创造一个稳定且支持多用户的 BLOG 平台,透过 BLOG 来强化网络社区的运作与交流。LifeType 是在 2003 年 2 月由项目创始人 Oscar Renalias 所建立的,他原来建立这个项目的目的只是因为他需要一个简单的环境能来管理他自己的网站。在那时候其实他并不知道什么是 BLOG,但是初期的项目其实已经具备有 BLOG 的雏形了。几个礼拜后,另一个共同项目创始人 Francesc,建议干脆以这样的基础来建立一个多用户的 BLOG 平台。看起来似乎是个不错的想法,所以几个月后就在 2003 年 9 月 2 日发布了 pLog 0.1 版。当初的一些功能,目前都还是留在 LifeType 项目的核心当中,例如多用户 BLOG 的管理、模版引擎、多国语言的支持以及透过外挂程序来扩充新功能。

### ● eGroupWare

eGroupWare 基于 PHP 语言开发的群组协同作业软件,采用 GPL 许可证发行,目前所包含的功能模块有 Web 化电子邮件、行事历、通讯簿、我的活页夹、记事本、项目管理、资源、手册、档案管理员、网站管理员、工作流程、WIKI、网站连接、投票、知识库等。eGroupWare 不仅具备工作流程引擎,而且内建 WIKI 百科全书功能,成为与商业版产品最大的区别与特色。目录 eGroupWare 也是群组软件中是最具代表的开源软件。

### ● phpGROUPWARE

phpGROUPWARE 以前称为 webdistro 是一个多用户群组软件,采用 PHP 开发,它提供了约 50 多个应用,包括日历、地址本、项目管理、TODO 名单、电子邮件、新闻、文件管理等应用。日历支持重复事件,包括警报功能,电子邮件的附加档案和图形系统支持理念。整个系统支持用户选择的主题,用户许可、多语言支持。

### ● osCommerce

osCommerce 是一个开放源码的电子商务系统,它具有丰富的网上购物的功能,让业主可以设置、管理、维护自己的网上商店。osCommerce 提供一个自由、开放的电子商务平台,包括强有力 PHP 语言编写和 MySQL 数据库服务器。

- Atutor

Atutor 是一个基于网络的开放源码学习内容管理系统(LCMS), 管理人可以在五分钟内安装或更新 Atutor 系统, 功能强大, 操作容易。你可以通过 Atutor 开发特色的课件, 并在此平台上进行教学, 作业, 考试等功能。

- Net office

Net office 是一个非常优秀的开源项目管理系统, 使用 Net office 可以在线管理项目、提供团队合作、用户管理、多访问级别、任务、项目、和时间跟踪、任务修改历史、文件批准跟踪、注释、客户项目站点、CRM、Gantt 图等等。

- phpBB

phpBB 是一个强大的、可升级、可高度自定义的开放源代码的社区论坛系统。phpBB 的操作界面友好, 管理员控制面板简单易懂。它基于 PHP 服务器语言和 MySQL 数据库服务器构建, phpBB 是论坛的理想选择。