第 2 章　取得并安装 Apache

本章内容：

- 到哪里去找最新的 Apache 源代码
- Apache 的系统需求
- 如何下载软件
- 如何编译源代码
- 如何在系统上安装可执行文件
- 在 Apache 中创建目录
- 紧跟 Apache 的发展

出于在第 1 章中所讨论的种种原因，我喜欢 Apache。但喜欢它的最重要原因是：任何人都可以自由得到它的源代码！这就给我们机会，可以浏览源代码，并看看这些功能是怎么实现的。如果愿意的话，甚至可以随意改变这些代码。然而，对那些不是 C 程序员，但仍需要功能强大的自由 Web 服务器的人来说，玩弄大量的 ANSI C 代码可不是受人欢迎的消遣。幸运的是，没有什么可以担心的——Apache 不仅带来了源代码，还带来了预先建立好的可执行软件包。在本章中，将讨论从源代码和预先建好的可执行程序来安装 Apache。

2.1　正式的 Apache 源代码

无论你何时得到了自由软件（源代码或可执行文件），确保不是从未知的、遥远的地方得到它。“未知的、遥远的地方”的含义在下面例子中可以得到更好的理解。譬如说，你想获得由 Sun Microsystems 开发的、自由的、基于 Java 的 Web 浏览器软件，不要从一个称为 troyzon.horse.getitnow.com.sg 的 FTP 站点得到它。你应该在 java.sun.com 站点去寻找它。你需要能够做某种鉴定检查，而这在 Internet 上常常是很困难的，因此，应该坚持访问那些被广泛使用的站点。例如，如果你为自己的 Windows 机器从 ftp.cdrom.com（一个有名的自由软件/共享软件源代码归档站点）得到一个实用程序，你不能确保 cdrom.com 已经全面检查了该软件任何隐藏的危险。如果有该软件问题的任何历史记录，那么，该软件可能就不会在那里了。你明白了吧。在任何时候如果有了怀疑，请在 USENET 上询问，或向适当的新闻组邮递关于你寻找的软件的正式位置的问题。

对我们来说很幸运，Apache 开发人员和支持人员设置了用来获得 Apache 软件的正式站点。该正式的 Apache Web 站点是：

```
http://www.apache.org
```

此站点包含了 Apache 的最新的稳定版、Apache 的最新发行版、补丁程序、捐献者提供的 Apache 模块以及诸如此类的内容。它是你要寻找自己所需的 Apache 全部内容的地方——你可能会被指引到一个镜像站点，该站点的地理位置离你比较近，以帮助减少带宽要求和网络拥挤。

2.2 系　统　需　求

在可以在自己的 Web 服务器上显示 Powered by Apache 标识（如图 2-1 所示）之前，你可能需要确保自己的 Web 服务器有足够的“能力”来运行它。

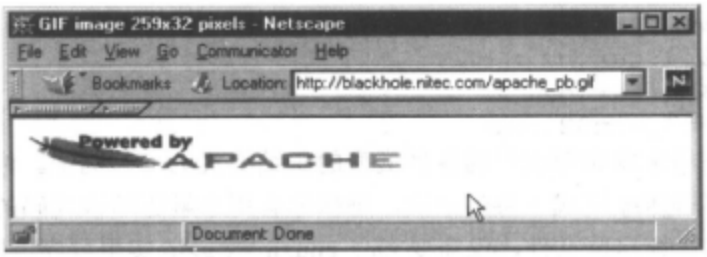


图 2-1　Powered by Apache 标识

运行 Apache 不需要太多的计算资源。它在有 6~10MB 硬盘空间和 8MB RAM 的 Linux 系统上运行得很好。然而，只运行 Apache 可能不是你想做的事情。更可能的是，你想运行 Apache 来提供 Web 服务、启动 CGI 进程以及充分利用所有 Web 能够提供的令人惊奇的功能。在这种情况下，你需要提供反映负载要求的额外的磁盘空间和内存空间。可以用两种方法来达到此目的：请教用 Apache 运行类似站点的人，了解他们正在使用何种类型的系统资源；或者，已经在系统中安装 Apache 之后，试验你自己的实际需要。

在后一种情况下，你可以使用系统实用程序，如 ps、top 以及诸如此类的东西，来显示 Apache 进程的内存消耗。然后，通过将单个进程的内存消耗乘以在高峰时间运行的 Apache 进程总数（请参见第 4 章的 MaxSpareServers 命令），就可以确定所需要的全部内存。这种方法会给出你的站点 Apache 内存需求的合理估算。请注意，如果计划在 Apache 服务器上运行几个 CGI 程序，你将不得不也为这些程序确定内存消耗，并将此额外的需要计算在内。

Apache 源代码文件和可执行文件的磁盘需求对大多数系统来说不会有问题，因为 Apache 的可执行文件只占 1MB 空间，而源代码文件所占空间大约是 5MB。但是，你应该对 Apache 创建的日志文件给予应有的注意，因为每个日志条目大约占去 80 字节的磁盘空间。如果你预计在一天中大约有 100000 次访问，那么你的 Apache 访问日志文件将占用 8000000 字节。

在第 11 章中，将在“日志维护”一节中讨论如何循环使用日志文件。

2.3 下 载 软 件

在第一次下载 Apache 软件之前，应该注意几件事情。可能你将发现可以得到两个 Apache 版本：一个是正式发行版，而另一个是有最新代码和功能的 beta 发行版。例如，如果看到一个 Apache 1.2.4 版，而另一个是 1.3b3 版，那么第一个就是正式发行版，而后者则是 beta 版。一个第三次发布的 beta 版，如 1.3b3（在它前面有 1.3b1 和 1.3b2），很可能是稳定的，但不建议将它用在生产的 Web 服务器中。要下载你需要的版本，只需要找到下载该软件的链接，或使用 www.apache.org/dist/来到达该站点的发布目录。在这里，你将在多个压缩软件包中找到该软件的发行版和 beta 版。例如：

```
apache_1.2.4.tar.Z
apache_1.2.4.tar.gz
apache_1.3b3.tar.gz
apache_1.3b3_win32.exe
```

上面列举的是用来发布源代码的各种类型的压缩格式的一些例子。你需要选择自己的系统可以处理的那种压缩格式（换句话说，确保你有用来解压缩该代码的实用程序）。在典型情况下，要解压缩这些文件，仅需要 tar、gnuzip 或 gzip 实用程序。例如，要在 Linux（RedHat 4.2）系统上解压缩 Apache 1.2.4.tar.gz 文件，使用下面的命令：

```
tar xvzf apache_1.2.4.tar.gz
```

或者，可以使用：

```
gzip  -d   apache_1.2.4.tar.gz
tar   xvf  apache_1.2.4.tar
```

它在保持每个文件相对路径不变的同时，解压缩并提取子目录中的所有文件。

Note

在 Apache 的 Windows NT/95 版中，可能有一些自解压压缩文件。对这样的文件可以通过简单运行下载的文件，就可以实现解压缩和提取。对于针对 Windows 95/NT 的安装和配置的详细内容，可跳过本章的其余部分，直接阅读第 20 章。

可执行文件通常被保存在不同的目录中，每个操作系统在那里都有自己的子目录。注意，如果你的操作系统没有出现在可执行文件的目录中，并不一定表示该操作系统不被支持。它只表示在 Apache 开发组或捐献组中尚未有人为你的系统编译可执行文件。你很可能会发现用于以下系统的可执行文件：

- AIX
- FreeBSD
- HPUX
- IRIX
- Linux NetBSD
- OS2，Solaris

♦SunOS

♦Ultrix

如果你决定下载预先建立的 Apache 可执行版，你可以跳过以下各节，并直接阅读在本章后面的“使用下载的可执行文件”一节。

2.4　为系统配置源代码

在这一节中，讨论如何为 Linux 系统编译 Apache。从本质上讲，这些步骤应该对几乎所有的 UNIX 系统都相同。

一旦在相应的目录中提取了源代码，你将看到 src 子目录，在其中保存了所有的 Apache 源代码。请按照着下面这些步骤进行下去：

1. 阅读在发行版的主目录（也就是 src 目录上一层的目录）中的 README 文件。
2. 将目录变为 Apache 源代码发行版的 src 目录。
3. 阅读名为 INSTALL 的文本文件。注意有关自己系统的任何特点。
4. 在相同目录中，将配置样本文件（Configuration.tmpl）复制为 Configuration 文件。

现在，你可以使用文本编辑程序（如 vi）来看在 Configuration 文件中的配置选项。

2.4.1　配置选项

在 Configuration 文件中可以得到五类信息。如下所示：

♦注释——以#（hash）符号作为第一个字符的行。

♦Makefile 选项——诸如 CC=gcc、EXTRA_CFLAGS 这样的的行。

♦Rules 选项——以关键字 Rules 开头的行。

♦AddModule 选项——以 AddModule 开头的行。

♦可选的 module 选项——以%Module 开头的行。

对于典型的安装，不需要改变这些行中的任一行。但是，如果确实要改变它们，确保你明白它们的用途。

2.4.1.1　Makefile 配置选项

这些选项行使你能够指定 Makefile 中的各种元素，此 Makefile 将在后面由 Configure 脚本来创建。对于大多数系统，不需要修改这些元素中的任一种。Configure 脚本会尝试决定在系统上使用哪种 C 编译软件。万一由于某种原因你认为它可能会找不到编译软件，通过在下面的行中给出编译软件可执行文件的名称，告诉 Configure 脚本编译软件是什么：

```
CC=
```

例如：

```
CC=gcc
```

现在，如果需要将额外的标志（参数）指定给 C 编译软件以编译代码，可以使用下面各行：

```
EXTRA_CFLAGS=
```

```
EXTRA_LFLAGS=
```

如果系统需要特殊的库和包含文件，可以在这里指定它们：

```
EXTRA_LIBS=
EXTRA_INCLUDES=
```


注意，Configure 脚本自动将代码优化设置为-O2。如果你想要一个不同的设置，请取消对下面行的注释：

```
#OPTIM=-O2
```

然后，将该值变为你想要的任何值，而同时你的 C 编译软件必须要支持该值。对于大多数安装，缺省设置就工作得很好了，就如在 RedHat Linux 系统上所做的一样。

2.4.1.2 Rules 配置

下一步，你必须告诉 Configure 决定需要什么功能。配置选项如下所示：

```
♦Rule  STATUS=yes
♦Rule  SOCKS4=no
♦Rule  WANTHSREGEX=default
♦Rule  IRIXNIS=no
```

第一条规则指定你在服务器中需要 STATUS 功能。这意味着你想在 Apache 中使用 status_module，而后者使服务器能够显示与性能相关的信息。要使此规则生效，需要确保在模块配置区域中包含了 status_module。

在缺省情况下 SOCKS4 功能是关闭的。对于那些不知道 SOCKS 是什么的人来说，它是一个控制系统，在那里，所有的 TCP/IP 网络应用程序数据流过 SOCKS 守护进程。这使 SOCKS 能够收集、审查、屏蔽、过滤以及控制网络数据。大多数人都将它用作一个基于软件的防火墙。如果想要使 Apache 使用 SOCKS 功能，需要将它设置为 Yes 来打开此项功能。还要确保修改在 Makefile 配置区域中的 EXTRA_LIBS 设置，以指向你的 SOCKS4 库文件。否则，Configure 脚本会将它设置为-L/usr/local/lib -lsocks。

WANTHSREGEX 选项被自动设置为缺省值。这就表明你想使用 Apache 提供的规则表达式软件包。然而，如果你宁愿使用自己系统的规则表达式软件包，可以将此选项设置为 No。

IRIXNIS 选项是为那些想要在运行 IRIX 和 NIS 的 Silicon Graphics 系统上使用 Apache 的人们准备的。

2.4.1.3 模块配置

模块是 Apache 的组成部分，它为 Apache 内核增加新功能。通过使用模块配置，你可以定义在 Apache 服务器中需要什么功能。模块配置行看起来和下面的相似：

```
AddModule modules/standard/mod_env.o
```

此行将在 src/modules/standard 子目录下找到的 mod_env 模块添加到 Apache 中。

Note

旧的 Apache 版本（1.2x 或以下的版本）使用下面格式的模块行：

```
Module  module_name  module_object_file
```

所提供的缺省模块被列举在列表 2-1 中。对于 Apache 的典型安装，这些模块应该足

够了。应该注意，每个模块都在最终的 Apache 可执行程序（httpd）上增加了更多的内存和磁盘空间需求，因此，应该仔细检查该列表，是否有哪个模块可以从缺省列表中删除。如果能删除模块， Apache 服务器的负载将更轻，并且可能会快一些。另一方面，如果发现一个模块有用，但却被注释掉了，就可以移去注释字符（#），以使该模块能加在 Apache 中。类似的，要禁止一个模块，只需要在该模块的指定行的开始插入一个注释字符。

列表 2-1：在 Configuration 文件中的缺省模块列表

```
AddModule modules/standard/mod_env.o
AddModule modules/standard/mod_log_config.o
AddModule modules/standard/mod_mime.o
AddModule modules/standard/mod_negotiation.o
AddModule modules/standard/mod_include.o
AddModule modules/standard/mod_autoindex.o
AddModule modules/standard/mod_dir.o
AddModule modules/standard/mod_cgi.o
AddModule modules/standard/mod_asis.o
AddModule modules/standard/mod_imap.o
AddModule modules/standard/mod_actions.o
AddModule modules/standard/mod_userdir.o
AddModule modules/standard/mod_alias.o
AddModule modules/standard/mod_access.o
AddModule modules/standard/mod_auth.o
AddModule modules/standard/mod_setenvif.o
```

如果选择将 STATUS 规则设置为 Yes，就需要取消对 status_module 行的注释，以使它看起来像这样：

```
AddModule modules/standard/mod_status.o
```

如果将 STATUS 规则设置为 Yes，但没有取消对 status_module 行的注释，该规则将被忽略。现在你就准备好运行 Configure 脚本了。

2.4.1.4　运行 Configure

要运行 Configure 脚本，需要处在 src 目录下（如果你正在按照说明来做，现在应该已经是在该目录下）。只需要在 shell 命令解释程序的提示符后输入下面的内容：

```
./Configure
```

Configure 运行并创建 Makefile。现在你已准备好编译源文件了。

2.4.1.5　编译 Apache

你需要运行 make 实用程序来编译源文件。在 shell 命令解释程序提示符后输入：

```
make
```

make 实用程序将使用由 Configure 脚本创建的 Makefile，并运行 C 编译软件来创建 Apache 可执行程序（httpd）。

编译 Apache 所需要的时间长短取决于你的计算机有多快以及资源有多丰富。一个强有力的 CPU 和大量的 RAM 对此会有所帮助。如果 make 实用程序由于某种原因编译失

败，请仔细检查错误消息，看看你是否可以从中得到帮助。在多数情况下错误是由下列原因引起的：缺少系统库文件，或者你的编译软件需要更多的标志。在每种情况下，你都将必须改变在 Configuration 文件中的 Makefile 选项（除非你想要修改由 Configure 生成的 Makefile 本身），并随后再次运行 Configure（如果你没有自己修改 Makefile）和 make 实用程序。如果问题还存在，请找比你更了解你的操作系统和 C 编译软件的人，并向他们请求帮助。

如果没有选择余地，请向适当的新闻组（参见附录 D）邮递消息来解释你的问题，要包括全部错误消息。

如果所有事情都正常，现在应该在 src 目录下有 **httpd** 可执行程序。如果你一直在屏住呼吸，现在可以放松一下了！

2.4.2 测试新建立的 Apache

现在是确保你的可执行程序真的“可执行”的时候了。首先，需要检查目录列表，以显示权限设置。例如：

```
ls -l httpd
```

在 src 目录下敲入此行将显示该文件的权限。请确保 **httpd** 是可读并且是可执行的。敲入下面的内容来检查你的 Apache 版本：

```
./httpd -v
```

这将打印出你的 Apache 服务器版本号。如果该版本匹配你下载的版本号，你就可以工作了！祝贺你！

现在，你可能还想编译支持实用程序。Apache 带来一些实用程序，它们在管理程序本身可能会有用。通过运行下面的命令，你可以从 support 目录中编译这些实用程序：

```
make all
```

这就编译了所有的实用程序，并创建可执行文件，如 **htpasswd**、**htdigest**、**httpd_monitor**、**rotatelogs**、**logresolve** 等等。注意，**suexec** 程序不被前面的命令编译。你将在第 12 章中学习该程序。

现在可以跳过下面这一节，直接阅读“创建 Apache 目录”。

2.5 使用下载的可执行文件

自己编译 Apache 的源代码，而不使用其他人的可执行程序是个好主意。当你使用下载的可执行文件时，你是在让其他人（可能是你不了解的人）来决定你的 Web 服务器使用哪些模块和功能。有时候，下载的可执行文件可能在你的系统中根本就不工作，因为在该可执行文件所需要的库文件和你的系统中的库文件之间可能存在不兼容性。如果使用可执行文件是可接受的，或者如果你不选择编译源代码，那么，确保下载可执行文件的站点是可信的。你可以从 www.apache.org 或它的镜像站点得到很多专用于常用操作系统的可执行文件。

请在一个目录中解压缩下载的可执行文件。找到 **httpd** 可执行文件，并将该文件的权

限设置为允许运行。请使用下面的命令来得到版本号。

```
./httpd -v
```

如果此版本与你想下载的版本相匹配，你就准备好为你的 Apache 服务器创建目录结构了。请继续阅读下一节。

Tip　RedHat Linux 的用户可以在 ftp://ftp.redhat.com/下载 Apache 的 RPM 版，以简化 Apache 的安装过程。

2.6　创建 Apache 目录

可以将 Apache 可执行程序保存在你愿意的任何地方，但是，从开始就组织好一切可能是个好主意。下面是用来做到这一步的具体建议：

在一个方便的位置下，如/usr/local 或/usr/home，创建名为 Apache 的目录。称此目录为%ApacheRoot%。请创建下面的目录。

```
%ApacheRoot%/bin
%ApacheRoot%/conf
%ApacheRoot%/logs
```

将 httpd 可执行程序和所有编译好的实用程序复制到%ApacheRoot%/bin 目录下。将样本配置文件（在 conf 目录下）复制到%ApacheRoot%/conf 目录下。

为 Apache 创建一个新用户和组是个好主意。这将使你能更好地控制哪些磁盘区域是 Apache 可以访问的。为此创建了用户 httpd 和组 httpd。现在，你可以为各个目录和可执行程序设置权限。

除了 root 用户，你不应该允许任何人访问这些可执行程序和配置文件。通过使用 chown 命令，可以只允许 root 用户和在 root 组中的用户访问这些文件。这些命令是：

```
chown -R root.root %ApacheRoot%/bin
chown -R root.root %ApacheRoot%/conf
```

还有，日志（logs）目录只应该由 Apache 用户来读取。你可以再次使用 chown 命令来为它设置权限，如下所示：

```
chown -R httpd.httpd %ApacheRoot%/logs
```

所有这些 chown 命令已经设置了正确的所有权，但是，你还需要设置访问权限。如果需要 root 用户拥有 bin 和 conf 目录的全部权限（读，写和执行），并且需要 httpd 用户拥有 logs 目录的全部权限，你就可以使用下面的 chmod 命令完成所有工作：

```
chmod -R 700 %ApacheRoot%
```

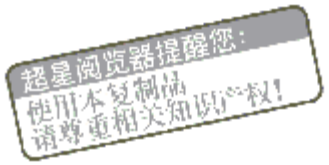
此命令设置的访问权限为只有每个目录的所有者才能访问此目录，这正是所需要的。

Caution　在%ApacheRoot%下的 logs 目录只能由运行 Apache 进程的用户帐号来读写，这是非常重要的。因为在单机模式下（将在第 3 章讨论这一概念），root 用户必须运行主 Apache 服务器进程，所以在这里建议的目录设置应该足够了。

第 3 章　配置并运行 Apache

本章内容：

- ◆Apache 服务器配置基础
- ◆启动、停止和重新启动 Apache 服务器
- ◆测试运行中的 Apache 服务器



在上一章中，讨论了如何在 UNIX 系统中编译和安装 Apache Web 服务器。然而，在运行该服务器之前，你需要编辑一些运行时的配置文件，以使服务器正常地运行。本章讨论基本的配置细节，以设置好你的服务器，并让它运行起来。

Apache 读取四个配置文件：httpd.conf、access.conf、srm.conf 和 mime.type。对于典型的站点，一般不需要改变最后一个文件。因此，在本章中就不对它作进一步的讨论。其余三个文件告诉 Apache 如何作为 Web 服务器来运转。通常在服务器启动过程中读取所有这些文件。

在本章中，将讨论如何配置每一个文件、运行服务器的不同方法，以及在启动和停止服务器时所涉及的问题。在本章末尾，还要说明如何测试服务器。

3.1　配置服务器

每个 Apache 源代码发行版都带一组样本配置文件。在标准的 Apache 源代码发行版中，你能找到名为 conf 的目录，它包含带有-dist 扩展名的样本配置文件。

在修改这些样本文件之前，你需要做的第一件事情是照下面那样复制这些文件：将

```
httpd.conf-dist    复制到  httpd.conf
access.conf-dist   复制到  access.conf
srm.conf-dist      复制到  srm.conf
```

虽然有三个文件，但它们都有相同的结构。这些文本文件包含两类信息：可选的注释行和服务器命令。行首带 # 符号的行是注释行。这些注释行对服务器软件来说没有意义，但为服务器管理员提供文档说明。你可以添加任意多的注释行。当服务器分析这些文件时，它忽略所有的注释行。

除了注释和空行之外，服务器认为所有其他行或者是完整命令，或者是命令的一部分。对服务器来说，directive 就是一条命令。它告诉服务器以特定的方式去完成一项具体任务。在编辑这些文件的时候，你需要作出如何使服务器运转的决策。在下面各节中，将讨论这些命令的含义，以及如何使用它们来定制服务器。