第 7 章　配置 Apache 服务器

本章内容：

7.1　WWW 服务器简介
7.2　Red Hat Linux 9 的默认配置
7.3　配置 Apache
7.4　访问控制、认证和授权
7.5　组织和管理站点内容
7.6　Apache 的日志管理和统计分析

学习目标：

- 熟悉 Apache 的特性
- 掌握 Apache 的安装和简单配置
- 掌握访问控制、认证和授权的配置
- 熟悉组织和管理站点内容的配置方法
- 掌握 Apache 的日志管理和统计分析方法

7.1 WWW 服务器简介

本节内容
Ø 选择使用 Apache Ø Apache 的特性 Ø Apache 2.0 的模块
学习目标
Ø 了解 Apache 的历史和现状 Ø 熟悉 Apache 的特性 Ø 了解 Apache 2.0 中的标准模块

7.1.1 选择使用 Apache

1．Web 服务器简介

Internet 上最热门的服务之一就是环球信息网 WWW（World Wide Web）服务，Web 已经成为很多人在网上查找、浏览信息的主要手段。WWW 是一种交互式图形界面的 Internet 服务，具有强大的信息连接功能。它使得成千上万的用户通过简单的图形界面就可以访问各个大学、组织、公司等机构和个人的最新信息和各种服务。

商业界很快看到了其价值，许多公司建立了主页，利用 Web 在网上发布消息，并将它作为各种服务的界面，如客户服务、特定产品和服务的详细说明、宣传广告以及日渐增长的产品销售和服务。商业用途促进了环球信息网络的迅速发展。

Web 服务具有如下特点：

Ø Web 是图形化的和易于导航的
Ø Web 是与平台无关的
Ø Web 是分布式的
Ø Web 是动态的
Ø Web 是交互的

Web 系统是客户/服务器式的。所以应该有服务器端程序和客户端程序两部分。常用的服务器是 Apache；常用的客户端程序是浏览器（如 IE、Netscape、Mozilla）。我们可以在浏览器的地址栏内输入统一资源定位地址（URL）来访问 Web 页面。Web 最基本的概念是超文本（Hypertext）。它使得文本不再是传统的书页式文本，而是可以在阅读过程中从一个页面位置跳转到另一个页面位置。用来书写 Web 页面的语言称为超文本标记语言，即：HTML。WWW 服务遵从 HTTP 协议，默认的 TCP/IP 端口是 80，客户与服务器的通信过程如图 7-1 所示。

整个通信流程简述如下：

图 7-1　Web 服务器与客户的通信过程

- Web 客户（浏览器）根据用户输入的 URL 连到相应的远端 WWW 服务器上。
- 从指定的服务器获得指定的 Web 文档。
- 断开与远端 WWW 服务器的连接。

也就是说，平时我们在浏览某个网站的时候是每取一个网页建立一次连接，读完后马上断开；当需要另一个网页时重新连接，周而复始。

2．最受欢迎的 Web 服务器

根据 Netcraft（http://news.netcraft.com）提供的最新调查资料，Apache Web 服务器是使用比例最高的 Web 服务器，如图 7-2 所示。

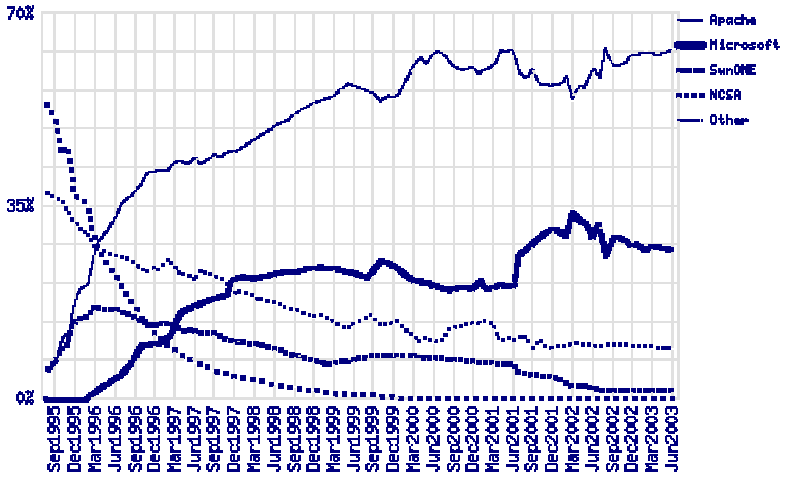

图 7-2　最受欢迎的 Web 服务器

在 2003 年 7 月，Netcraft 所调查的 40936076 个 Web 站点中，有 25856505 个 Web 站点使用 Apache 作为其 Web 服务器，占总数的 63.16 %，而使用 Microsoft IIS 的站点数为 10992195，只占 26.85 %。由此可见，Apache 是目前使用率最高的 Web 服务器。

3．Apache 的历史

Apache WWW Server 最初的源码和思想基于最流行的 HTTP 服务器——NCSA httpd 1.3，经过较为完整的代码重写，它如今已在功能、效率及速度方面居于领先的地位，Apache 项目成立的最初目的是为了解答公用 HTTP Server 发展中人们所关心的一些问题，例如如

何在现有的 HTTP 标准下提供更为安全、有效、易于扩展的服务器。

Apache 的开发人员全部为志愿者，而不含任何商业行为。其名称 Apache 意为 A Patchy Server，即它是基于现存的代码和一系列的 Patch 文件。

下面将 Apache 的简单发展历史列于表 7-1。

表 7-1　Apache 的发展

时　　间	事　　件
1995.3	Apache 0.6.2 版发行，这是第一个公开版本
1995.8	Apache 0.8.8 版发行，新增部分所包含的模块结构沿用至今
1995.10.1	Apache 1.0.0 版发行
1996.7	Apache 1.1 版发行。支持 HTTP1.1，基于名称的虚拟主机等
1997.6	Apache 1.2 版发行
1998.3	Apache 1.3 版发行
1998.6.12	mod_perl 1.0.0 版发行
2000	Apache 2.0 测试版发行
2002	Apache 2.0 发行

4．ASF

早期的 Apache 服务器由 Apache Group 来维护，直到 1999 年 6 月 Apache Group 在美国德拉瓦市成立了非盈利性组织的公司，即 Apache 软件基金会（Apache Software Foundation，ASF）。ASF 现在维护着包括 Apache 在内的多个项目，还包括 Perl、PHP、Java、Tcl、XML 等。ASF 的网址是 http://www.apache.org。

5．谁在使用 Apache

使用 Apache 的著名站点数不胜数，下面列出其中最知名的几个：

- Yahoo！
- IBM
- Amasom.com
- Hotmail.com
- Red Hat

7.1.2　Apache 的特性

1．Apache 1.3 的性能

选择 Web 服务器时，其功能和运行性能是最重要的因素。Apache 的众多特性保证了它可以高效而且稳定的运行。其性能主要表现在如下几个方面：

- 实现了动态共享对象(DSO)，允许在运行时动态装载功能模块。
- 采用预生成模式的技术提高响应速度。

- 可以运行在几乎所有计算机平台。
- 支持最新的 HTTP 1.1 协议。
- 简单而强有力的基于文件的配置。
- 支持虚拟主机。
- 支持 HTTP 认证。
- 集成了代理服务器。
- 具有可定制的服务器日志。
- 支持安全 Socket 层（SSL）。
- 用户会话过程的跟踪能力。
- 支持通用网关接口 CGI。
- 集成 Perl 脚本编程语言。
- 支持服务器端包含命令（SSI）。
- 支持 FastCGI。
- 支持 PHP。
- 支持 Java Servlets。
- 支持第三方软件开发商提供的大量功能模块。

2．Apache 2.0 的新特性

Apache 2.0 具备 Apache 1.3 的几乎所有特性。除此之外，Apache 2.0 添加了附加功能层，最基本的组件是可移植运行环境（Apache Portable Runtime，APR），它提高了 Apache 的跨平台性能。

另外，Apache 2.0 使用新的多处理模块（Multi-Processing Module，MPM），使用此模块会在服务器处理多个请求时，控制 Apache 的运行方式。Apache 中的 3 种运行方式分别是：

- 预派生（Profork）MPM：此模块在功能上兼容于 Apache 1.3 的运行模型。这种运行方式首先启动一个父进程，然后创建并启动一定（可配置）数量的子进程监听客户的请求。当监听到客户的服务请求后，子进程就响应此请求。重要的是父进程始终监控子进程，当没有足够的空闲子进程为客户服务时，父进程就会创建并运行新的子进程准备为客户提供服务；如果存在过多的空闲子进程，父进程就会依次终止这些空闲的子进程，直到服务器回到最大空闲子进程（可配置）数量之下。通过始终保持一定数量的空闲子进程来响应客户的请求，服务器可以避免在接收到客户请求时启动新进程的开销。
- 工作者（Worker）MPM：此模块是混合使用进程和线程的运行模型。这种运行方式首先启动一个父进程，然后创建并启动一定（可配置）数量的子进程，每个子进程都创建并启动相同数量的线程，由线程监听客户请求，而子进程并不监听客户请求。重要的是父进程始终监控子进程，当没有足够的空闲线程为客户服务时，父进程就会创建并运行新的子进程，并在子进程中创建与先前子进程创建的相同数量的线程准备为客户提供服务。这种运行方式是以牺牲可靠性和健壮性来换取可扩展性的。
- 独立子进程（Perchild）MPM：这是一种运行于类 UNIX 系统上的运行模式，它

也是混合使用进程和线程的运行模型。这种运行模式与工作者 MPM 类似，只是每个子进程创建的线程数量可以不一致，即每个子进程都可以创建指定数量（可配置）的线程。当服务器上负载增加后，Apache 不会创建新的子进程，而是在当前的子进程之一上创建新的线程为客户提供服务。这种运行方式具有最高的可扩展性，但却具有最低的可靠性。

7.1.3 Apache 2.0 的模块

与 Apache 1.3 类似，Apache 2.0 仍旧使用模块的方式运行。Apache 由内核、标准模块和第三方提供的模块 3 个层次组成。表 7.2 列出了 Apache 2.0 的标准模块。

表 7-2 Apache 2.0 的标准模块

模 块 名	说 明
Core	Apache HTTP 服务器核心模块
mpm_common	被 MPM 执行的一组指令
mpm_netware	专为 Novell NetWare 服务器优化的 MPM 模块
mpm_winnt	专为 Windows NT 优化的 MPM
Perchild	独立子进程（Perchild）运行方式的 MPM
Prefork	预派生（Profork）运行方式的 MPM
Worker	工作者（Worker）运行方式的 MPM
mod_access	提供基于主机名、IP 地址或者其他客户请求的访问控制
mod_actions	模块为基于媒体类型请求方式执行 CGI 脚本
mod_alias	提供文档树中主机文件系统各部分的映射和 URL 重定向
mod_asis	传送包含只有 HTTP 头的文件
mod_auth	使用文本文件的用户身份验证
mod_auth_anon	允许匿名用户访问身份验证
mod_auth_dbm	提供使用 DBM 数据库文件的用户身份验证
mod_auth_digest	使用 MD5 深层身份验证的用户身份验证
mod_autoindex	自动生成类似于 Unix 的 ls 命令或 Win32 dir shell 命令的目录索引
mod_cache	通向 URI 的内容 cache
mod_cern_meta	CERN httpd 原文件语意
mod_cgi	执行 CGI 脚本（用于进程方式的 MPM）
mod_cgid	执行 CGI 脚本（用于线程方式的 MPM）
mod_charset_lite	设定翻译和重编码的特别字符
mod_dav	实现分布式授权和版本发行（DAV）功能
mod_deflate	传送至客户端前进行内容压缩
mod_dir	提供用于“trailing slash”重定向和服务的目录索引文件
mod_echo	解释协议模块的简单映射服务器
mod_env	调整传送给 CGI 脚本和 SSI 页的环境
mod_example	解释 Apache 模块的 API
mod_expires	根据用户限定标准生成到期的 HTTP 头

（续）

模　块　名	说　　明
mod_ext_filter	在传达给客户之前通过外部程序发出回应体
mod_file_cache	在内存中缓存一个文件静态列表
mod_headers	HTTP 请求和回应头的个性化处理
mod_imap	服务器端镜像处理
mod_include	支持 SSI
mod_info	生成服务器配置信息
mod_isapi	Apache 中为 Windows 提供的 ISAPI 扩展
mod_log_config	记录发向服务器的请求日志
mod_mime	联合被请求文件扩展名和文件行为（处理和筛选）的内容（mime 类型，语言，字符集和编码）
mod_mime_magic	通过查看文件内容的几个字节确定 MIME 类型
mod_negotiation	提供内容协商
mod_proxy	支持 HTTP/1.1 协议的代理/网关服务器
mod_rewrite	提供 URL 请求的复杂重定向功能
mod_setenvif	允许基于请求类型的环境变量设置
mod_so	在启动或重启时提供可执行编码和模块的启动
mod_speling	试图更正因用户忽略大小写或一处错误拼写而引起的错误 URL
mod_ssl	使用 SSL 和 TLS 的密码技术
mod_status	提供服务器运行性能信息
mod_suexec	允许作为特殊用户或组运行 CGI 脚本
mod_unique_id	为每个请求提供具有单一身份的环境变量
mod_userdir	设置基于每个用户的站点目录
mod_usertrack	跟踪用户在访问一个站点时的行为，记入日志
mod_vhost_alias	提供大量虚拟主机的动态配置

除了标准模块之外，还可以找到许多第三方模块。用户可以连接如下网址查看有关第三方模块的信息：

http://modules.apache.org

7.2　Red Hat Linux 9 的默认配置

本节内容
Ø　安装和启动 Apache Ø　查看 Red Hat Linux 9 的默认配置
学习目标
Ø　掌握 Apache 的安装和启动方法 Ø　熟悉 Apache 的默认配置

7.2.1 安装和启动 Apache

1．从 RPM 安装 Apache

Red Hat Linux 9 自带了 Apache 2.0，有如下两个：

- httpd：Apache 2.0
- httpd-manual：Apache 2.0 手册

下面以 RPM 包的安装为例介绍 Apache 的安装。若用户在安装 Red Hat 时已经安装了 Apache 服务器，则可跳过下面的安装步骤。

操作步骤 7.1　安装 Apache 2.0

```
//查看是否安装了 Apache
# rpm -qa|grep httpd
//将 Red Hat Linux 9 的第 1 张安装光盘放入光驱后挂装
# mount /mnt/cdrom
//进入光盘的 Red Hat/RPMS 目录
# cd /mnt/cdrom/Red Hat/RPMS
//安装所需的 RPM 包
# rpm –ivh httpd-2.0.40-21.i386.rpm
# rpm -ivh httpd-manual-2.0.40-21.i386.rpm
//弹出光盘
# cd;eject
```

2．启动 Apache

安装完 Apache 后，下一步就是启动了。Red Hat Linux 9 默认 Apache 以独立运行方式启动，所以需要执行如下的操作步骤。

操作步骤 7.2　启动并检验 Apache 是否被运行

```
// 立即启动
# service httpd start
//下面的操作用于检验 httpd 是否被启动
# pstree|grep httpd
    |- httpd---8*[httpd]
//表示已经启动
#
//可以使用下面的命令检测配置文件语法的正确性
# apachectl configtest
httpd: Could not determine the server's fully qualified domain name,
using 127.0.0.1 for ServerName
Syntax OK
#
//可以使用如下命令查看运行状态
# service httpd status
```

```
httpd (pid 2759 2758 2757 2756 2755 2754 2753 2752 2749) 正在运行...
#
```

注意

（1）若希望 httpd 在下次计算机启动时自动启动，请使用命令:

ntsysv

选中 httpd。

（2）检测配置文件语法时，还可以使用下面的命令:

httpd -t

7.2.2　查看 Red Hat Linux 9 的默认配置

1．查看 Red Hat 发布的 Apache 2.0 的相关信息

执行如下的步骤可以查看 Red Hat 发布的 Apache 2.0 的一些相关信息。

操作步骤 7.3　查看 Red Hat 发布的 Apache 2.0 的一些相关信息

```
//查看编译配置参数
# apachectl -V
Server version: Apache/2.0.40
Server built:   Feb 25 2003 05:01:56
Server's Module Magic Number: 20020628:0
Architecture:   32-bit
Server compiled with....
 -D APACHE_MPM_DIR="server/mpm/prefork"
 -D APR_HAS_SENDFILE
 -D APR_HAS_MMAP
 -D APR_HAVE_IPV6
 -D APR_USE_SYSVSEM_SERIALIZE
 -D APR_USE_PTHREAD_SERIALIZE
 -D SINGLE_LISTEN_UNSERIALIZED_ACCEPT
 -D APR_HAS_OTHER_CHILD
 -D AP_HAVE_RELIABLE_PIPED_LOGS
 -D HTTPD_ROOT="/etc/httpd"
 -D SUEXEC_BIN="/usr/sbin/suexec"
 -D DEFAULT_PIDLOG="logs/httpd.pid"
 -D DEFAULT_SCOREBOARD="logs/apache_runtime_status"
 -D DEFAULT_LOCKFILE="logs/accept.lock"
 -D DEFAULT_ERRORLOG="logs/error_log"
 -D AP_TYPES_CONFIG_FILE="conf/mime.types"
 -D SERVER_CONFIG_FILE="conf/httpd.conf"
#
//下面查看已经被编译的模块
# apachectl -l
Compiled in modules:
```

```
  core.c
  prefork.c
  http_core.c
  mod_so.c
#
```


重点

（1）被编译的模块中包含 mod_so.c，表示当前的 Apache 支持 Dynamic Shared Objects（DSO），即用户可以在不重新编译 Apache 的情况下使用 APache eXtenSion（apxs）编译 Apache 的第三方模块。

（2）被编译的模块中包含 prefork.c 表示 Red Hat 发布的 Apache 是使用预派生（Profork）MPM 模式运行的。

2．Red Hat Linux 9 中的配置文件

由查看的编译参数可知，在 Red Hat Linux 9 中 httpd 的配置文件是：/etc/httpd/conf/httpd.conf

执行下面的操作步骤查看 Apache 在 Red Hat Linux 9 中的默认配置。

操作步骤 7.4　查看 Apache 在 Red Hat 9 中的默认配置

```
//查看配置文件
# grep -v "#" /etc/httpd/conf/httpd.conf
//当服务器响应主机头（header）信息时显示 Apache 的版本和操作系统名称
ServerTokens OS
//设置服务器的根目录
ServerRoot "/etc/httpd"
//设置运行 Apache 时使用的 PidFile 的路径
PidFile run/httpd.pid
//若 300 秒后没有收到或送出任何数据就切断该连接
Timeout 300
//不使用保持连接的功能，即客户一次请求连接只能响应一个文件
/建议用户将此参数的值设置为 On，即允许使用保持连接的功能
KeepAlive Off
//在使用保持连接功能时，设置客户一次请求连接能响应文件的最大上限
MaxKeepAliveRequests 100
//在使用保持连接功能时，两个相邻的连接的时间间隔超过 15 秒，就切断连接
KeepAliveTimeout 15

//设置使用 prefork MPM 运行方式的参数，此运行方式是 Red Hat 默认的方式
<IfModule prefork.c>
//设置服务器启动时运行的进程数
StartServers       8
//Apache 在运行时会根据负载的轻重自动调整空闲子进程的数目，
//若存在低于 5 个空闲子进程，就创建一个新的子进程准备为客户提供服务
MinSpareServers    5
//若存在高于 20 个空闲子进程，就创建逐一删除子进程来提高系统性能
```

```
MaxSpareServers   20
//限制同一时间的连接数不能超过 150
MaxClients      150
//限制每个子进程在结束处理请求之前能处理的连接请求为 1000
MaxRequestsPerChild  1000
</IfModule>

//设置使用 worker MPM 运行方式的参数
<IfModule worker.c>
……
</IfModule>

//设置使用 perchild MPM 运行方式的参数
<IfModule perchild.c>
……
</IfModule>

//设置服务器的监听端口
Listen 80
//将/etc/httpd/conf.d 目录下的所有以 conf 结尾的配置文件包含进来
Include conf.d/*.conf

//动态加载模块(DSO)
LoadModule access_module modules/mod_access.so
LoadModule auth_module modules/mod_auth.so
……
LoadModule proxy_connect_module modules/mod_proxy_connect.so

//当使用内置模块 prefork.c 时动态加载 cgi_module
<IfModule prefork.c>
LoadModule cgi_module modules/mod_cgi.so
</IfModule>
//当使用内置模块 worker.c 时动态加载 cgid_module
<IfModule worker.c>
LoadModule cgid_module modules/mod_cgid.so
</IfModule>

//设置运行 Apache 服务器的用户和组
User apache
Group apache

//设置 Apache 服务器管理员的 E-mail 地址
ServerAdmin root@localhost

//关闭此选项，当 Apache 服务器需要指向本身的连接时使用
//ServerName:Port 作为主机名，例如 www.jamond.net:80
//若打开此选项将使用 www.jamond.net port 80 作为主机名
```

```
UseCanonicalName Off

//设置根文档路径
DocumentRoot "/var/www/html"

//设置 Apache 服务器根的访问权限
<Directory />
//允许符号链接跟随，访问不在本目录下的文件
Options FollowSymLinks
//禁止读取.htaccess 配置文件的内容
   AllowOverride None
</Directory>

//设置根文档目录的访问权限
<Directory "/var/www/html">
//Indexes: 当在目录中找不到 DirectoryIndex 列表中指定的文件
//          就生成当前目录的文件列表
//FollowSymLinks: 允许符号链接跟随，访问不在本目录下的文件
   Options Indexes FollowSymLinks
//禁止读取.htaccess 配置文件的内容
AllowOverride None
//指定先执行 Allow（允许）访问规则，再执行 Deny（拒绝）访问规则
Order allow,deny
//设置 Allow（允许）访问规则，允许所有连接
   Allow from all
</Directory>

//对 Apache 服务器根的访问不生成目录列表，同时指定错误输出页面
<LocationMatch "^/$>
   Options -Indexes
   ErrorDocument 403 /error/noindex.html
</LocationMatch>

//不允许每用户的服务器配置
<IfModule mod_userdir.c>
   UserDir disable
</IfModule>
//当访问服务器时，依次查找页面 index.html、index.html.var
DirectoryIndex index.html index.html.var

//指定保护目录配置文件的名称
AccessFileName .htaccess

//拒绝访问以.ht 开头的文件，即保证.htaccess 不被访问
<Files ~ "^\.ht">
   Order allow,deny
   Deny from all
```

```
</Files>

//指定负责处理 MIME 对应格式的配置文件的存放位置
TypesConfig /etc/mime.types
//指定默认的 MIME 文件类型为纯文本或 HTML 文件
DefaultType text/plain

//当 mod_mime_magic.c 模块被加载时，指定 Magic 信息码配置文件的存放位置
<IfModule mod_mime_magic.c>
    MIMEMagicFile conf/magic
</IfModule>

//只记录连接 Apache 服务器的 IP 地址，而不记录主机名
HostnameLookups Off
//指定错误日志存放位置
ErrorLog logs/error_log
//指定记录的错误信息的详细等级为 warn 级别
LogLevel warn
//定义四种记录日志的格式
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
//指定访问日志的记录格式为 combined（混合型），并指定访问日志存放位置
CustomLog logs/access_log combined
//设置 Apache 自己产生的页面中使用 Apache 服务器版本的签名
ServerSignature On

//设置内容协商目录的访问别名
Alias /icons/ "/var/www/icons/"
//设置/var/www/icons 目录的访问权限
<Directory "/var/www/icons">
// MultiViews: 使用内容协商功能决定被发送的网页的性质
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

//设置 Apche 手册的访问别名
Alias /manual "/var/www/manual"
//设置/var/www/manual 目录的访问权限
<Directory "/var/www/manual">
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Order allow,deny
```

```
    Allow from all
</Directory>

//指定 DAV 加锁数据库文件的存放位置
<IfModule mod_dav_fs.c>
    DAVLockDB /var/lib/dav/lockdb
</IfModule>

//设置 CGI 目录的访问别名
ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"
//由于 Red Hat 中不使用 worker MPM 运行方式，所以不加载 mod_cgid.c 模块
<IfModule mod_cgid.c>
Scriptsock          run/httpd.cgid
</IfModule>
//设置 CGI 目录的访问权限
<Directory "/var/www/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>

//设置自动生成目录列表的显示方式
// FancyIndexing: 对每种类型的文件前加上一个小图标以示区别
// VersionSort: 对同一个软件的多个版本进行排序
// NameWidth=*: 文件名子段自动适应当前目录下最长文件名
IndexOptions FancyIndexing VersionSort NameWidth=*
//当使用 IndexOptions FancyIndexing 之后，配置下面的参数，
//用于告知服务器在遇到不同的文件类型或扩展名时采用 MIME 编码格式
//辨别文件类型并显示相应的图标
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*
//当使用 IndexOptions FancyIndexing 之后，配置下面的参数，
//用于告知服务器在遇到不同的文件类型或扩展名时采用所指定的格式
//并显示所对应的图标
AddIcon /icons/binary.gif .bin .exe
……
AddIcon /icons/blank.gif ^^BLANKICON^^
//当使用 IndexOptions FancyIndexing 之后，且无法识别文件类型时
//显示此处定义的图标
DefaultIcon /icons/unknown.gif

//当服务器自动列出目录列表时，在所生成的页面之后显示 README.html 的内容
```

```
ReadmeName README.html
//当服务器自动列出目录列表时，在所生成的页面之前显示 HEADER.html 的内容
HeaderName HEADER.html

//设置在线浏览用户可以实时解压缩.Z .gz .tgz 类型的文件
//并非所有浏览器都支持
AddEncoding x-compress Z
AddEncoding x-gzip gz tgz

//设置网页内容的语言种类（浏览器要启用内容协商）
//对中文网页，此项无实际意义
AddLanguage da .dk
……
AddLanguage hr .hr

//当启用内容协商时，设置语言的先后顺序
LanguagePriority en da nl et fr de el it ja kr no pl pt pt-br ltz ca es sv
tw
// Prefer: 当有多种语言可以匹配时，使用 LanguagePriority 列表的第 1 项
// Fallback: 当没有语言可以匹配时，使用 LanguagePriority 列表的第 1 项
ForceLanguagePriority Prefer Fallback

//设置默认字符集
AddDefaultCharset ISO-8859-1
//设置各种字符集
AddCharset ISO-8859-1  .iso8859-1  .latin1
……
AddCharset shift_jis   .sjis

//添加新的 MIME 类型（避免用户编辑/etc/mime.types）
AddType application/x-tar .tgz

//设置 Apache 对某些扩展名的处理方式
AddHandler imap-file map
AddHandler type-map var
//使用过滤器执行 SSI
AddOutputFilter INCLUDES .shtml
//设置错误页面目录的别名
Alias /error/ "/var/www/error/"
//设置/var/www/error 目录的访问权限
<IfModule mod_negotiation.c>
<IfModule mod_include.c>
   <Directory "/var/www/error">
      AllowOverride None
      Options IncludesNoExec
      AddOutputFilter Includes html
      AddHandler type-map var
```

```
        Order allow,deny
        Allow from all
        LanguagePriority en es de fr
        ForceLanguagePriority Prefer Fallback
    </Directory>
    //设置错误输出页面
    ErrorDocument 400 /error/HTTP_BAD_REQUEST.html.var
    ErrorDocument 401 /error/HTTP_UNAUTHORIZED.html.var
    ……
    ErrorDocument 506 /error/HTTP_VARIANT_ALSO_VARIES.html.var
</IfModule>
</IfModule>
//设置浏览器匹配
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0
BrowserMatch "Microsoft Data Access Internet Publishing Provider"
redirect-carefully
BrowserMatch "^WebDrive" redirect-carefully
#
```

重点

下面将 Red Hat 9 的 Apache 的默认重要配置信息汇总如下:

配置文件: /etc/httpd/conf/httpd.conf

服务器的根目录: /etc/httpd

根文档目录: /var/www/html

访问日志文件: /var/log/httpd/access_log

错误日志文件: /var/log/httpd/error_log

运行 Apache 的用户: apache

运行 Apache 的组: apache

端口: 80

模块存放路径: /usr/lib/httpd/modules

prefork MPM 运行方式的参数:

StartServers 8

MinSpareServers 5

MaxSpareServers 20

MaxClients 150

MaxRequestsPerChild 1000

7.3　配置 Apache

本节内容
Ø　基本配置 Ø　分割配置任务 Ø　配置每个用户的 Web 站点
学习目标
Ø　掌握 Apache 最基本的配置指令 Ø　学会使用 Include 指令和.htaccess 文件分割配置任务 Ø　掌握为每个用户配置 Web 站点的的方法

7.3.1　基本配置

默认配置为用户提供了一个良好的模板。基本的配置几乎不需要进行修改。但用户应该考虑修改或添加如下的基本配置指令。

1．KeepAlive

将 KeepAlive 的值设为 On，以便提高访问性能。

2．MaxClients

根据服务容量修改此值。

3．ServerAdmin

将 ServerAdmin 的值设为 Apache 服务器管理员的 E-mail 地址。

4．ServerName

首先删除 ServerName 前的注释符号“#”，然后设置服务器的 FQDN。

5．DirectoryIndex

在此指令后添加其他的默认主页文件名，例如可以添加 index.htm 等。

6．IndexOptions

可以在此指令后添加 FoldersFirst 表示让目录列在前面（类似于资源管理器）。

7.3.2　分割配置任务

1．使用 Include 指令

可以使用 Include 指令将主配置文件进行分割。例如可以将所有与虚拟主机配置相关的

配置单独存成一个配置文件，然后在主配置文件中将其包含进来。

在 Red Hat Linux 9 的默认配置中，就包含了一个 Include 指令 Include conf.d/*.conf，用于将/etc/httpd/conf.d 目录下的所有以 conf 结尾的配置文件包含进来。

2．使用.htaccess 文件

可以使用.htaccess 文件改变主配置文件中的配置，但是它只能设置对目录的访问控制，这个目录就是.htaccess 文件存放的目录。与使用 Include 指令不同，.htaccess 文件中的配置可以覆盖主配置文件中的配置，而使用 Include 指令只是将子配置文件简单的包含进主配置文件之中。

（1）何时使用.htaccess 文件。有如下两种情况需要使用.htaccess 文件：

- 在多个用户之间分割配置。
- 想在不重新启动服务器的情况下改变服务器配置。


提示

在可能的情况下尽量避免使用.htaccess 文件，因为使用.htaccess 文件会降低服务器的运行性能。

（2）要使用.htaccess 文件必须经过两个配置步骤：

- 首先在主配置文件中启用并控制对.htaccess 文件的使用。
- 然后在需要覆盖主配置文件的目录下生成.htaccess 文件。

3．启用并控制使用.htaccess 文件

（1）设置文件名称。必须保证在主配置文件中包含如下的配置语句：

```
AccessFileName .htaccess
<Files ~ "^\.htaccess">
    Order allow,deny
    Deny from all
</Files>
```

（2）控制在.htaccess 文件中可以使用的指令组。要控制在.htaccess 文件中可以使用的指令组，需要在主配置文件中使用 AllowOverride 指令。表 7-3 列出了可以在 AllowOverride 指令所使用的指令组。

表 7-3　AllowOverride 指令所使用的指令组

指 令 组	可　用　指　令	说　　明
AuthConfig	AuthDBMGroupFile, AuthDBMUserFile, AuthGroupFile, AuthName, AuthType, AuthUserFile, Require	进行认证、授权以及安全的相关指令
FileInfo	DefaultType, ErrorDocument, ForceType, LanguagePriority, SetHandler, SetInputFilter, SetOutputFilter	控制文件处理方式的相关指令
Indexes	AddDescription, AddIcon, AddIconByEncoding, AddIconByType,	控制目录列表方式的相关指令

（续）

指 令 组	可　用　指　令	说　　明
	DefaultIcon, DirectoryIndex, FancyIndexing, HeaderName, IndexIgnore, IndexOptions, ReadmeName	
Limit	Allow,Deny,Order	进行目录访问控制的相关指令
Options	Options, XBitHack	启用不能在主配置文件中使用的各种选项
All	全部指令组	可以使用以上所有指令
None	禁止使用所有指令	禁止处理.htaccess 文件

4．生成.htaccess 文件

当在主配置文件中配置了对.htaccess 文件的启用和控制之后，接下来就可以在需要覆盖主配置文件的目录下生成.htaccess 文件。.htaccess 文件中可以使用的配置指令取决于主配置文件中 AllowOverride 指令的设置。

5．使用.htaccess 文件举例

下面举一个简单的例子说明.htaccess 文件的使用。

操作步骤 7.5　配置使用.htaccess 文件

```
//首先在文档根目录下生成一个 private 目录，并创建测试文件
# cd /var/www/html
# mkdir private
# cd private
# touch test
//修改配置前，在客户浏览器查看结果，如图 7-3 所示
//修改主配置文件
# vi /etc/httpd/conf/httpd.conf
//添加如下配置语句
<Directory "/var/www/html/private">
   AllowOverride Options
</Directory>
#
//重新启动 httpd
# service httpd restart
//在/var/www/html/private 目录下生成.htaccess 文件
# vi /var/www/html/private/.htaccess
//添加如下配置语句
Options –Indexes
#
//在客户浏览器中查看结果，如图 7-4 所示
//通过查看配置结果，可以证明.htaccess 已经生效。即对 private 目录
//的访问不生成文件列表
```

图 7-3　在配置.htaccess 文件之前查看结果

图 7-4　在配置.htaccess 文件之后查看结果

注意　在上面的例子中，是先重新启动了 Apache 服务器，然后才生成.htaccess 文件。也就是说，对.htaccess 文件的修改不用重新启动服务器即可生效。

7.3.3　配置每个用户的 Web 站点

1．配置步骤

配置每个用户的 Web 站点的意图是使在安装了 Apache 的本地计算机上，拥有用户账号的每个用户都能够架设自己单独的 Web 站点。

要配置每个用户的 Web 站点，要经过下面的配置步骤：

- 修改主配置文件启用每个用户的 Web 站点配置。
- 修改主配置文件为每个用户的 Web 站点目录配置访问控制。

2．配置举例

下面举例说明配置步骤。

操作步骤 7.6　配置每个用户的 Web 站点

```
//修改主配置文件
# vi /etc/httpd/conf/httpd.conf
//修改如下部分的配置
<IfModule mod_userdir.c>
    //基于安全考虑，禁止 root 用户使用自己的个人站点
    UserDir disable root
    //配置对每个用户 Web 站点目录的设置
    UserDir public_html
</IfModule>
//设置每个用户 Web 站点目录的访问权限，将下面配置行前的“#”去掉
<Directory /home/*/public_html>
   AllowOverride FileInfo AuthConfig Limit
   Options  MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
   <Limit GET POST OPTIONS>
      Order allow,deny
      Allow from all
   </Limit>
   <LimitExcept GET POST OPTIONS>
      Order deny,allow
      Deny from all
    </LimitExcept>
</Directory>
//重新启动 httpd
# service httpd restart
```

下面说明每个用户要创建自己的 Web 站点，需要执行的步骤。以 osmond 用户为例，想要创建自己的 Web 站点的每个用户都要执行下面的步骤。

操作步骤 7.7　用户为创建自己的 Web 站点需要执行的步骤

```
//查看当前用户
$ whoami
osmond
//回到自家目录的根
$ cd
//创建目录 public_html
$ mkdir public_html
//修改 osmond 目录的权限
$ cd ..
$ chmod 711 osmond
//创建 index.html
```

```
$ cd ~/public_html/
$ vi index.html
//显示 index.html 的内容
$ cat index.html
Osmond's Web Site.
//使用客户浏览器访问自己的主页，如图 7-5 所示
```

图 7-5　访问用户自己的 Web 站点


提示

（1）上例中直接用 vi 编辑了一个 index.html 文件用于测试。实际应该使用 ftp 或 DAV 方式上传文件。

（2）使用浏览器访问自己的主页时，使用下面格式的 URL:

http://IP 地址或 FQDN/~用户名

若主机的 IP 地址为 192.168.1.100，用户名为 osmond，URL 为 http://192.168.1.100/~osmond。

7.4　访问控制、认证和授权

本节内容
Ø 访问控制 Ø 认证和授权 Ø 认证和授权的配置举例 Ø 访问控制、认证和授权的综合应用 Ø WebDAV
学习目标
Ø 掌握基于主机的访问控制的配置方法 Ø 学会使用认证和授权指令配置对授权区的访问 Ø 掌握对访问控制和认证授权进行控制指令的使用 Ø 了解 WebDAV 并掌握其配置方法

7.4.1　访问控制

1．访问控制的配置指令

Apache 使用下面的 3 个指令配置访问控制：

- Order：用于指定执行允许访问规则和执行拒绝访问规则的先后顺序。
- Deny：定义拒绝访问列表。
- Allow：定义允许访问列表。

（1）Order。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。


注意

在书写 Allow,Deny 和 Deny,Allow 时，中间不能添加空格字符。

（2）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 jamond.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 192.168.1.0/255.255.255.255.0 。
- CIDR 规范：如 192.168.1.0/24。

2．访问控制配置举例

下面将以查看服务器配置信息为例讲解访问控制的使用。

操作步骤 7.8　配置访问控制

```
//修改主配置文件
# vi /etc/httpd/conf/httpd.conf
//将下面配置行前的“#”去掉
<Location /server-info>
    //由 mod_info 模块生成服务器配置信息
    SetHandler server-info
    //先执行 deny 规则再执行 allow 规则
    Order deny,allow
    //拒绝所有的客户，只允许来自 192.168.1.77 的访问
    Deny from all
    Allow from 192.168.1.77
```

```
</Location>
#
//重新启动 httpd
# service httpd restart
//下面在客户浏览器上进行测试
//在 IP 地址为 192.168.1.77 的主机上的结果如图 7-6 所示
```

图 7-6　访问控制配置测试－在被允许的主机上访问

```
//在其他主机上的结果如图 7-7 所示
```

图 7-7　访问控制配置测试－在被拒绝的主机上访问


提示

虽然上例中的访问控制是在 Location 容器中设置的，但这种方法也适用于其他容器，如 Directory 容器和 Files 容器。

7.4.2　认证和授权

1．认证

（1）两种认证类型。在 RFC 2617 中定义了两种认证方式，分别为：

- 基本（Basic）认证
- 摘要（Digest）认证

摘要认证比基本认证更加安全，但遗憾的是目前并非所有的浏览器都支持摘要认证，所以大多数情况下用户只使用基本认证。本节主要介绍基本认证。

（2）认证的配置指令。所有的认证配置指令既可以出现在主配置文件的 Directory 容器中，也可以出现在./htaccess 文件中。表 7-4 列出了可用的认证配置指令。

表 7-4　Apache 的认证配置指令

指　令	指 令 语 法	说　明
AuthName	AuthName 领域名称	定义受保护领域的名称
AuthType	AuthType Basic 或 Digest	定义使用的认证方式
AuthGroupFile	AuthGroupFile 文件名	指定认证口令文件的位置
AuthUserFile	AuthUserFile 文件名	指定认证组文件的位置

2．授权

当使用认证指令配置了认证之后，还需要为指定的用户或组进行授权。为用户或组进行授权的指令是 Require。Require 指令的三种使用格式如表 7-5 中的说明。

表 7-5　Apache 的授权配置指令的使用格式

指令语法格式	说　明
Require user 用户名 [用户名] ……	授权给指定的一个或多个用户
Require group 组名 [组名] ……	授权给指定的一个或多个组
Require valid-user	授权给认证口令文件中的所有用户

3．管理认证口令文件和认证组文件

本节讲述基于文本文件的认证口令文件和认证组文件，关于基于数据库的认证口令文件和认证组文件的相关内容请参考 Apache 手册。

（1）管理认证口令文件

1）创建新的认证口令文件

可以使用如下命令，在添加一个认证用户的同时创建认证口令文件：

```
# htpasswd –c 认证口令文件名 用户名
```

2）修改认证口令文件

可以使用如下命令，向现存的口令文件中添加用户或修改已存在的用户的口令：

```
# htpasswd 认证口令文件名 用户名
```

3）认证口令文件的格式

与系统中的/etc/shadow 文件类似，认证口令文件中每一行包含一个用户的用户名和加密的口令：

```
用户名：加密的口令
```


注意

（1）基于安全因素的考虑，认证口令文件和下面讲述的认证组文件不应该与 Web 文档存在于相同的目录下，建议存放在/var/www/目录或其子目录下，也可以存放在配置目录/etc/httpd/目录或其子目录下。

（2）htpasswd 没有提供删除用户的选项，要想删除用户，可以直接使用文本编辑器对认证口令文件进行编辑，删除指定用户的行即可。

（2）管理认证组文件。Apache 没有提供创建认证组文件的命令，它只是一个文本文件，用户可以使用任何的文本编辑器创建并修改此文件。该文件中每一行的格式如下：

```
组名：用户名 用户名 ……
```


注意

在认证组文件中指定的用户名，必须先添加到认证口令文件中。

7.4.3 认证和授权配置举例

1．在主配置文件中配置认证和授权

下面举例说明具体的配置步骤。

操作步骤 7.9　在主配置文件中配置认证和授权

```
//修改主配置文件
# vi /etc/httpd/conf/httpd.conf
//添加如下的配置行
<Directory "/var/www/html/private">
       //不使用.htaccess 文件
       AllowOverride None
       //指定使用基本认证方式
       AuthType Basic
       //指定认证领域名称
       AuthName "jamond"
       //指定认证口令文件的存放位置
       AuthUserFile /var/www/passwd/jamond
```

```
        //授权给认证口令文件中的所有用户
        require valid-user
</Directory>
#
//创建认证口令文件，并添加两个用户
# mkdir /var/www/passwd
# cd /var/www/passwd
# htpasswd -c jamond osmond
New password:
Re-type new password:
Adding password for user osmond
# htpasswd  jamond jason
New password:
Re-type new password:
Adding password for user jason
#
//将认证口令文件的属主改为 apache
# chown apache.apache jamond
#
//重新启动 httpd
# service httpd restart
#
//在客户端使用浏览器检测配置，结果如图 7-8 所示
```

图 7-8　认证和授权测试

输入了用户名和口令之后，将进入如图 7-3 所示的界面。


注意

由于 Apache 的子进程以 apache 用户运行，所以认证口令文件和认证组文件的属主应该设为 apache。只有这样才能让 apache 子进程在认证过程中读取这两个文件的内容。

2．在.htaccess 文件中配置认证和授权

下面举例说明具体的配置步骤。

操作步骤 7.10　在.htaccess 文件中配置认证和授权

```
//修改主配置文件
# vi /etc/httpd/conf/httpd.conf
//将/var/www/html/private 目录的访问权限设为
<Directory "/var/www/html/private">
       //允许在.htaccess 文件中使用认证和授权指令
       AllowOverride AuthConfig
</Directory>
#
//重新启动 httpd
# service httpd restart
//继续使用操作步骤 7.8 中生成的认证口令文件
//接下来，在/var/www/html/private 目录下生成.htaccess 文件
# vi .htaccess
//添加如下内容
AuthType Basic
AuthName "jamond"
AuthUserFile /var/www/passwd/jamond
//授权给用户 osmond 和 jason
require user osmond Jason
#
//在客户端使用浏览器检测配置，结果与图 7-8 所示的一致
```


提示

通过以上例子可知使用两种方法都可以使指定的用户访问授权区，它们具有相同的效果。具体使用哪种方法将由管理员做出权衡。

7.4.4　访问控制、认证和授权的综合应用

1．对访问控制和认证授权进行控制

在对一个容器（Directory 或 Files 或 Location）同时配置了访问控制和认证授权之后，这两类指令是否都会起作用，将由一个指令进行控制，这个指令是 Satisfy。Satisfy 指令只有在对一个容器同时设置了访问控制和认证授权后才起作用。Satisfy 指令有两种可能的取值：

- Satisfy all：访问控制和认证授权两类指令均起作用（默认值）。

Ø　Satisfy any：只要一类指令满足条件即可以访问。

下面举例说明 Satisfy 指令的使用。

2．配置指定的用户在指定的网段上访问资源

下面将以查看服务器运行状态信息为例说明 Satisfy all 的使用。

操作步骤 7.11　配置指定的用户在指定的网段上访问资源

```
//修改主配置文件
# vi /etc/httpd/conf/httpd.conf
//添加如下的配置行
<Location /server-status>
    SetHandler server-status
    Order deny,allow
    Deny from all
    //允许 192.168.1 网段内主机的访问
    Allow from  192.168.1
    //配置认证授权
    AuthType Basic
    AuthName "Admin"
    AuthUserFile /var/www/passwd/jamond
    AuthGroupFile /var/www/passwd/admingrp
    require group admin
    Satisfy all
</Location>
#
//创建认证组文件并更改属主
# vi /var/www/passwd/admingrp
//添加下面的行
admin: osmond jason
# chown apache.apache /var/www/passwd/admingrp
#
//重新启动 httpd
# service httpd restart
#
//在客户端使用浏览器检测配置,在 192.168.1 网段上的主机的访问结果如图 7-9
所示
//当通过用户认证之后即可看到如图 7-10 所示的服务器运行状态信息
//而非 192.168.1 网段的主机以及没有 Admin 组身份的人都不能访问
```

3．允许网段内用户无条件访问而其他用户授权访问

只要将操作步骤 7.10 中的 Satisfy all 改为 Satisfy any 就可以实现 192.168.1 网段内的用户无需认证即可访问，而在外网上可以允许 admin 组内的用户经过认证后访问。

图 7-9　配置指定的用户在指定的网段上访问资源

图 7-10　查看服务器运行状态信息

7.4.5　WebDAV

1．WebDAV 简介

DAV 是分布式授权和版本控制的缩写，而 WebDAV 是基于 Web 的分布式授权和版本控制。传统情况下，用户使用 FTP 或 NFS 对站点内容进行上传或更新。但是有许多人士认为 FTP 和 NFS 是不安全的协议，尽量不要在运行 Web 服务器的计算机上运行 FTP 和 NFS 服务器。然而不运行这两种服务器，用户就无法对自己的站点内容进行维护，WebDAV 提供了一种新的基于 HTTP 协议的解决方案。WebDAV 的官方网站是 http://www.webdav.org。

当对 Apache 配置了对 WebDAV 支持以后，用户就可以在支持 WebDAV 的客户端上对站点内容进行上传和维护。

2．限制应用认证授权的时机

在介绍下面的内容之前有必要先了解一下 HTTP 的请求方法。当 HTTP 客户端与服务器建立连接后就要发送请求，请求可以分为 3 部分：方法、请求资源和 HTTP 版本号。HTTP 请求方法通常是 GET、POST 和 HEAD，但 HTTP 规范（RPC 2616）中还定义了许多 HTTP 请求方法。这些方法包括：GET、POST、PUT、DELETE、CONNECT、OPTIONS、TRACE、PATCH、PROPFIND、PROPPATCH、MKCOL、COPY、MOVE、LOCK 和 UNLOCK。

要告知 Apache 为指定 HTTP 的请求方法进行配置，可以使用<Limit>和<LimitExcept>指令。例如下面的指令片断可以允许任何人浏览网站，但是只允许一组特殊的用户应答 CGI 程序。

```
AuthType Basic
AuthName "Example"
AuthUserFile /var/www/passwd/jamond
AuthGroupFile /var/www/passwd/admingrp
<Limit POST>
    require group admin
</Limit>
```

在对 Apache 配置 WebDAV 时也要使用类似的配置，从而让任何人都能浏览网站，指定的用户才能对网站进行上传和更新。

3．配置 WebDAV

在 Apache 2.0 中默认包含了支持 WebDAV 的模块 mod_dav。下面讲述配置过程。

操作步骤 7.12　配置 WebDAV

```
//修改主配置文件
# vi /etc/httpd/conf/httpd.conf
//将文档根目录的访问控制改为
<Directory "/var/www/html">
   Options Indexes FollowSymLinks
   AllowOverride None
   //启用 WebDAV
   Dav On
   //配置认证指令
   AuthType Basic
   AuthName "Admin"
   AuthUserFile /var/www/passwd/jamond
   AuthGroupFile /var/www/passwd/admingrp
   //配置条件授权，即对非浏览的 HTTP 请求方法进行认证授权
   <LimitExcept GET OPTIONS>
      require group admin
   </LimitExcept>
```

```
</Directory>
#
//重新启动 httpd
# service httpd restart
#
//将服务器根文档目录的属主设为 apache
# chown -R apache.apache /var/www/html/
#
```

注意

必须将 WebDAV 所管理的目录的属主设为 apache，以便以 apache 用户运行的 Apache 子进程能对目录内容进行更新。

4．使用 WebDAV 客户

在各种操作系统平台上都有 WebDAV 客户端。下面以 Windows 2000 中提供的 Web 文件夹的使用为例介绍 WebDAV 客户的使用。另外，Windows 平台上 FrontPage 和 DreamWeaver 也支持 DAV。

为了使用 Windows 2000 提供的 Web 文件夹，可以先打开“网上邻居”，然后双击“添加网上邻居”，如图 7-11 所示。

图 7-11　添加网上邻居向导

在对话框中输入使用 HTTP 协议的 URL，既可以使用 IP 地址也可以使用 FQDN。之后单击“下一步”按钮，弹出认证对话框，如图 7-12 所示。

图 7-12　认证对话框

输入 Admin 组中的用户名和密码后，单击“确定”按钮进入如图 7-13 所示的界面。

图 7-13　完成添加网上邻居

更改网上邻居的名称后，将会看到如图 7-14 所示的 Web 文件夹。

在该界面下，用户可以像使用局域网资源一样来对 Web 文件夹进行操作，对站点内容进行更新。如果用户使用浏览器访问此站，将会直接浏览到主页而不会弹出认证界面，这正是我们所需要的结果。

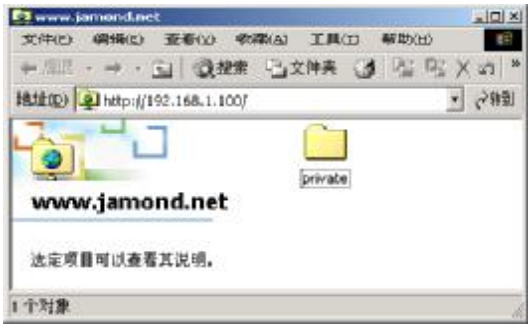

图 7-14　Web 文件夹


注意　为了安全的考虑，当用户更新站点完毕，一定要将此 Web 文件夹删除。

7.5 组织和管理站点内容

本节内容
Ø 组织和管理站点内容的方法 Ø 符号链接和别名 Ø 页面重定向
学习目标
Ø 掌握管理和组织站点内容的方法 Ø 学会使用符号链接和别名对站点内容进行扩充 Ø 掌握页面重定向的配置方法

7.5.1 组织和管理站点内容的方法

Web 服务器是更新量最大的服务器之一，随着时间的推移站点内容将越来越多。在维护方面会带来两方面的问题：

（1）如何在文档根目录空间不足的情况下继续添加站点内容。

（2）如何在文件移动位置之后使得用户仍然能够访问。

对于第 1 个问题有 3 种解决方法：

Ø 挂装新的磁盘分区到文档目录（可以是本地磁盘分区或远程 NFS 分区）。

Ø 在文档目录下使用符号链接使文档目录之外的内容被访问。

Ø 使用别名机制使文档目录之外的内容被访问。

对于第 2 个问题的解决方法是使用页面重定向。

7.5.2 符号链接和别名

关于挂装磁盘分区的方法属于系统管理的范畴，本书不做过多叙述。下面举例讲解符号链接和别名的使用。

1．符号链接

在 Apache 的默认配置中已经包含了符号链接的指令：

```
<Directory />
    Options FollowSymLinks
</Directory>
<Directory "/var/www/html">
    Options Indexes FollowSymLinks
</Directory>
```

所以只要在根文档目录下使用 ln –s 命令创建符号链接即可。

操作步骤 7.13　创建符号链接对站点内容进行扩展

```
# cd /var/www/html
# ln -s /usr/share/doc doc
//可以使用客户端浏览器进行测试，如图 7-15 所示
```

图 7-15　测试符号链接对站点内容进行扩展

注意

为了提高 Apache 服务器的性能，建议不要修改 Apache 默认对符号链接跟随的配置，因为如果取消了对符号链接的跟随，服务器在每收到一个请求时都要判断其是否为符号链接，从而影响服务器的性能。但是启用符号链接跟随将带来安全性隐患，这需要管理员在性能和安全两方面做出权衡。

2．别名

使用别名是另一种将根文档目录以外的内容加入站点的方法。在 Apache 的默认配置中，由于 error 目录和 manual 目录都在文档根目录 html 之外，所以设置了这两个目录的别名访问，同时还使用 Directory 容器配置了对别名目录的访问权限。下面再举一个简单的使用别名的例子。

操作步骤 7.14　使用别名对站点内容进行扩展

```
//编辑主配置文件
# vi /etc/httpd/conf/httpd.conf
//添加如下的配置
Alias /ftp /var/ftp/pub
<Directory "/var/ftp/pub">
    AllowOverride None
    Options Indexes
    Order allow,deny
```

```
    Allow from all
</Directory>
#
//重新启动 apache 之后可以使用客户端浏览器进行测试，如图 7-16 所示
```

图 7-16　测试别名对站点内容进行扩展

7.5.3　页面重定向

当用户经常访问某个站点的目录时，他很可能会记住这个目录的 URL，在每次访问站点时直接在浏览器地址栏里键入 URL。但是如果站点进行了结构更新后，用户再使用原来的 URL 访问就会出现“页面没找到”的信息，为了方便用户可以继续使用原来的 URL 访问，就需要配置页面重定向。

1．页面重定向配置指令

进行页面重定向需要使用 Redirect 指令，其命令语法为：

```
Redirect　[错误响应代码]　用户请求的 URL　[重定向的 URL]
```

其中常用的错误响应代码及其说明列于表 7-6。

表 7-6　常用的错误响应代码

代　码	说　明
301	告知用户请求的 URL 已经永久的移动到新的 URL，用户可以记住新的 URL，以便日后直接使用新的 URL 进行访问
302	告知用户请求的 URL 临时的移动到新的 URL，用户无需记住新的 URL，如果省略错误响应代码，默认就是此值
303	告知用户页面已经被替换，用户应该记住新的 URL
410	告知用户请求的页面已经不再存在，使用此代码时不应该使用重定向的 URL 参数

由于重定向的是 URL 地址，所以重定向后的地址可以是源站点之外的任意站点地址。

2．页面重定向配置举例

假如某个静态站点用如图 7-17 所示的目录结构组织站点新闻。

当月的新闻存放在 news 目录下的子文件夹中，例如：六月的新闻存放在 news/jun 目录下，当六月过去之后，管理员将 jun 目录移动到 old-news 目录，同时将访问 news/jun 的 URL 重定向到 old-news，那么需要执行如下的操作步骤。

图 7-17　页面重定向举例－站点结构

操作步骤 7.15　页面重定向

```
//为了进行试验，先创建目录结构和页面
# cd /var/www/html
# mkdir news old-news
# mkdir news/jun
# mkdir old-news/jun
# echo "Jun news">news/jun/index.html
# echo "Jun news (old)">old-news/jun/index.html
#
//在配置前进行测试，如图 7-18 所示
```

图 7-18　在页面重定向之前测试

```
//编辑主配置文件
# vi /etc/httpd/conf/httpd.conf
//添加如下行
Redirect 303 /news/jun http://192.168.1.100/old-news/jun
#
//重新启动 httpd
# service httpd restart
//在配置后使用原来的 URL 进行测试，页面已经重定向，如图 7-19 所示。
```

图 7-19　在页面重定向之后测试

7.6　Apache 的日志管理和统计分析

本节内容
Ø　日志管理简介 Ø　配置错误日志 Ø　配置访问日志 Ø　日志滚动 Ø　日志统计分析
学习目标
Ø　理解为什么要对 Web 日志加以重视 Ø　学会配置错误日志并能看懂其内容 Ø　学会配置访问日志并能看懂其内容 Ø　掌握两种日志滚动的配置方法 Ø　学会配置 Webalizer 并对访问日志进行分析

7.6.1　日志管理简介

1．Web 日志的重要性

对于所有的公司或 ICP 来说，除了要保证网站稳定正常的运行以外，一个重要的问题就是网站访问量的统计和分析报表，这对于了解和监控网站的运行状态，对提高各个网站的服务能力和服务水平是必不可少的。通过对 Web 服务器的日志文件进行分析和统计，能够有效掌握系统运行情况以及网站内容的被访问情况，加强对整个网站及其内容的维护与管理。

管理 Web 网站不只是监视 Web 的速度和 Web 的内容传送，它要求不仅仅关注服务器每天的吞吐量，还要了解对这些 Web 网站的外来访问，了解网站各页面的访问情况，根据各页面的点击频率来改善网页的内容和质量，提高内容的可读性，跟踪包含有商业交易的步骤以及管理 Web 网站“幕后”的数据等。

在某种程度上讲，“日志就是金钱”。因为如果能够通过日志分析出一个网站具有高流

量，则广告商就愿意为此支付费用。

2．日志的种类

Apache 的标准中规定了 4 类日志：

- 错误日志
- 访问日志
- 传输日志
- Cookie 日志

其中：传输日志和 Cookie 日志被 Apache 2.0 认为已经过时。所以本节仅仅讨论错误日志和访问日志。同时错误日志和访问日志被 Apache 2.0 默认设置。

3．日志相关配置指令

Apache 中有如下 4 条与日志相关的配置指令，见表 7-7。

表 7-7　Apache 与日志相关的配置指令

指　令	格　式	说　明
ErrorLog	ErrorLog 错误日志文件名	指定错误日志存放路径
LogLevel	LogLevel 错误日志记录等级	指定错误日志的记录等级
LogFormat	LogFormat 记录格式说明串 格式昵称	为一个日志记录格式命名
CustomLog	CustomLog 访问日志文件名 格式昵称 CustomLog“|程序名 访问日志文件名”格式昵称	指定访问日志存放路径和记录格式 指定访问日志由指定的程序生成并指定日志记录格式（管道日志）

表中前两条指令用于配置错误日志，后两条指令用于配置访问日志。

7.6.2　配置错误日志

1．Apache 默认的错误日志配置

```
ErrorLog logs/error_log
LogLevel warn
```

配置错误日志相对简单，只要说明日志文件的存放路径和日志记录等级即可。

2．日志记录等级

下面着重说说日志记录等级，如表 7-8 所示。

表 7-8　错误日志记录等级

紧急程度	等　级	说　明
1	emerg	出现紧急情况使得该系统不可用，如系统宕机等
2	alert	需要立即引起注意的情况
3	crit	危险情况的警告

（续）

紧急程度	等　　级	说　　明
4	error	除了 emerg、alert、crit 的其他错误
5	warn	警告信息
6	notice	需要引起注意的情况，但不如 error、warn 重要
7	info	值得报告的一般消息
8	debug	由运行于 debug 模式的程序所产生的消息

如果指定了等级 warn，那么就记录紧急程度为 1 至 5 的所有错误信息。

3．错误日志文件举例

下面是一个错误日志文件的截取。

```
# cat /etc/httpd/logs/error_log
[Thu Jun 19 15:28:56 2003] [error] [client 192.168.1.77] Directory index forbidden by rule:
/var/ftp/pub/
[Thu Jun 19 17:16:24 2003] [notice] caught SIGTERM, shutting down
[Thu Jun 19 17:17:07 2003] [notice] Digest: done
```

从文件内容可以看出，每一行记录了一个错误。格式为：

```
日期和时间　　错误等级　　错误消息
```

7.6.3　配置访问日志

1．访问日志的分类

为了便于分析 Apache 的访问日志，Apache 的默认配置文件中，按记录的信息不同（用不同格式昵称说明不同的信息）将访问日志分为 4 类，并由 LogFormat 指令定义了昵称，如表 7-9 所示。

表 7-9　访问日志的格式分类

格 式 分 类	格式昵称	说　　明
普通日志格式(common log format,CLF)	common	大多数日志分析软件都支持这种格式
参考日志格式(referer log format)	referer	记录客户访问站点的用户身份
代理日志格式(agent log format)	agent	记录请求的用户代理
综合日志格式(combined log format)	combined	结合以上三种日志信息

由于综合日志格式简单地结合了 3 种日志信息，所以在配置访问日志时，要么使用 3 个文件分别记录，要么使用一个综合文件进行记录。

若使用一个综合文件进行记录（默认的），配置为：

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
CustomLog logs/access_log combined
```

若使用 3 个文件分别进行记录，配置为：

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
CustomLog logs/access_log common
CustomLog logs/referer_log referer
CustomLog logs/agent_log agent
```

2．LogFormat 指令

下面讲解一下格式相对复杂的 LogFormat 指令。在使用 LogFormat 指令定义格式昵称时可以使用如表 7-10 所示的格式说明符。

表 7-10 中的例子是对如下日志记录的分解。

```
192.168.1.77 - - [19/Jun/2003:23:03:33 +0800] "GET /manual/style/manual.css HTTP/1.1"
404 1203 "http://192.168.1.100/manual/logs.html" "Mozilla/4.0 (compatible; MSIE 6.0;
Windows NT 5.0)"
```

表 7-10　LogFormat 指令常用的格式说明符

格式说明符	说　　明	举　　例
%h	客户机的 IP 地址	192.168.1.77
%l	从 identd 服务器中获取远程登录名称	-（-表示没有取得该项信息）
%u	来自于认证的远程用户	-（-表示没有取得该项信息）
%t	连接的日期和时间	[19/Jun/2003:23:03:33 +0800]
%r	HTTP 请求的首行信息	GET /manual/style/manual.css HTTP/1.1
%>s	响应请求的状态代码	404
%b	传送的字节数（不包含 HTTP 头信息）	1203
%{Referer}i	发给服务器的请求头信息	http://192.168.1.100/manual/logs.html
%{User-Agent}i	使用的浏览器信息	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)

提示　由于整个格式说明字符串是放在""之内的，所以若要输出的日志信息内含有引号，需要将"前加转义符\。例如：若要输出子串"GET /apache_pb.gif HTTP/1.0"，则格式字符串为\"%r\"。

7.6.4　日志滚动

1．为什么使用日志滚动

由于 Apache 进程本身不负责对日志文件进行滚动，因此必须使用其他程序配置日志滚动。

所有的日志文件都会随着时间的推移和访问次数的增加而迅速增长，因此必须对日志文件进行定期清理以免造成磁盘空间的不必要的浪费。同时也加快了管理员查看日志所用的时间，因为打开小文件的速度比打开大文件的速度要快。

2．使系统重新使用空的日志文件

要使系统重新使用空的日志文件，可以执行如下的命令：

```
cd /etc/httpd/logs/
mv access_log access_log.old
mv error_log error_log.old
apachectl graceful
```

上面的指令片断是进行日志滚动的基础，用户可以自行编制脚本让 cron 执行。下面介绍两种常用的日志滚动配置方法。

3．使用系统的 logrotate 实现日志滚动

logrotate 是 Linux 系统自带的一个日志滚动程序，是专门对各种系统日志（包括：syslogd，mail 等）进行日志滚动的程序。使用 logrotate 和 crond 是 Red Hat Linux 9 默认的实现 Apache 日志滚动的方法。

操作步骤 7.16　查看 logrotate 实现日志滚动的配置

```
//由以下命令可以看出 logrotate 是由 crond 每天运行一次实现日志滚动
# grep logrotate /etc/cron.daily/logrotate
/usr/sbin/logrotate /etc/logrotate.conf
#
//查看 logrotate 的配置文件
# cat /etc/logrotate.conf
//每周清理一次日志文件
weekly
//保存过去四周的日志文件
rotate 4
//清除旧日志文件的同时，创建新的空日志文件
create
//包含/etc/logrotate.d 目录下的所有配置文件
include /etc/logrotate.d
#
//查看/etc/logrotate.d 目录下的与 Apache 相关的配置文件
# cat /etc/logrotate.d/httpd
//对 var/log/httpd/目录下的所有 log 文件进行操作
/var/log/httpd/*log {
    //如果日志文件丢失，将重新启用一个新的
    missingok
    //如果日志文件为空，不进行滚动
    notifempty
```

```
    //调用日志滚动通用函数
    sharedscripts
    //在日志滚动之后让 Apache 重新启动
    postrotate
       /bin/kill -HUP `cat /var/run/httpd.pid 2>/dev/null` \
    2> /dev/null || true
    endscript
    // postrotate 和 endscript 是语句括号
}
#
```

4．使用 Apache 自带的 rotatelogs 实现日志滚动

Apache 提供了不把日志直接写入文件，而是通过管道发送给另外一个程序的能力，这样就大大的加强了对日志进行处理的能力，这个通过管道得到的程序可以是任何程序：如日志分析，压缩日志等。

Apache 自带的 rotatelogs 程序可以实现日志滚动而无需使用 crond 运行。为了使用 rotatelogs 程序，需要在主配置文件中使用管道日志的配置。

即需要将：

CustomLog logs/access_log combined

修该为：

CustomLog "|/usr/sbin/rotatelogs　logs/access_log　86400" combined

其中 86400（秒）是日志滚动的时间。86400 秒就是 1 天。滚动以后的文件名为 /etc/httpd/logs/access_log.nnnnnnnnnn，这里 nnnnnnnnnn 是开始记录日志的格林威治时间距离 1970 年 1 月 1 日的秒数。当日志滚动一次后将生成一个新的日志文件，后缀为前一个日志文件的后缀值加 86400。

7.6.5　日志统计分析

1．Webalizer 简介

目前有许多日志统计分析软件，本节将介绍 Webalizer 的使用。Webalizer 是一个高效的、免费的 Web 服务器日志分析程序。其分析结果是 HTML 文件格式，从而可以很方便的通过 Web 服务器进行浏览。Internet 上的很多站点都使用 Webalizer 进行 Web 服务器日志分析。

Webalizer 具有以下一些特性：

- 是用 C 写的程序，所以具有很高的运行效率。在主频为 200Mhz 的机器上，Webalizer 每秒钟可以分析 10000 条记录，所以分析一个 40M 大小的日志文件只需要 15 秒。
- Webalizer 支持标准的普通日志文件格式；除此之外，也支持几种组合日志格式的变种，从而可以统计客户情况以及客户操作系统类型。并且现在 Webalizer 已经可以支持 wu-ftpd xferlog 日志格式以及 Squid 日志文件格式。
- 支持命令行配置以及配置文件。

- 可以支持多种语言，也可以自己进行本地化工作。
- 支持多种平台，如 UNIX、Linux、NT、 OS/2 和 MacOS 等。

2．安装 Webalizer

Red Hat Linux 9 中提供了 Webalizer 的 RPM 包，下面以 RPM 包的安装为例介绍 Webalizer 的安装。若用户在安装 Red Hat 时已经安装了 Webalizer，则可跳过下面的安装步骤。

操作步骤 7.17　安装 webalizer

```
//查看是否安装了 webalizer
# rpm -qa|grep webalizer
//将 Red Hat Linux 9 的第 1 张安装光盘放入光驱后挂装
# mount /mnt/cdrom
//进入光盘的 Red Hat/RPMS 目录
# cd /mnt/cdrom/Red Hat/RPMS
//安装所需的 RPM 包
# rpm -ivh webalizer-2.01_10-11.i386.rpm
//弹出光盘
# cd;eject
//查看安装的文件
# rpm -ql webalizer
// webalizer 会由 crond 每天运行一次
/etc/cron.daily/00webalizer
/etc/webalizer.conf
/etc/webalizer.conf.sample
/usr/bin/webalizer
/usr/bin/webazolver
/usr/share/doc/webalizer-2.01_10
/usr/share/doc/webalizer-2.01_10/README
/usr/share/man/man1/webalizer.1.gz
/var/lib/webalizer
//生成的 HTML 文件存放的路径
/var/www/html/usage
/var/www/html/usage/msfree.png
/var/www/html/usage/webalizer.png
#
```

3．配置 Webalizer

配置 webalizer 分为两个步骤：

- 配置 webalizer 的配置文件/etc/webalizer.conf。
- 配置 webalizer 的认证和授权。

Red Hat Linux 9 自带的 webalizer 的默认配置即可工作的很好。无需进行更多的配置，用户要了解/etc/webalizer.conf 的配置参数，可以使用 man 命令查看其手册页。

webalizer 的分析数据应该只能由管理员浏览，所以需要在 Apache 的主配置文件中进行认证和授权的配置。使用 7.4 节讲述的方法配置如下的指令：

```
<Directory "/var/www/html/usage">
    AuthType Basic
    AuthName "Admin"
    AuthUserFile /var/www/passwd/jamond
    AuthGroupFile /var/www/passwd/admingrp
    require group admin
</Directory>
```

修改配置之后重新启动 Apache。接着在客户浏览器上进行测试，经过身份认证之后可以看到如图 7-20 所示的页面。

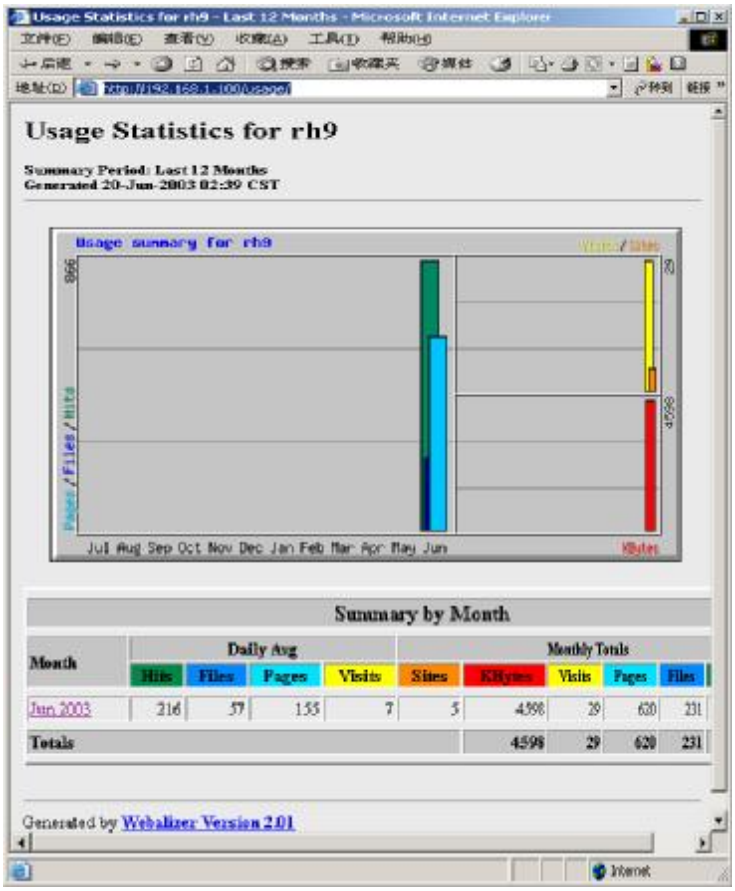

图 7-20　使用 webalizer 工具分析访问日志

单击页面下方的“Jun 2003”文字链接可以查看本月更多的日志分析信息。