



在第 4 章中找到对所有内核命令的深入和透彻的解释。

3.1.1 httpd.conf

httpd.conf 是主配置文件。它告诉服务器将如何运行。列表 3-1 显示了由 httpd.conf-dist 复制的缺省 httpd.conf。

列表 3-1: 由 httpd.conf-dist 复制的缺省 httpd.conf

```
# This is the main server configuration file. See URL
http://www.apache.org/
# for instructions.

# Do NOT simply read the instructions in here without
understanding
# what they do, if you are unsure consult the online docs. You
have been
# warned.

#Originally by Rob McCool

#ServerType is either inetd, or standalone.
ServerType standalone

# If you are running from inetd, go to "ServerAdmin".
#Port: The port the standalone listens to. For ports < 1023,
you will
# need httpd to be run as root initially.
Port 80

# HostnameLookups: Log the names of clients or just their IP numbers
# e.g. www.apache.org (on) or 204.62.129.132 (off)
# You should probably turn this off unless you are going to actually
# use the information in your logs, or with a CGI. Leaving
this on
# can slow down access to your site.
HostnameLookups on
# If you wish httpd to run as a different user or group, you
must run
# httpd as root initially and it will switch.

# User/Group: The name (or #number) of the user/group to run
httpd as.
# On SCO (ODT 3) use User nouser and Group nogroup
# On HPUX you may not be able to use shared memory as nobody,
and the
```

```
# suggested wordaround is to create a user www and use that
user.
User nobody
Group #-1

# The following directive disables keepalives and HTTP header
flushes for
# Netscape 2.x and browsers which spoof it. There are known
problems with
# these

BrowserMatch Mozilla/2 nokeepalive

# ServerAdmin: Your address, where problems with the server
should be
# e-mailed.

ServerAdmin you@your.address

# ServerRoot: The directory the server's config, error, and log
files
# are kept in

ServerRoot /usr/local/etc/httpd

# BindAddress: You can support virtual hosts with this option.
This option
# is used to tell the server which IP address to listen to. It
can either

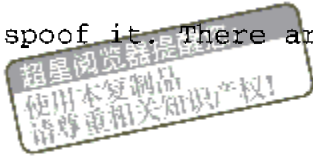
# contain "*", an IP address, or a fully qualified Internet
domain name.
# See also the VirtualHost directive.

# BindAddress *

# ErrorLog: The location of the error log file. If this does
not start
# with /, ServerRoot is prepended to it.

ErrorLog logs/error_log

# TransferLog: The location of the transfer log file. If this
does not
# start with /, ServerRoot is prepended to it.
```



```
TransferLog logs/access_log
# PidFile: The file the server should log its pid to
PidFile logs/httpd.pid

# ScoreBoardFile: File used to store internal server process
information.
# Not all architectures require this. But if yours does
(you'll know because
# this file is created when you run Apache) then you *must*
ensure that
# no two invocations of Apache share the same scoreboard file.
ScoreBoardFile logs/apache_status

# ServerName allows you to set a host name which is sent back
to clients for
# your server if it's different than the one the program would
get (i.e. use
# "www" instead of the host's real name).
#
# Note: You cannot just invent host names and hope they work.
The name you
# define here must be a valid DNS name for your host. If you
don't understand
# this, ask your network administrator.

# ServerName new.host.name

# CacheNegotiatedDocs: By default, Apache sends Proxym: no-
cache with each
# document that was negotiated on the basis of content. This
asks proxy
# servers not to cache the document. Uncommenting the following
line disables
# this behavior, and proxies will be allowed to cache the
documents.

# CacheNegotiatedDocs

# Timeout: The number of seconds before receives and sends time
out

Timeout 300

# KeepAlive: Whether or not to allow persistent connections
(more than
# one request per connection). Set to "Off" to deactivate.
```



KeepAlive On

MaxKeepAliveRequests: The maximum number of requests to allow
during a persistent connection. Set to 0 to allow an
unlimited amount.
We recommend you leave this number high, for maximum
performance.

MaxKeepAliveRequests 100

KeepAliveTimeout: Number of seconds to wait for the next
request

KeepAliveTimeout 15

Server-pool size regulation. Rather than making you guess
how many
server processes you need, Apache dynamically adapts to the
load it
sees --that is, it tries to maintain enough server processes
to
handle the current load, plus a few spare servers to handle
transient
load spikes (e.g., multiple simultaneous requests from a
single
Netscape browser).

It does this by periodically checking how many servers are
waiting
for a request. If there are fewer than MinSpareServers, it
creates

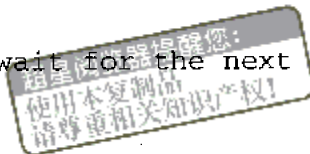
a new spare. If there are more than MaxSpareServers, some of
the
spares die off. These values are probably OK for most sites

MinSpareServers 5

MaxSpareServers 10

Number of servers to start --- should be a reasonable
ballpark figure.

StartServers 5



```
# Limit on total number of servers running, i.e., limit on the
number
# of clients who can simultaneously connect --- if this limit
is ever
# reached, clients will be LOCKED OUT, so it should NOT BE SET
TOO LOW.
# It is intended mainly as a brake to keep a runaway server
from taking
# Unix with it as it spirals down...
```

```
MaxClients 150
```

```
# MaxRequestsPerChild: the number of requests each child
process is
# allowed to process before the child dies.
# The child will exit so as to avoid problems after prolonged
use when
# Apache (and maybe the libraries it uses) leaks. On most
systems, this
# isn't really needed, but a few (such as Solaris) do have
notable leaks
# in the libraries.
```

```
MaxRequestsPerChild 30
```

```
# Proxy Server directives. Uncomment the following line to
# enable the proxy server:
# ProxyRequests On
```

```
# To enable the cache as well, edit and uncomment the following
lines:
```

```
# CacheRoot /usr/local/etc/httpd/proxy
# CacheSize 5
# CacheGcInterval 4
# CacheMaxExpire 24
# CacheLastModifiedFactor 0.1
# CacheDefaultExpire 1
# NoCache a_domain.com another_domain.edu joes.garage_sale.com
```

```
# Listen: Allows you to bind Apache to specific IP addresses
and/or
# ports, in addition to the default. See also the VirtualHost
command
```

```
#Listen 3000
```

```
#Listen 12.34.56.78:80

# VirtualHost: Allows the daemon to respond to requests for
more than one
# server address, if your server machine is configured to
accept IP packets
# for multiple addresses. This can be accomplished with the
ifconfig
# alias flag, or through kernel patches like VIF.

# Any httpd.conf or srm.conf directive may go into a
VirtualHost command.
# See also the BindAddress entry.

# <VirtualHost hots.some_domain.com>
# ServerAdmin webmaster@host.some_domain.com
# DocumentRoot /www/docs/host.some_domain.com
# ServerName host.some_domain.com
# ErrorLog logs/host.some_domain.com-error_log
# TransferLog logs/host.some_domain.com-access_log
# </VirtualHost>
```



注意本章的目标是用最小配置准备好服务器，并让它运行起来，因此，本章不提供所讨论三个配置文件的深入细节。

你需要配置的第一条命令是：

```
ServerType standalone | inetd
```

此命令指定如何运行 Web 服务器。可以用两种方法来运行服务器：standalone（独立的）和 inetd（由 inetd 运行的）。从功能的角度看，standalone 和 inetd 方法（我将在下面的各节讨论）几乎是相同的。但它们之间实际有很大区别，区别在于服务器的性能。一个由 inetd 运行的服务器进程在它结束对请求服务的同时立刻退出，而在 standalone 模式下，子 Web 服务器进程在退出之前要挂起一段时间，这就给它们提供了机会，可以重新用来服务新的请求。在 standalone 模式下，不存在对每个请求启动新进程的开销，所以它的效率更高。

然而，运行在 inetd 模式下也有一些优点。例如，运行在 inetd 模式下被认为比 standalone 模式更安全。很多系统管理员在 inetd 启动服务器进程之前，使用 TCP wrapper 程序来验证请求的有效性。如果你更关心安全，而且预期你的 Web 站点信息流量比较低，你可能会考虑将 ServerType 命令设置为 inetd，而不是 standalone。

让我们讨论这两种方法如何进行工作，并看一些常用的命令。

3.1.1.1 独立的 Apache 服务器

在 standalone 方法中（缺省设置），ServerType 命令被设置为 standalone。因此主 Web

服务器侦听特定端口的连接请求。当客户机器发出到特定端口地址的连接请求时，主服务器进程启动子 Web 服务器进程来服务该请求。图 3-1 表示了 standalone 方法。

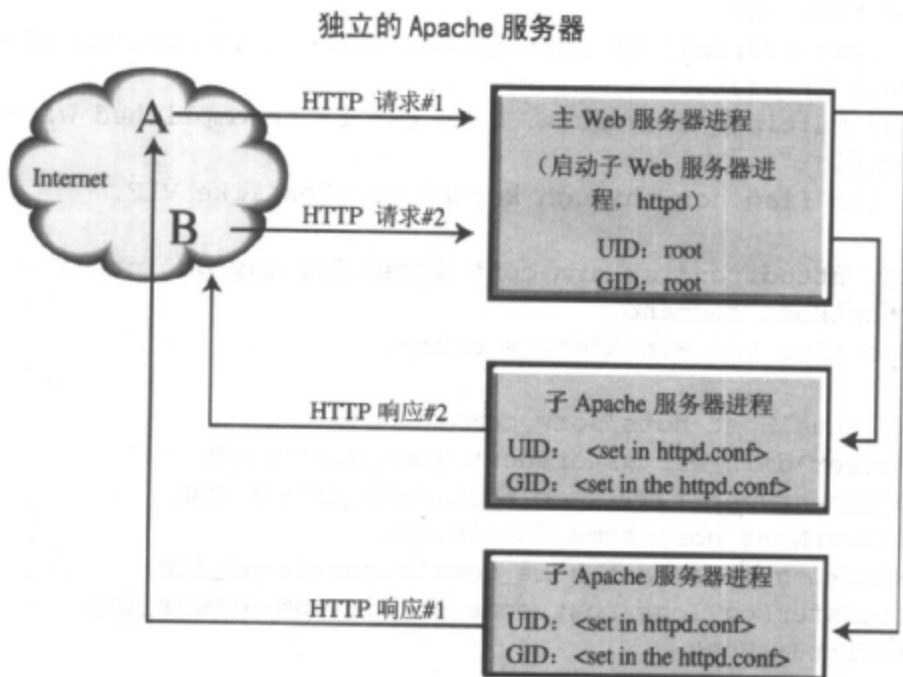


图 3-1 独立的服务器

1. 独立 Apache 服务器端口

用来告诉主服务器进程侦听特定端口地址的命令是：

Port [number]

缺省的 HTTP 端口是 80，而且它应当被用在典型的 Web 站点中。然而，如果你不是系统的 root 用户，并且想要运行该 Web 服务器，那么，你需要使用大于 1023 而小于 32768 的端口号。所有小于 1024 的端口都被认为是标准的保留端口，而要在这些端口上启动一项服务，需要 inetd 级（root 级）的权限。如果你需要在系统上用非 root 帐号试验 Web 服务器，可以使用上面提到的范围内的端口，只要它（端口）尚未被使用。如果试图使用已经被另一个服务器使用的端口地址，当启动该服务器时，你会得到错误信息。注意，如果你使用标准的 HTTP 端口 80 之外的任何端口，应该对所有到该服务器的 URL 请求提供端口号。例如，如果设置如下的命令：

Port 8080

则需要按如下方式请求在此服务器上的资源（例如，mypage.html 页）：

http://www.yourcompany.com:8080/mypage.html

如果决定使用 standalone 方法来运行 Apache 服务器，还需要告诉 Apache 它的用户名和组名是什么。

2. 独立 Apache 服务器的用户名和组名

用来告诉 Apache 用户名和组名的两条命令是：

```
User [username | #UID]
Group [group name | #GID]
```

由于安全原因，这两条命令非常重要。当主 Web 服务器进程启动子服务器进程来服务请求时，它根据为这些命令设置的值来改变子服务器进程的 UID 和 GID。参考图 3-1，侦听连接请求的主 Web 服务器进程是作为 root 用户进程运行的，而子进程是作为不同的用户/组进程运行的。如果子进程作为 root 用户进程来运行，那就打开了将遭受黑客攻击的潜在安全漏洞。允许与 root 用户进程进行对话的能力会使该系统中的潜在安全漏洞达到最严重的地步；因此，不建议这种做法。确切的说，我竭力建议将子服务器进程作为属于权限非常低的组的权限非常低的用户来运行。在大多数 UNIX 系统中，名为 nobody（通常 UID=-1）的用户和名为 nogroup（通常 GID=-1）的组是低权限的。你应该查阅/etc/group 和/etc/passwd 文件来确定这些设置。



如果计划将主 Web 服务器由非 root（普通）用户来运行，它将不能够改变子进程的 UID 和 GID，因为只有 root 用户进程才能改变其他进程的 UID 和 GID。因此，如果将主服务器由名为 foobar 的用户来运行，那么，所有的子进程将拥有与 foobar 相同的权限。与此类似，foobar 用户具有的任何组 ID 也将成为子进程的组 ID。

注意，如果计划使用数字格式表示用户和（或）组 ID，你需要在数字值前面插入一个#号，数字值可以在/etc/passwd 和/etc/group 文件中找到。

3.1.1.2 由 inetd 进程运行的 Apache 服务器

另外一种运行 Apache 服务器的方式是使用 inetd 进程。在此种情况下，ServerType 命令被设置为 inetd。这将改变整个过程。图 3-2 表示此种方式是如何工作的。inetd 是侦听所有小于 1024 的端口连接请求的 Internet 守护进程（一个服务器进程）。与前面的方法不同，inetd 控制哪些进程服务哪些请求。当客户系统发出到 Web 服务器的连接请求时，inetd 启动一个 Web 服务器进程，由此进程服务该请求，完成服务后即退出。

如果你选择通过 inetd 服务器来运行 Apache，你需要编辑/etc/inetd.conf 文件为 Apache 添加一个新记录。此文本文件有特殊的记录格式，你能够通过察看文件中的条目来确定该格式。除非你使用不寻常的 UNIX 系统，一般情况下，将会有使用以下记录定义的 inetd.conf 文件。

```
<service_name> <sock_type> <proto> <flags> <user> <server_path>
<args>
```

正如从上面行中所看到的，该项服务是由特殊用户来运行的。你需要决定 Apache 服务器由哪个用户来运行。最简单的方法是或者使用 nobody 用户，或者创建特殊的名为 httpd 的用户来运行服务器进程。如果对其他服务使用了 nobody 用户，那么，不要再将它用于 Apache。当使用 nobody 帐号的其他服务修改目录/文件设置时，重新将 nobody 用于 Web 服务可能会影响 Web 服务器可以访问的内容。建议创建特殊的 httpd 帐号，并按照如下方

由 inetd 运行的 Apache 服务器

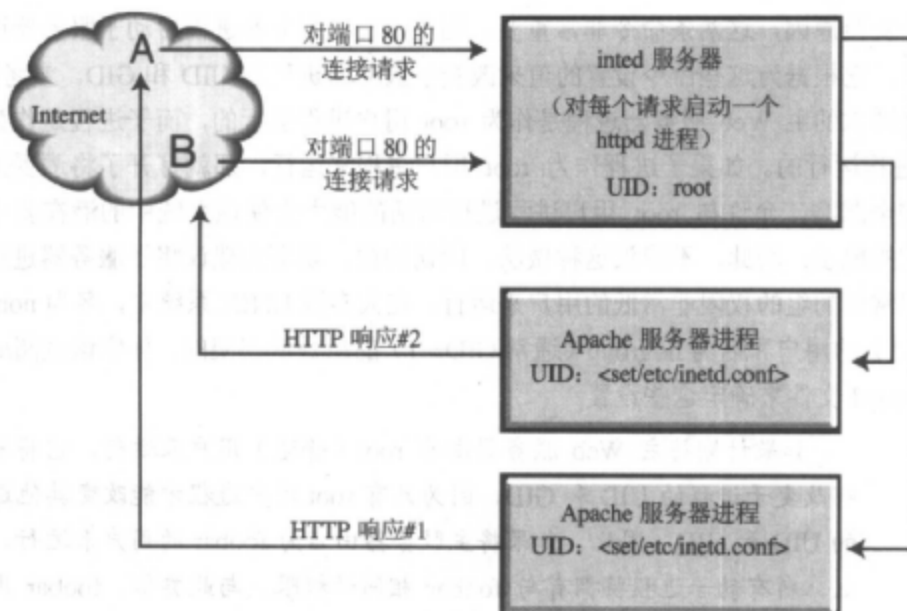


图 3-2 由 inetd 运行的服务器

式使用它：

```
httpd stream tcp nowait httpd /usr/sbin/httpd -f
/etc/httpd/conf/httpd.conf
```

一旦修改 `inetd.conf` 文件后，就需要修改 `/etc/services` 文件，该文件有如下所示的记录结构：

```
<service name> <port number>/<protocol name> <service entry in
inetd.conf>
```

于是，在 `/etc/services` 中要添加的行是：

```
httpd 80/tcp httpd
```

上述条目说明可以使用 `httpd` 服务，它由 `inetd` 服务器运行。并指出在端口 80 上使用 HTTP 服务。如果需要使用不同端口提供 Web (HTTP) 服务，请使用尚未被其他服务使用的端口号来替换 80。因为所有小于 1024 的端口号是为标准服务保留的，你需要使用大于 1024 而小于 32768 的端口地址（例如，8080）。

现在，需要重新启动 `inetd` 进程。首先，需要知道它的进程 ID (PID)。你可以使用下面的命令来获取该 ID：

```
ps auxw | grep inetd
```

注意，取决于你使用的 UNIX 系统，`ps` 实用程序使用的参数可能不同（如有必要，请参见 `ps` 手册页）。将 `ps` 的输出通过管道传送给 `grep` 实用程序，`grep` 在各行中搜索与关键字 `inetd` 匹配的任何行，并将它显示在屏幕（标准输出）上。尽管 `ps` 的输出格式在不同的系统上有所不同，通常在输出中的第一个数字列是进程 ID。现在，请按照如下所示

使用 kill 实用程序：

```
kill -HUP <PID of inetd>
```

不要忘记用实际的 inetd 进程 ID 替换<PID of inetd>部分。kill 实用程序将 HUP 信号发送给指定 PID 的进程。这将重新启动你的 inetd 服务器，并允许它重新读取被修改的配置文件。现在你的 inetd 配置就完成了。

一旦将 Apache 的 ServerType 命令指定为 inetd，并配置/etc/inetd.conf 和/etc/services 文件，在 httpd.conf 文件中的 User 和 Group 命令就不起作用了。然而，确保在/etc/inetd.conf 文件中使用的用户名对你的 Web 目录和存储该服务器日志文件的目录有访问权限。在本章的后面，你将定义 Web 目录以及日志文件的目录。

如果你的系统只有很少的空闲内存，或者预期你的 Web 站点信息流量不高，那么，建议对 Apache 使用 inetd 选项。

3.1.1.3 常用命令

standalone 和 inetd-run 方法共同使用一些常用命令。将讨论的第一条命令是无论何时当它发现问题时，它都会告诉服务器显示带有电子邮件地址的错误页面。这就给访问者一个机会，通过电子邮件报告错误。要将此命令设置为打开，请在下面的语法中使用负责维护该服务器的个人电子邮件地址：

```
ServerAdmin [email address]
```

另一个常用命令是 ServerRoot，它指定在何处保存服务器的配置、错误以及日志文件。它是所有与服务器相关文件的父目录。服务器的缺省目录是/usr/local/etc/httpd。如果你将服务器安装到了其他位置，请改变路径，以反映你的变动。它的语法是：

```
ServerRoot [fully qualified path name]
```

有两条命令指定服务器用来记录错误和访问信息的各个文件。第一条命令告诉服务器错误日志文件的路径和名称：

```
ErrorLog [fully qualified path name or error log file]
```

第二条命令说明访问信息日志文件的路径和名称：

```
TransferLog [fully qualified path name or access log file]
```

缺省的错误日志文件的位置和文件名是 logs/error_log，它的意思是，错误日志文件的名称是 error_log，它被存储在由 ServerRoot 指定目录下面的名为 logs 的子目录中。如果你希望将错误日志文件保存在其他位置，请输入该文件的完整路径，并确保该路径名以/（正斜杠）字符开始。类似的，缺省的访问信息日志文件存储在 logs 目录下，而它的名称是 access_log。

不管将日志文件保存在什么目录下，确保只有主服务器进程拥有对该目录的写权限。这是主要安全问题，因为，允许其他用户或进程写此日志目录可能意味着，某个未经授权的人会利用主 Web 服务器进程的 UID，而后者通常是 root 帐号。

另一个常用命令是 Apache 用来在一个文件中写它的主服务器进程 ID (PID) 的。按照下面方式来使用 PidFile 命令，它告诉 Apache 到哪里去写此文件：

```
PidFile [fully qualified path name of the pid file]
```