# LPI 考试准备: Web 服务

## 中级管理 (LPIC-2) 主题 208

David Mertz, Ph.D. (mertz@gnosis.cx)
开发人员
Gnosis Software

2006 年 6 月 01 日

本系列包括 7 个教程，介绍了 Linux 上的中级网络管理。本教程是第 4 部分，David Mertz 继续帮助您准备参加 Linux Professional Institute® 中级管理 (LPIC-2) 考试 208。这里，David Mertz 讨论了如何配置和运行 Apache HTTP 服务器及 Squid 代理服务器。

查看本系列更多内容

## 开始之前

了解这些教程讲授什么内容，以及如何从这些教程获得最大的收益。

### 关于本系列

Linux Professional Institute (LPI) 对 Linux 系统管理员进行认证分为两个级别：初级（也称为“1 级认证”）和中级（也称为“2 级认证”）。要获得 1 级认证，必须通过考试 101 和 102；要获得 2 级认证，必须通过考试 201 和 202。

developerWorks 提供教程来帮助您准备这 4 门考试。每门考试包含几个主题，每个主题在 developerWorks 上都有一个对应的自学教程。对于 LPI 202 考试，有下面 7 个主题和对应的 developerWorks 教程：

表 1. LPI 202 考试：教程和主题

| LPI 202 考试主题 | developerWorks 教程 | 教程摘要 |
|---|---|---|
| 主题 205 | LPI 202 考试准备（主题 205）：网络配置 | 学习如何配置基本的 TCP/IP 网络，从硬件层（通常是 Ethernet、modem、ISDN 或 802.11）到网络地址的路由。 |
| 主题 206 | LPI 202 考试准备（主题 206）：邮件和新闻 | 学习如何使用 Linux 作为邮件服务器和作为新闻服务器。了解邮件传输、本地邮件过滤、邮件列表维护软件和用于 NNTP 协议的服务器软件。 |
| 主题 207 | LPI 202 考试准备（主题 207）：DNS | 学习如何使用 Linux 作为 DNS 服务器，主要使用 BIND。了解如何执行基本的 BIND 配 |

| | | |
|---|---|---|
| | | 置、管理 DNS 区域和确保 DNS 服务器的安全。 |
| 主题 208 | LPI 202 考试准备（主题 208）：<br>Web 服务 | （本教程）学习如何安装和配置 Apache Web 服务器，以及如何实现 Squid 代理服务器。参见下面详细的 目标。 |
| 主题 210 | LPI 202 考试准备（主题 210）：<br>网络客户机管理 | 敬请期待！ |
| 主题 212 | LPI 202 考试准备（主题 212）：<br>系统安全性 | 敬请期待！ |
| 主题 214 | LPI 202 考试准备（主题 214）：<br>网络故障诊断 | 敬请期待！ |

要开始准备 1 级认证，请参考 针对 LPI 101 考试的 developerWorks 教程。要准备 2 级认证的其他考试，请参考 针对 LPI 201 考试的 developerWorks 教程。阅读 全套 developerWorks LPI 教程。

Linux Professional Institute 不为任何第三方考试准备资料或技术做担保。详情请联系 info@lpi.org。

## 关于本教程

欢迎阅读 “Web 服务”，这是介绍 Linux 上中级网络管理的 7 个教程中的第 4 个。在本教程中，您将学习如何配置及运行 Apache HTTP 服务器和 Squid Web Proxy Cache。

与 developerWorks 201 和 202 系列中的其他教程一样，本教程旨在充当考试准备的学习指南和入门，而不是该主题的完全文档。鼓励读者参考 LPI 的 详细目标列表 并在需要时参考其他资料补充这里的知识。

本教程是按照这个主题的 LPI 目标组织的。大致来说，权值越高的学习目标，在考试中出的题就越多。

表 2. Web 服务：本教程中涉及的考试目标

| LPI 考试目标 | 目标权值 | 目标摘要 |
|---|---|---|
| 2.208.1<br>实现 Web 服务器 | 权值 2 | 安装和配置 Web 服务器。该目标包括监视服务器的负载和性能，限制客户机用户的访问，为脚本语言如 modules 配置支持，以及设置客户机用户身份认证。另外还包括的能力是配置服务器选项以限制资源的使用。 |
| 2.208.2<br>维护 Web 服务器 | 权值 2 | 配置 Web 服务器以使用虚拟主机、Secure Sockets Layer (SSL) 和定制文件访问。 |
| 2.208.3<br>实现代理服务器 | 权值 2 | 安装和配置代理服务器，包括访问策略、身份认证和资源使用。 |

## 前提条件

要想从本教程获得最大的受益，您应该具备 Linux 的基础知识并且拥有一个可以用来实践本教程中介绍的命令的 Linux 系统。

# 关于 Apache 和 Squid

## Apache Web 服务器

总体来说，Apache 是 Internet 上的主流 Web 服务器，甚至是主流的 Linux 服务器。还有一些更加特殊用途的 Web 服务器可用（有些为特定的任务提供更高的性能），但是 Apache 总是默认的选择。

Apache 预安装在大多数 Linux 发行版上，并且在初始化期间，通常在启动之前就已经在运行了，即使您没有特别配置 Web 服务器。如果没有安装 Apache，那么使用发行版的普通安装系统安装它，或者从 Apache HTTP Server Project 下载最新的 HTTP 服务器。很多额外的功能是 modules 提供的，还有很多是 Apache 自己发布的，再有一些就是通过第三方得到的。

尽管最新版的 Apache 已于 2001 年达到了 2.x 级别，但是 Apache 1.3.x 还在广泛使用，并且 1.3.x 系列还继续维护着 bug 修复和安全更新。1.3 和 2.x 之间存在着细微的配置差异；一些 modules 对 1.3 可用，却对 2.x 不可用。在撰写本教程时，最新版是 1.3.34（稳定的）、2.0.55（稳定的）和 2.1.9（beta 版）。

通常来说，新服务器应该使用 2.x 系列中的最新稳定版。除非您特别需要不常见的较老的 module，2.x 提供了很好的稳定性、更多的功能和更好的整体性能（在有些任务中，比如 PHP 支持，1.3 仍然执行得较好）。随着时间的推移，新的特性无疑将在 2.x 中比在 1.3.x 中得到更好支持。

### Squid 代理服务器

Squid 是一个针对 Web 客户机的代理缓存服务器，支持 HTTP、FTP、TLS、SSL 和 HTTPS 协议。通过在本地网络上运行高速缓存，或者至少比所需的资源离您的网络更近，速度将会提高，所需的网络带宽将会减少。当相同的资源被由相同的 Squid 服务器为其服务的机器请求多次时，资源是从服务器本地副本交付的，而不需要请求通过多个网络路由器并潜在地减慢或超载目标服务器。

可以将 Squid 配置为必须配置在每个 Web 客户机（浏览器）中的显式代理，或者可以配置它来截取 LAN 外的所有 Web 请求，并缓存所有此类流量。可以用各种选项配置 Squid，比如缓存多长的页面，在什么条件下缓存页面。

### 其他资源

对于大多数 Linux 工具来说，通常有必要查看所讨论的任何实用工具的手册。实用工具或内核之间的版本和开关会有所不同，或者具有不同的 Linux 发行版。有关更深入的信息，Linux Documentation Project 具有很多有用的文档，尤其是它的 HOWTO 文档。已经出版了很多关于 Linux 网络的书籍，我觉得 O'Reilly 出版的由 Craig Hunt 编写的 TCP/IP Network Administration 一书相当有用。（到这些资源的连接请参见本教程后面的 参考资料。）

已经有很多很好的书籍讲述了使用 Apache。有些讲述一般的管理，而另一些则讲述 Apache 特定模块或特殊配置。到您喜欢的书商那里去询问这些已出版的书籍。

## 实现 Web 服务器

### 一堆守护程序

启动 Apache 类似于启动任何其他守护程序。通常，您想要将它的启动放入系统初始化脚本中，但是原则上您可以在任何时候启动 Apache。在大多数系统上，Apache 服务器叫做 httpd，但是也可能叫做 apache2。服务器可能安装在 /usr/sbin/ 中，但是取决于服务器的发行版和您如何安装它，也可能是其他位置。

大多数时候，您将不用任何选项启动 Apache，但是应该记住 `-d serverroot` 和 `-f config` 选项。第一个选项用于指定本地磁盘上的一个位置（本地磁盘是提供内容的地方）；第二个选项用于指定一个非默认的配置文件。使用 `ServerRoot` 指令，配置文件可能覆盖 `-f` 选项（通常会覆盖）。默认情况下，取决于编译选项，配置文件要么是 apache2.conf，要么是 httpd.conf。这些文件可能存储在 /

etc/apache2/、/etc/apache/、/etc/httpd/conf/、/etc/httpd/apache/conf 或一些其他位置上，这取决于版本、Linux 发行版以及如何安装或编译 Apache。检查 `man apache2` 或 `man httpd` 应该能够得到特定于系统的详细信息。

Apache 守护程序相对于其他服务器来说不太一样，因为它通常创建自己的几个运行副本。主副本只是衍生其他副本，而这些次副本则服务于实际的入站请求。具有多个运行副本的目的是充当成束进来的请求的 “池”；守护程序的其他副本根据几个配置参数而按需启动。主副本通常充当根，而其他副本出于安全原因充当一个更加受限制的用户。例如：

## 清单 1. Apache 的多个运行副本的许多方面

```
# ps axu | grep apache2
root      6620     Ss   Nov12   0:00 /usr/sbin/apache2 -k start -DSSL
www-data  6621     S    Nov12   0:00 /usr/sbin/apache2 -k start -DSSL
www-data  6622     Sl   Nov12   0:00 /usr/sbin/apache2 -k start -DSSL
www-data  6624     Sl   Nov12   0:00 /usr/sbin/apache2 -k start -DSSL
dqm        313     S+   03:44   0:00 man apache2
root       637     S+   03:59   0:00 grep apache2
```

在许多系统上，受限制的用户将是 nobody。在清单 1 中，它是 `www-data`。

## 包含配置文件

正如所提到的，Apache 的行为受到其配置文件中的指令的影响。对于 Apache2 系统，主配置文件可能在 /etc/apache2/apache2.conf 中，但是该文件通常包含多个 `Include` 语句，以从其他文件添加配置信息，可能会通过通配符模式。总体来说，一个 Apache 配置文件可能包含数百个指令和选项（大多数未在本教程中特别描述）。

一些文件很可能已经包含进来。可以参见 httpd.conf 获得用户设置，以利用以前的使用该名称的 Apache 1.3 配置文件。虚拟主机通常是在单独的配置文件中指定的，匹配一个通配符，如下所示：

## 清单 2. 指定虚拟主机

```
# Include the virtual host configurations:
Include /etc/apache2/sites-enabled/[^.#]*
```

对于 Apache 2.x，模块也通常是在单独的配置文件中指定的（通常是在 1.3.x 中的相同文件中）。例如，我的系统包含：

## 清单 3. 来自 /etc/apache2/apache2.conf

```
# Include module configuration:
Include /etc/apache2/mods-enabled/*.load
Include /etc/apache2/mods-enabled/*.conf
```

实际使用正在运行的 Apache 服务器中的模块需要配置文件中的两个步骤，即加载它和启用它：

## 清单 4. 加载可选的 Apache 模块

```
# cat /etc/apache2/mods-enabled/userdir.load
LoadModule userdir_module /usr/lib/apache2/modules/mod_userdir.so
# cat /etc/apache2/mods-enabled/userdir.conf
<IfModule mod_userdir.c>
    UserDir public_html
    UserDir disabled root

    <Directory /home/*/public_html>
        AllowOverride FileInfo AuthConfig Limit
        Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
    </Directory>
</IfModule>
```

`Include` 行中的通配符将把所有的 .load 和 .conf 文件插入 /etc/apache2/mods-enabled/ 目录。

注意一般模式：基本指令是具有一些选项的单行命令；较复杂的指令使用 XML 之类的起始/结束标记嵌套命令。您只需要知道每个指令是单行或起始/结束样式 —— 不能随意选择样式。

## 日志文件

一类重要的配置指令关注 Apache 操作的日志记录。您可以维护 Apache 操作的不同类型的信息和详细等级。保持一个错误日志总是一个不错的主意；您可以用单个指令指定它：

## 清单 5. 指定一个错误日志

```
# Global error log.
ErrorLog /var/log/apache2/error.log
```

您可以定制服务器访问、referrers 和其他信息等的其他日志，以满足自己个人的设置。一个日志操作是用两个指令配置的。首先，`LogFormat` 指令使用一组特殊变量来指定日志文件中将存储什么；其次，`CustomLog` 指令告诉 Apache 以指定格式实际记录事件。您可以指定无限多种格式，尽管不一定每一种都实际使用了。这允许您根据发展的需要，切换记录详细信息的开关。

`LogFormat` 中的变量类似于 shell 变量，但是具有一个前导的 `%`。有些变量具有单个字母，而另一些则具有由括号括住的长名称，如清单 5 所示。

## 清单 6. LogFormat 变量

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
CustomLog /var/log/apache2/referer_log combined
```

参考一本书或者完整的 Apache 文档，了解所有的变量。通常使用包含 `%h` 的变量来代表请求客户机的 IP 地址，`%t` 代表请求的日期时间，`%>s` 代表 HTTP 状态码，错误拼写的 `%{Referer}` 代表导致提供的页面的引用站点。

`LogFormat` 和 `CustomLog` 指令中使用的名称是任意的。在清单 6 中，使用了名称 `combined`，但是它也可以是 `myfoobarlog`。然而，有几个名称是经常使用的，出现在示例配置文件中，比如 `combined`、`common`、`referer` 和 `agent`。这些特定的格式通常受到日志分析器工具的直接支持。

# 维护 Web 服务器

## 虚拟主机、multi-homing 和 per-directory 选项

Apache 服务器提供的个别目录可能具有它们自己的配置选项。然而，主配置可能限制哪些选项可以本地配置。如果需要 per-directory 配置，那么使用 `AccessFileName` 指令并且一般指定本地配置文件名 .htaccess。本地配置的限制是在 `<code><Directory></code>` 指令中指定的。例如：

## 清单 7. Directory 指令的例子

```
#Let's have some Icons, shall we?
Alias /icons/ "/usr/share/apache2/icons/"
<Directory "/usr/share/apache2/icons">
      Options Indexes MultiViews
      AllowOverride None
      Order allow,deny
      Allow from all
</Directory>
```

通常利用 per-directory 选项，Apache 可以充当虚拟主机。多个域名可以从相同的 Apache 进程提供，每个域名访问一个适当的目录。可以用 `<code><VirtualHost></code>` 指令定义虚拟主机；将配置文件放置在一个被包含的目录中，比如 /etc/apache2/sites-enabled/，或者放置在一个主配置文件中。例如，您可以指定：

## 清单 8. 配置虚拟主机

```
      <VirtualHost "foo.example.com">
          ServerAdmin webmaster@foo.example.com
          DocumentRoot /var/www/foo
          ServerName foo.example.com
          <Directory /var/www/foo>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
          </Directory>
          ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
          <Directory "/usr/lib/cgi-bin">
                AllowOverride None
                Options ExecCGI -MultiViews +SymLinksIfOwnerMatch
                Order allow,deny
                Allow from all
          </Directory>
          CustomLog /var/log/apache2/foo_access.log combined
      </VirtualHost>
      <VirtualHost "bar.example.org">
          DocumentRoot /var/www/bar
          ServerName bar.example.org
      </VirtualHost>
      <VirtualHost *>
          DocumentRoot /var/www
      </VirtualHost>
```

最后的 `*` 选项挑出任何不指向显式指定的名称（比如 IP 地址寻址的名称或者寻址为也解析到服务器的未指定符号域）的 HTTP 请求。要虚拟域工作，DNS 必须用一个 CNAME 记录定义每个别名。

多引导的服务器似乎类似于虚拟主机化，但是概念上是不同的。使用 multi-homing，您可以为了允许 Web 请求而指定机器连接到的 IP 地址。例如，您可能只向本地 LAN 而不向外部提供 HTTP 访

问。如果指定一个监听的地址，也指示了一个非默认的端口。`BindAddress` 的默认值是 `*`，这意味着接受每个 IP 地址上的请求（服务器达到该地址）。一个混合的例子如下所示：

### 清单 9. 配置 multi-homing

```
BindAddress 192.168.2.2
Listen 192.168.2.2:8000
Listen 64.41.64.172:8080
```

在本例中，我将在默认端口 80 和特殊端口 8000 上接受来自本地 LAN的所有客户机请求（使用 192.168.2.2 地址）。该 Apache 安装也将监视来自 WAN 地址的客户机 HTTP 请求，但是只在端口 8080 上接受请求。

### 限制 Web 访问

您可以在 `<code><Directory></code>` 指令中使用 `Order`、`Allow from` 和 `Deny from` 命令来启用 per-directory 服务器访问。拒绝或允许的地址可由完整或部分主机名或 IP 地址来指定。`Order` 用于在接受列表和拒绝列表之间给出优先权。

在很多情况下，需要更加调优的控制，而不仅仅是通过简单地允许特定主机访问您的 Web 服务器而得到的控制。要启用用户登录要求，可使用 `Auth*` 系列命令，该命令还是在 `<Directory>` 指令中。例如，要设置 Basic Authentication，可使用清单 10 所示的指令。

### 清单 10. 配置 Basic Authentication

```
      <Directory "/var/www/baz">
          AuthName "Baz"
          AuthType Basic
          AuthUserFile /etc/apache2/http.passwords
          AuthGroupFile /etc/apache2/http.groups
          Require john jill sally bob
      </Directory>
```

也可以在 per-directory .htaccess 文件中指定 Basic Authentication。Digest Authentication 比 Basic 更加安全，但是 Digest Authentication 在浏览器中不太广泛实现。然而，Basic 的弱点（以明文传输密码）用一个 SSL 层已较好地解决了。

对 Web 流量的 SSL 加密的支持由模块 `mod_ssl` 提供。当使用 SSL 时，服务器和客户机之间传输的数据用一个防止被截取的动态协商的密码进行了加密。所有主要的浏览器都支持 SSL。有关用 `mod_ssl` 配置 Apache 2.x 的更多信息，请参见 Apache Web 站点上的描述（参见 参考资料 中的链接）。

## 实现代理服务器

### 安装和运行 Squid

在大多数发行版中，可以使用通常的安装过程安装 Squid。从 Squid Web Proxy Cache Web 站点（参见 参考资料 中的链接）获得 Squid 的源代码版本。从源代码进行构建需要使用基本的 `./configure; make; make install` 序列。

安装好之后，就可以简单地将它运行为根 /usr/sbin/squid（或者您的发行版所使用的任何位置，也许是 /usr/local/sbin/）。当然，要做更多有用的事情，您将需要在 /etc/squid/squid.conf 中配置

Squid 配置文件 /usr/local/squid/etc/squid.conf，或者更准确地说，您的系统位于 squid.conf 中。与大多数守护程序一样，您可以使用一个不同的配置文件，在本例中利用 `-f` 选项。

## 端口、IP 地址、http_access 和 ACL

Squid 最重要的配置选项是您选择的 `http_port` 选项。您可以监视您希望监视的任何端口，可选地将每个端口附加到一个特定的 IP 地址或主机名。Squid 的默认端口是 3128，允许连接到 Squid 服务器的任何 IP 地址。要只为 LAN 缓存 Squid，而是指定本地 IP 地址，如下所示：

## 清单 11. 只为 LAN 缓存 Squid

```
# default (disabled)
# http_port 3128
# LAN only
http_port 192.168.2.2:3128
```

也可以使用 `icp_port` 和 `htcp_port` 通过其他 Squid 服务器启用缓存。IPC 和 HTCP 协议用于缓存，以在它们自己之间而不是通过 Web 服务器和客户机本身进行通信。要缓存组播通信，可使用 `mcast_groups`。

要让客户机连接到您的 Squid 服务器，您需要给予它们这样的权限。与 Web 服务器不一样，Squid 对它的资源不是很大方。在简单的情况下，我们可以只使用一组子网/网络掩码或 CIDR (Classless Internet Domain Routing) 模式来控制权限：

## 清单 12. 简单的 Squid 访问权限

```
http_access deny 10.0.1.0/255.255.255.0
http_access allow 10.0.0.0/8
icp_access allow 10.0.0.0/8
```

可以使用 `acl` 指令来命名访问控制列表（access control list，ACL）。可以命名像清单 12 中一样只指定地址范围的 `src` ACL，但是也可以创建其他类型的 ACL。例如：

## 清单 13. 经过调优的访问权限

```
acl mynetwork     src             192.168/16
acl asp           urlpath_regex   \.asp$
acl bad_ports     port            70 873
acl javascript    rep_mime_type -i ^application/x-javascript$
# what HTTP access to allow classes
http_access deny asp          # don't cache active server pages
http_access deny bad_ports    # don't cache gopher or rsync
http_access deny javascript   # don't cache javascript content
http_access allow mynetwork   # allow LAN everything not denied
```

清单 13 只展示了可用 ACL 类型的一个很小的子集。参见一个示例 squid.conf，了解许多其他类型的例子。或者参见 Squid User's Guide 中的第 6 章 “Access control documentation”（参见 参考资料 中的链接）。

在清单 13 中，我们决定不缓存以 .asp 结尾的 URL（可能是动态内容），不缓存端口 70 和 873，也不缓存返回的 JavaScript 对象。与被拒绝的不一样，LAN 上的机器（给出了 /16 范围）将缓存它们的

所有请求。注意，定义的每个 ACL 具有一个惟一而任意的名称（使用有意义的名称；Squid 不保留名称）。

## 缓存模式

运行 Squid 最简单的方式是代理模式。如果这样做，那么客户机必须被显式地配置为使用缓存。Web 浏览器客户机具有配置屏幕，以允许它们指定代理地址和端口而不是直接的 HTTP 连接。该设置使得配置 Squid 非常简单，但是也使得客户机要受益于 Squid 缓存则需要进行一些设置。

也可以配置 Squid 运行为透明的缓存。为此，您需要配置基于策略的路由（在 Squid 本身的外面，使用 `ipchains` 或 `ipfilter` 进行配置）或者使用您的 Squid 服务器作为网关。假设您可以通过 Squid 服务器定向外部请求，那么 Squid 需要进行如下配置。您也许需要用 `--enable-ipf-transparent` 选项重新编译 Squid；然而，在大多数 Linux 安装中，这项工作应该已经完成了。要为透明的缓存配置服务器（在它得到重定向的数据包之后），可将清单 14 所示的内容添加到 squid.conf 中：

## 清单 14. 为透明的缓存配置 Squid

```
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

# 参考资料

## 学习

- 您可以参阅本文在 developerWorks 全球站点上的 英文原文。
- 复习 developerWorks 上的整个 LPI 考试准备教程系列，学习 Linux 基本知识，准备系统管理员认证。
- 在 LPIC Program 处可以找到三级 Linux Professional Institute 的 Linux 系统管理认证的任务列表、样例问题和详细的目标。
- 获得有关 configuring Apache 2.x with mod_ssl 的更多信息。
- 阅读教程 “Customizing Apache for maximum performance”（developerWorks，2002 年 6 月），了解如何为满足特定的环境和需求而调整 Apache。
- Squid User's Guide 中有关 Access Control and Access Control Operators 的一章，描述了可用的 ACL 类型。
- Craig Hunt 编写的 TCP/IP Network Administration, Third Edition（O'Reilly，2002 年 4 月），是有关 Linux 网络的优秀资源。
- Linux Users Groups WorldWide 主页列出了全世界 700 个 Linux 用户组。许多 LUG 具有 LPI 考试的本地和远程学习组。
- Linux Documentation Project 具有大量有用的文档，特别是 HOWTO 文档。
- 在 developerWorks Linux 专区 中，找到更多针对 Linux 开发人员的资源。
- 随时关注 developerWorks 技术事件和网络广播。

## 获得产品和技术

- 下载最新版 Apache Web 服务器。
- 下载 Squid 和附加的 Squid 文档。
- 利用可直接从 developerWorks 下载的 IBM 试用软件 构建您的下一个 Linux 开发项目。

## 讨论

- 参与与本文档相关的论坛讨论。
- 查看 developerWorks blogs 并加入 developerWorks 社区。

# 关于作者

David Mertz, Ph.D.

David Mertz 自 2000 年以来已经撰写了 developerWorks 专栏 Charming Python 和 XML Matters。阅读他的书籍 Text Processing in Python。要更多地了解 David，请参见他的个人网页。