第五部分

内容发布与分发

第五部分讲述了 Web 内容发布和传播的各种技术。

- 第 18 章介绍了在现代的 Web 托管环境中部署服务器的若干方法，HTTP 对虚拟 Web 托管的支持以及如何在地理上相距遥远的服务器之间复制内容。
- 第 19 章讨论了创建 Web 内容并将其放置到 Web 服务器上去的各种技术。
- 第 20 章探讨了各种将来访的 Web 流量分发到一组服务器上的技术和工具。
- 第 21 章解释了日志的各种格式和各种常见问题。

408 ~ 410

第18章

Web主机托管

当你把资源放在公共的 Web 服务器上时，因特网社区就可以使用它们了。这些资源可以是简单的文本文件或图像，也可以是复杂的实时导航地图或电子商务购物网关。能够将这些由不同组织拥有的种类繁多的资源便利地发布到网站上，并将其放置在能以合理价格提供很好性能的 Web 服务器上，是很关键的。

对内容资源的存储、协调以及管理的职责统称为 Web *主机托管*。主机托管是 Web 服务器的主要功能之一。保存并提供内容，记录对内容的访问以及管理内容都离不开服务器。如果不想自行管理服务器所需的软硬件，就需要主机托管服务，即*托管者*。托管者出租服务和网站管理维护业务，并提供各种不同程度的安全级别、报告及易用性。托管者通常把很多网站放在一些强大的 Web 服务器上联合运行，这样可以获得更高的成本效益、可靠性和性能。

本章讲解 Web 主机托管服务中的某些重要特征和它们如何与 HTTP 应用程序交互。本章的主要内容包括：

- 不同的网站如何被“虚拟地托管”在同一个服务器上，这样会对 HTTP 产生怎样的影响；
- 在很大的流量压力下，如何确保网站更可靠；
- 如何使网站加载更快。

18.1 主机托管服务

在万维网的早期，每个组织自行购买自己的计算机硬件，搭建自己的计算机房，申请自己的网络连接，并管理自己的 Web 服务器软件。

随着 Web 迅速成为主流，每人都想要一个网站，但很少有人有能力或时间来搭建带
411 空调的服务器机房，注册域名，或购买网络带宽。为了满足人们的迫切需求，出现了很多新的企业，提供了专业化管理的 Web 主机托管服务。服务级别有多种，从物理上的设备管理（提供空间、空调以及线缆）到完整的 Web 主机托管，顾客只需要提供内容就行了。

本章主要探讨托管 Web 服务器要提供什么服务。网站运作需要的很多东西（例如，它支持不同语言的能力和进行安全的电子商务交易的能力）都取决于托管 Web 服务器提供的功能。

简单例子——专用托管

假设 Joe 的五金商店和 Mary 的古董拍卖店都需要大容量的网站。Irene 网络服务提供商那里有很多机架，机架上全是一样的高性能 Web 服务器，可以租给 Joe 和 Mary，

这样，他俩就不用自行购买自己的服务器并管理服务器软件了。

在图 18-1 中，Joe 和 Mary 都签约使用 Irene 的网络服务提供商提供的专用 Web 托管服务。Joe 租了专用的 Web 服务器，该服务器是 Irene 网络服务提供商购买和维护的。Mary 也从 Irene 网络服务提供商那里租了另一个专用服务器。Irene 网络服务提供商大批量地购买服务器硬件，它们选择的硬件经久耐用且相对便宜。如果 Joe 或 Mary 的网站变得更受欢迎，Irene 网络服务提供商可以立刻给 Joe 或 Mary 提供更多的服务器。

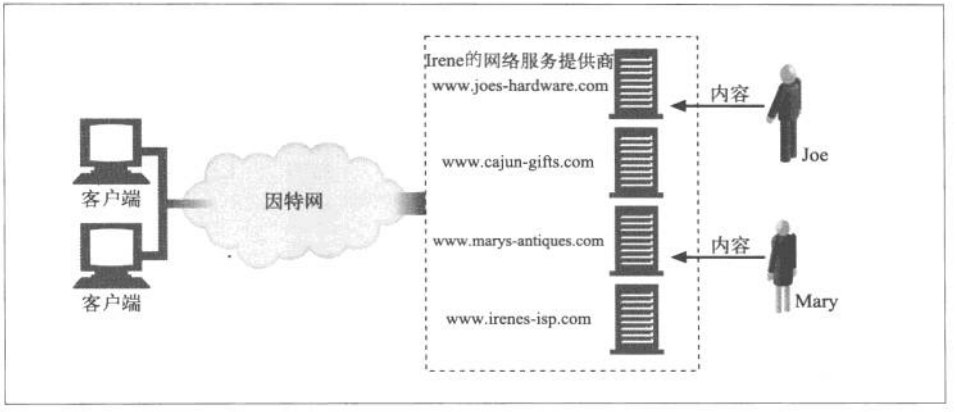


图 18-1　外包的专用托管服务

在这个例子中，浏览器向 Joe 服务器的 IP 地址发送对 www.joes-hardware.com 的 HTTP 请求，向 Mary 服务器（不同于 Joe）的 IP 地址发送对 www.marys-antiques.com 的请求。

412

18.2　虚拟主机托管

许多人想要在 Web 上展现自己，但他们的网站流量都不大。对这些人来说，使用专用的 Web 服务器可能有点儿大材小用，因为他们每月花费数百美元租来的服务器大部分时间都是空闲的！

许多 Web 托管者通过让一些顾客共享一台计算机来提供便宜的 Web 主机托管服务。这称为*共享主机托管*或*虚拟主机托管*。每个网站看起来是托管在不同的服务器上，但实际上是托管在同一个物理服务器上。从最终用户的角度来看，被虚拟托管的网站应当和托管在专用服务器上的网站没什么区别。

从成本效益、空间以及管理方面考虑，提供虚拟主机托管的公司希望能在同一个

服务器上托管数十、上百，甚至上千个网站——但这不一定意味着上千个网站是用一台 PC 机来提供服务的。托管者可以创建成排同样的服务器，称为**服务器集群**（server farm），把负载分摊在群里的服务器上。因为群里的每台服务器都一样，并且托管了许多虚拟网站，所以管理起来更加方便。（我们将在第 20 章更详细地介绍服务器集群。）

当 Joe 和 Mary 刚开始商务运作时，他们可能会选择虚拟主机托管，以节省费用，直到他们网站的流量规模达到值得使用专用服务器的水平为止（参见图 18-2）。

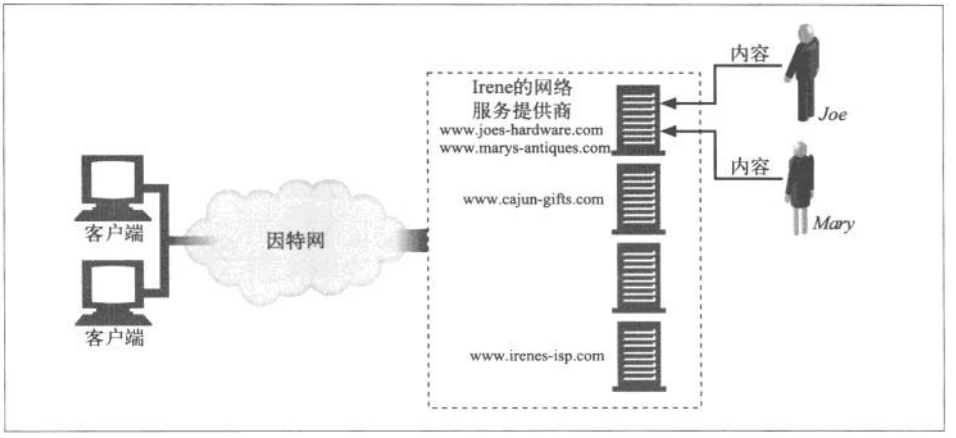


图 18-2　外包的虚拟主机托管

18.2.1　虚拟服务器请求缺乏主机信息

不幸的是，HTTP/1.0 中的一个设计缺陷会使虚拟主机托管者抓狂。HTTP/1.0 规范
413 中没有为共享的 Web 服务器提供任何方法来识别要访问的是哪一个托管的网站。

回想一下，HTTP/1.0 请求在报文中只发送了 URL 的路径部分。如果要访问 http://www.joes-hardware.com/index.html，浏览器会连接到服务器 www.joes-hardware.com，但 HTTP/1.0 请求中只提到 GET /index.html，没有提到主机名。如果服务器虚拟托管了多个站点，就没有足够的信息能指出要访问的是哪个虚拟网站。图 18-3 就是这样的一个示例。

- 如果客户端 A 试图访问 http://www.joes-hardware.com/index.html，请求 GET /index.html 将被发送到共享的 Web 服务器。
- 如果客户端 B 试图访问 http://www.marys-antiques.com/index.html，同样的请求 GET /index.html 也将被发送到共享的 Web 服务器。

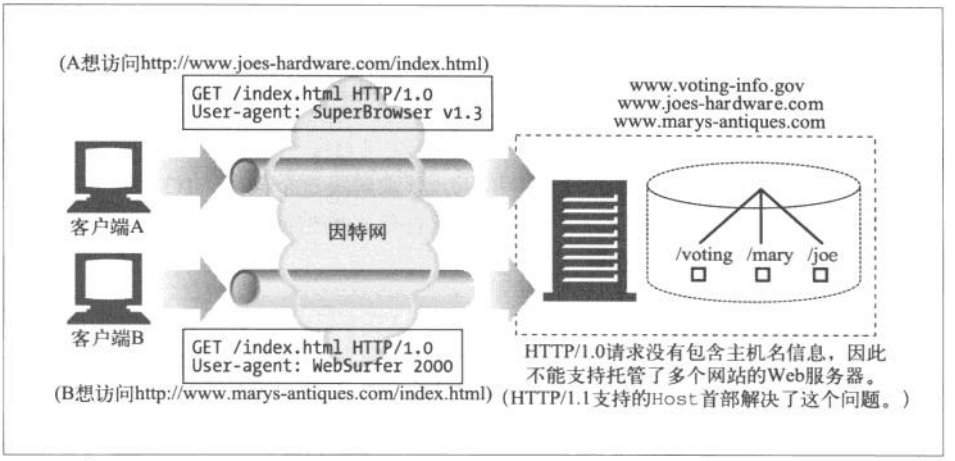


图 18-3 HTTP/1.0 服务器请求中没有主机名信息

就 Web 服务器而言，没有足够的信息可供其判断究竟要访问的是哪个网站。尽管请求的是完全不同的文档（来自不同的网站），但这两个请求看起来是一样的，这是因为网站的主机信息已经从请求中剥离了。

我们已经在第 6 章中介绍过，HTTP 替代物（反向代理）和拦截代理也都需要明确的站点信息。

18.2.2 设法让虚拟主机托管正常工作

缺失的主机信息是原始 HTTP 规范的疏忽，它错误地假设了每个 Web 服务器上只托管了一个网站。HTTP 的设计者没有为进行虚拟主机托管的共享服务器提供支持。正因为如此，URL 中的主机名信息被当作冗余信息剥离了，只要求发送路径部分。

因为早期的规范没有考虑到虚拟主机托管，Web 托管者需要开发变通的方案和约定来支持共享的虚拟主机托管。这个问题本可以通过要求所有 HTTP 请求报文发送完整的 URL 而不只是路径部分来简单地解决。而 HTTP/1.1 的确要求服务器能够处理 HTTP 报文请求行上的完整 URL，但将现存的应用程序都升级到这个规范还需要很长时间。在此期间，涌现了以下 4 种技术。 414

- *通过 URL 路径进行虚拟主机托管*
  在 URL 中增添专门的路径部分，以便服务器判断是哪个网站。

- *通过端口号进行主机托管*
  为每个站点分配不同的端口号，这样请求就由 Web 服务器的单独实例来处理。

- 通过 IP 地址进行主机托管
  为不同的虚拟站点分配专门的 IP 地址，把这些地址都绑定到一台单独的机器上。这样，Web 服务器就可以通过 IP 地址来识别网站名了。

- 通过 Host 首部进行主机托管
  很多 Web 托管者向 HTTP 的设计者施压，要求解决这个问题。HTTP/1.0 的增强版和 HTTP/1.1 的正式版定义了 Host 请求首部来携带网站名称。Web 服务器可以通过 Host 首部识别虚拟站点。

接下来详细介绍每种技术。

1. 通过URL路径进行虚拟主机托管

可以通过分配不同的 URL 路径，用这种笨方法把共享服务器上的虚拟站点隔离开。例如，可以给每个逻辑网站一个专门的路径前缀。

- Joe 的五金商店可以是：http://www.joes-hardware.com/joe/index.html。
- Mary 的古董拍卖店可以是：http://www.marys-antiques.com/mary/index.html。

当请求到达服务器时，其中并没有主机名信息，但服务器可以通过路径来区分它们。

- 请求 Joe 的五金商店的网址是 GET /joe/index.html。
- 请求 Mary 的古董拍卖店的网址是 GET /mary/index.html。

这不是一个好办法。/joe 和 /mary 这样的前缀是多余的（主机名中已经提到 joe 了）。更糟的是，描述主页链接的常见约定：http://www.joes-hardware.com 或 http://www.joes-hardware.com/index.html 都不能用了。

总之，按 URL 来进行虚拟主机托管是一个糟糕的解决方案，很少会用到。

2. 通过端口号进行虚拟主机托管

除了修改路径名，还可以在 Web 服务器上为 Joe 和 Mary 的网站分配不同的端口号。
415 不再使用端口 80，而是采用其他端口号，例如，Joe 用 82 Mary 用 83。但这个解决方案也有同样的问题：终端用户不会乐意在 URL 中指定非标准的端口号。

3. 通过IP地址进行虚拟主机托管

一个更常用的、更好的方法是通过 IP 地址进行虚拟化。每个虚拟网站都分配一个或多个唯一的 IP 地址。所有虚拟网站的 IP 地址都绑定到同一个共享的服务器上。服务器可以查询 HTTP 连接的目的 IP 地址，并以此来判断客户端的目标网站。

比方说，托管者把 IP 地址 209.172.34.3 分配给 www.joes-hardware.com，把 IP 地址 209.172.34.4 分配给 www.marys-antiques.com，把这两个 IP 地址都绑定到同一个物理服务器上。Web 服务器就能使用目的 IP 地址来识别用户请求的是哪个虚拟站点了，参见图 18-4。

- 客户端 A 获取 http://www.joes-hardware.com/index.html。
- 客户端 A 查询 www.joes-hardware.com 的 IP 地址，得到 209.172.34.3。
- 客户端 A 打开到共享服务器的 TCP 连接，目的地址是 209.172.34.3。
- 客户端 A 发送请求，内容为 GET /index.html HTTP/1.0。
- 在 Web 服务器提供响应之前，它注意到实际的目的 IP 地址（209.172.34.3），判断出这是 Joe 的五金网站的虚拟 IP 地址，就根据子目录 /joe 来完成请求。返回的是文件 /joe/index.html。

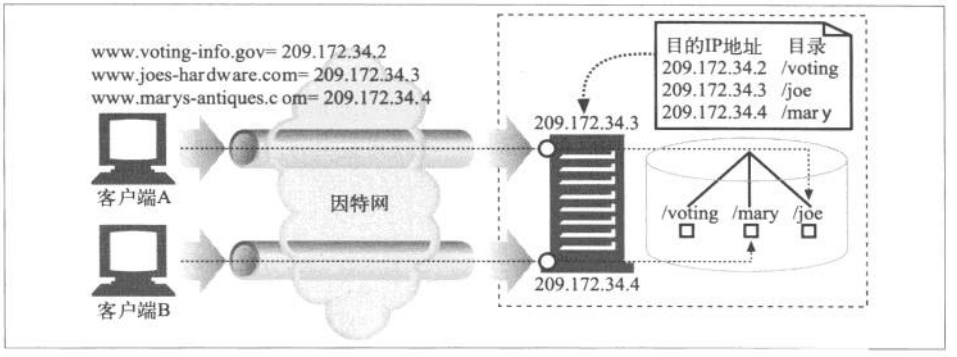


图 18-4 虚拟 IP 的主机托管

类似地，如果客户端 B 请求 http://www.marys-antiques.com/index.html。

- 客户端 B 查询 www.marys-antiques.com 的 IP 地址，得到 209.172.34.4。
- 客户端 B 打开到 Web 服务器的 TCP 连接，目的地址是 209.172.34.4。
- 客户端 B 发送请求，内容是 GET /index.html HTTP/1.0。
- Web 服务器判断出 209.172.34.4 是 Mary 的网站，根据 /mary 目录来完成请求，返回的是文件 /mary/index.html。

416

对大的托管者来说，虚拟 IP 的主机托管能够工作，但它会带来一些麻烦。

- 在计算机系统上能绑定的虚拟 IP 地址通常是有限制的。想在共享的服务器上托管成百上千的虚拟站点的服务商不一定能实现愿望。
- IP 地址是稀缺资源。有很多虚拟站点的托管者不一定能为被托管的网站获取足够多的 IP 地址。

- 托管者通过复制服务器来增加容量时，IP 地址短缺的问题就更严重了。随负载均衡体系的不同，可能会要求每个复制的服务器上有不同的虚拟 IP 地址，因此 IP 地址的需求量可能会随复制服务器的数量而倍增。

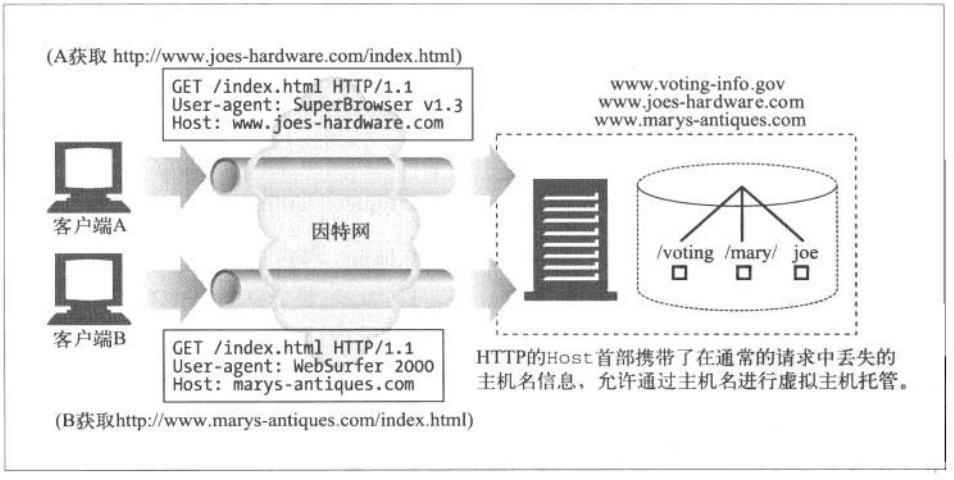
尽管虚拟 IP 的主机托管存在消耗地址的问题，但它仍然得到了广泛的运用。

4. 通过Host首部进行虚拟主机托管

为了避免过度的地址消耗和虚拟 IP 地址的限制，我们希望在虚拟站点间共享同一个 IP 地址，且仍能区分站点。但正如我们看到的那样，因为大多数浏览器只是把 URL 的路径发给服务器，关键的虚拟主机名信息被其丢掉了。

为了解决这个问题，浏览器和服务器的实现者扩展了 HTTP，把原始的主机名提供给服务器。不过，浏览器不能只发送完整的 URL，因为这会使许多只能接收路径的服务器无法工作。替代的方法是，把主机名（和端口号）放在所有请求的 Host 扩展首部中传送。

在图 18-5 中，客户端 A 和客户端 B 都发送了携带有要访问的原始主机名的 Host 首部。当服务器收到对 /index.html 的请求时，可以通过 Host 首部来判断要使用哪个资源。



417 图 18-5 用 Host 首部区分请求的虚拟主机

Host 首部最早是在 HTTP/1.0+ 中引入的，它是开发商实现的 HTTP/1.0 的扩展超集。遵循 HTTP/1.1 标准则必须支持 Host 首部。绝大多数现代浏览器和服务器都支持 Host 首部，但仍有一些客户端和服务器（以及网络机器人）不支持它。

18.2.3 HTTP/1.1的Host首部

Host 首部是 HTTP/1.1 的请求首部，定义在 RFC 2068 中。由于虚拟服务器的流行，绝大多数 HTTP 客户端（即使是不遵循 HTTP/1.1 的客户端），都实现了 Host 首部。

1. 语法与用法

Host 首部描述了所请求的资源所在的因特网主机和端口号，和原始的 URL 中得到的一样：

```
Host = "Host" ":" host [ ":" port ]
```

但要注意以下问题。

- 如果 Host 首部不包含端口，就使用地址方案中默认的端口。
- 如果 URL 中包含 IP 地址，Host 首部就应当包含同样的地址。
- 如果 URL 中包含主机名，Host 首部就必须包含同样的名字。
- 如果 URL 中包含主机名，Host 首部就不应当包含 URL 中这个主机名对应的 IP 地址，因为这样会扰乱虚拟主机托管服务器的工作，它在同一个 IP 地址上堆叠了很多虚拟站点。
- 如果 URL 中包含主机名，Host 首部就不应当包含这个主机名的其他别名，因为这样也会扰乱虚拟主机托管服务器的工作。
- 如果客户端显式地使用代理服务器，客户端就必须把原始服务器，而不是代理服务器的名字和端口放在 Host 首部中。以往，若干个 Web 客户端在启用客户端代理设置时，错误地把发出的 Host 首部设置成代理的主机名。这种错误行为会使代理和原始服务器都无法正常处理请求。
- Web 客户端必须在所有请求报文中包含 Host 首部。
- Web 代理必须在转发请求报文之前，添加 Host 首部。
- HTTP/1.1 的 Web 服务器必须用 400 状态码来响应所有缺少 Host 首部字段的 HTTP/1.1 请求报文。

下面是一段简单的 HTTP 请求报文，用于获取 www.joes-hardware.com 的主页，其中带有必须的 Host 首部字段：

418

```
GET http://www.joes-hardware.com/index.html HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.51 [en] (X11; U; IRIX 6.2 IP22)
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/
png, */*
Accept-Encoding: gzip
Accept-Language: en
Host: www.joes-hardware.com
```

2. 缺失Host首部

有少量在用的老式浏览器不会发送 Host 首部。如果某个虚拟主机托管服务器使用 Host 首部来判断所服务的是哪个网站，而报文中没有出现 Host 首部的话，那它可能会把用户导向某个默认的 Web 页面（例如网络服务提供商的 Web 页面），也可能返回一个错误页面建议用户升级浏览器。

3. 解释Host首部

对于没有进行虚拟主机托管，而且不允许资源随请求主机的不同而变化的原始服务器来说，可以忽略 Host 首部字段的值。但资源会随主机名的不同而变化的原始服务器，都必须在一条 HTTP/1.1 请求判断其所请求的资源时使用下列规则。

（1）如果 HTTP 请求报文中的 URL 是绝对的（也就是说，包含方案和主机部分），就忽略 Host 首部的值。
（2）如果 HTTP 请求报文中的 URL 没有主机部分，而该请求带有 Host 首部，则主机 / 端口的值就从 Host 首部中取。
（3）如果通过第（1）步或第（2）步都无法获得有效的主机，就向客户端返回 400 Bad Request 响应。

4. Host首部与代理

某些版本的浏览器发送的 Host 首部不正确，尤其是配置使用代理的时候。例如，配置使用代理时，一些老版本的 Apple 和 PointCast 客户端会错误地把代理的名字，而不是原始服务器的名字放在 Host 首部里发送。

18.3 使网站更可靠

在下面列出的这些时间段内，网站通常是无法运作的。

- 服务器宕机的时候。
- 交通拥堵：突然间很多人都要看某个特别的新闻广播或涌向某个大甩卖网店。突然的拥堵可以使 Web 服务器过载，降低其响应速度，甚至使它彻底停机。
- 网络中断或掉线。

419 本节会展示一些预判和处理这些常见问题的方法。

18.3.1 镜像的服务器集群

服务器集群是一排配置相同的 Web 服务器，互相可以替换。每个服务器上的内容可以通过镜像复制，这样当某个服务器出问题的时候，其他的可以顶上。

镜像的服务器常常组成层次化的关系。某个服务器可能充当“内容权威”——它含有原始内容（可能就是内容作者上传的那个服务器）。这个服务器称为**主原始服务器**（master origin server）。从主原始服务器接收内容的镜像服务器称为**复制原始服务器**（replica origin server）。一种简单的部署服务器集群的方法是用网络交换机把请求分发给服务器。托管在服务器上的每个网站的 IP 地址就设置为交换机的 IP 地址。

在图 18-6 显示的镜像服务器集群中，主原始服务器负责把内容发送给复制原始服务器。对集群外部来说，内容所在的 IP 地址就是交换机的 IP 地址。交换机负责把请求发送到服务器上去。

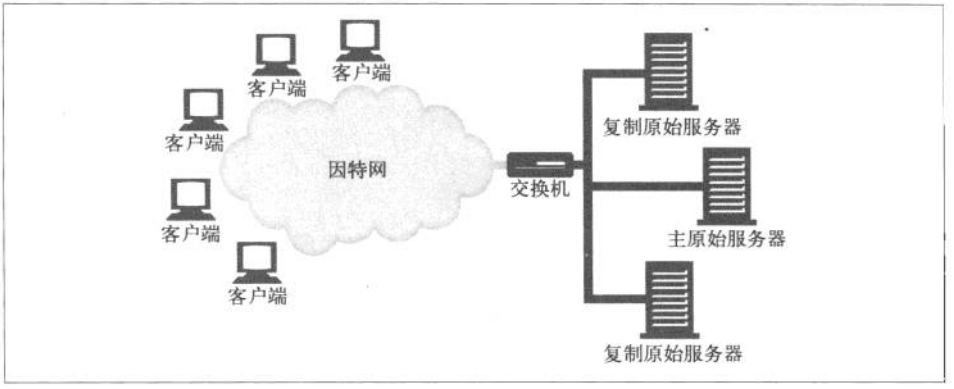


图 18-6　镜像的服务器集群

镜像 Web 服务器可以在不同的地点包含同样内容的副本。图 18-7 展示了 4 个镜像服务器，其中主服务器在芝加哥，复制服务器在纽约、迈阿密和小石城。主服务器为芝加哥地区的客户端服务，并肩负把内容传播给复制服务器的任务。

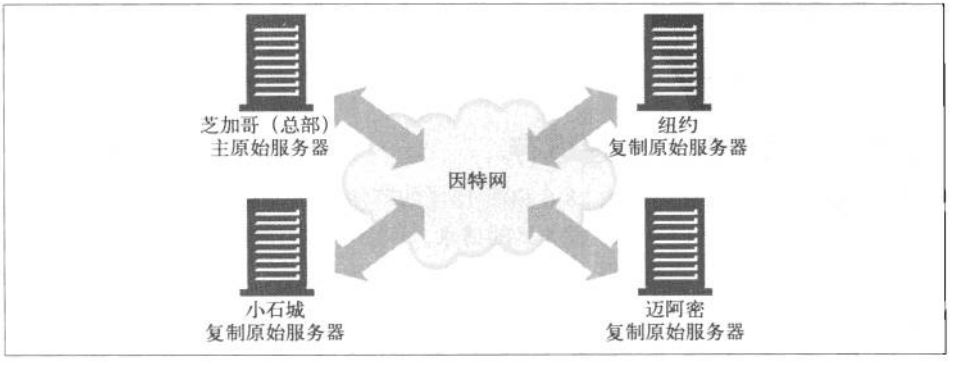


图 18-7　分散的镜像服务器

在图 18-7 的场景中，有以下两种方法把客户端的请求导向特定的服务器。

- HTTP 重定向

420 该内容的 URL 会解析到主服务器的 IP 地址，然后它会发送重定向到复制服务器。

- DNS 重定向

  该内容的 URL 会解析到 4 个 IP 地址，DNS 服务器可以选择发送给客户端的 IP 地址。

请参见第 20 章，以获取详细信息。

18.3.2 内容分发网络

简单地说，内容分发网络（CDN）就是对特定内容进行分发的专门网络。这个网络中的节点可以是 Web 服务器、反向代理或缓存。

18.3.3 CDN中的反向代理缓存

在图 18-6 和图 18-7 中，复制原始服务器可以用反向代理（也称为替代物）缓存来代替。反向代理缓存可以像镜像服务器一样接受服务器请求。它们代表原始服务器中的一个特定集合来接收服务器请求。（根据内容所在的 IP 地址的广告方式，这是有可能的，原始服务器和反向代理缓存之间通常有协作关系，到特定的原始服务器的请求就由反向代理缓存来接收。）

反向代理和镜像服务器之间的区别在于反向代理通常是需求驱动的。它们不会保存原始服务器的全部内容副本，它们只保存客户端请求的那部分内容。内容在其高速缓存中的分布情况取决于它们收到的请求，原始服务器不负责更新它们的内容。为了更容易地访问“热点”内容（就是高请求率的内容），有些反向代理具有“预取”特性，可以在用户请求之前就从服务器上载入内容。

421 CDN 中带有反向代理时，可能会由于存在代理的层次关系而增加其复杂性。

18.3.4 CDN中的代理缓存

代理缓存也可以部署在类似图 18-6 和图 18-7 的环境中。与反向代理不同，传统的代理缓存能收到发往任何 Web 服务器的请求。（在代理缓存与原始服务器之间不需要有任何工作关系或 IP 地址约定。）但是与反向代理比起来，代理缓存的内容一般都是按需驱动的，不能指望它是对原始服务器内容的精确复制。某些代理缓存也可以预先载入热点内容。

按需驱动的代理缓存可以部署在其他环境中——尤其是拦截环境，在这种情况下，2层或3层设备（交换机或路由器）会拦截Web流量并将其发送给代理缓存（参见图18-8）。

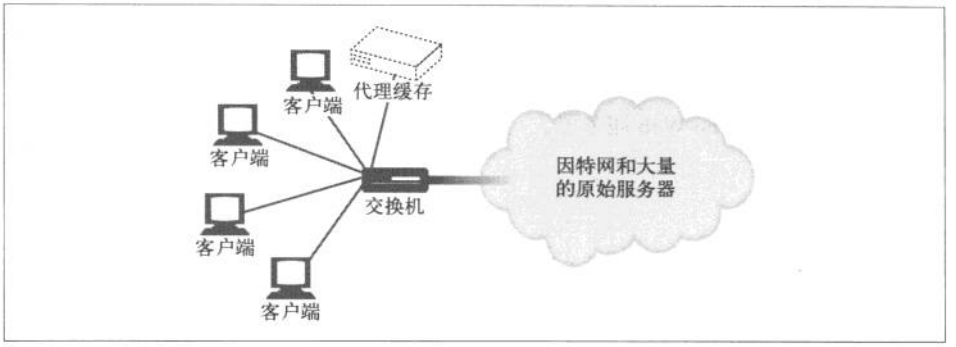


图18-8　客户端的请求被交换机拦截并发给代理缓存

拦截环境依赖于在客户端和服务器之间设置网络的能力，这样，所有合适的HTTP请求才能真正发送到缓存中去。（参见第20章）。根据收到的请求，将内容分布在缓存中。

18.4　让网站更快

前面一节提到的很多技术也能帮助网站更快地加载。服务器集群和分布式代理缓存或反向代理服务器分散了网络流量，可以避免拥塞。分发内容使之更靠近终端用户，这样从服务器到客户端的传输时间就更短了。请求和响应穿过因特网，在客户端和服务器间传输的方式是影响资源访问速度最主要的因素。重定向方法的详细内容参见第20章。

加速网站访问的另一种方法是对内容进行编码以便更快地传输。比如，对内容进行压缩，但前提是接收的客户端能够把内容解压缩。请参见第15章了解更多细节。 422

18.5　更多信息

参阅第3部分以了解如何使Web站点安全。下面的因特网草案和文档提供了Web虚拟主机服务和内容分发的更多细节。

- http://www.ietf.org/rfc/rfc3040.txt
  RFC 3040，“Internet Web Replication and Caching Taxonomy”（“因特网Web复制和缓存分类法”），这份文档是关于Web复制与缓存应用术语的参考文献。

- http://search.ietf.org/internet-drafts/draft-ietf-cdi-request-routing-reqs-00.txt
  "Request-Routing Requirements for Content Internetworking"("内容网际互连的请求路由需求")。

- *Apache: The Definitive Guide*[1]（《Apache 权威指南》）
  Ben Laurie 和 Peter Laurie 著，O'Reilly & Associates 公司出版。这本书讲述如何
423 运行开源的 Apache Web 服务器。

注 1：本书影印版由人民邮电出版社出版。(编者注)