第7章 应 用 层

在完成了所有预备知识的学习后，我们现在来到应用层，这里可以找到所有的网络应用。应用层下面的各层提供了传输服务，但它们并不真正为用户工作。在本章中，我们将学习一些实际的网络应用。

然而，即使在应用层也仍然需要协议的支持，以便各种应用程序能够工作。因此，在开始介绍这些应用之前，我们将先介绍其中的一个协议。这个协议就是DNS，它负责处理Internet命名问题。之后，我们将介绍3个实际应用：电子邮件、万维网（World Wide Web）和多媒体。我们将以对内容分发应用的简单介绍来结束本章，包括对等网络的基本概念。

7.1 DNS——域名系统

虽然在理论上，所有程序通过使用它们存储的计算机网络地址（例如IP），就可以访问Web主页、邮箱和其他资源，但是这些地址很难让人记住。而且浏览一个公司在128.111.24.41上的Web主页，意味着如果该公司将主页移到了另一台机器上，且该机器具有了不同的IP地址时，那么必须将该机器的IP地址通知到每个人。因此，人们引入了可读性好的高层名字，以便将机器名字与机器地址分离开。在这种方式下，无论真正实用的IP地址是什么，人们熟知的公司Web服务器为www.cs.washington.edu。然而，因为网络本身只能理解数字形式的地址，所以需要某种机制将名字转换成网络地址。下面，我们学习如何在Internet上完成这种从名字到地址的映射。

我们回到早期的ARPANET，那时只有一个简单的文件，名叫hosts.txt，它列出了所有的计算机名字和它们的IP地址。每天晚上，所有主机都从一个维护此文件的站点将该文件取回，然后在本地进行更新。对于一个拥有几百台大型分时机器的网络而言，这种方法工作得相当好。

然而，当几百万台PC连接到Internet以后，使用网络的每个人都意识到这种方法将不能继续有效工作了。一方面，这个文件变得非常庞大。然而，更重要的是，除非集中管理机器名字，否则主机名冲突的现象将会频繁发生；而且在一个巨大的国际性网络中，由于负载和延迟，要实现这种集中式的管理简直难以想象。为了解决这些问题，1983年人们发明了**域名系统**（DNS，Domain Name System）。自发明以来，它一直是Internet的关键组成部分。

DNS的本质是发明了一种层次的、基于域的命名方案，并且用一个分布式数据库系统加以实现。DNS的主要用途是将主机名映射成IP地址，但它也可以用于其他用途。RFC 1034、1035、2181给出了DNS的定义，后来其他文档对它又做了进一步的阐述。

简要地说，DNS的使用方法如下所述。为了将一个名字映射成IP地址，应用程序调用一个名为**解析器**（resolver）的库程序，并将名字作为参数传递给此程序。在图6-6中我

们看到的 gethostbyname 就是一个解析器例子。解析器向本地 DNS 服务器发送一个包含该名字的请求报文；本地 DNS 服务器查询该名字，并且返回一个包含该名字对应 IP 地址的响应报文给解析器，然后解析器再将 IP 地址返回给调用方。查询报文和响应报文都作为 UDP 数据包发送。有了 IP 地址以后，应用程序就可以与目标主机建立一个 TCP 连接，或者给它发送 UDP 数据包。

7.1.1 DNS 名字空间

管理一个大型并且经常变化的名字集合不是个简单的问题。在邮政系统中，名字管理要求所有的信件必须（隐式地或显式地）指定国家、州或省、城市、街道地址以及收件人的名字。通过这种地址的层次结构，纽约州 White Plains 市 Main 大街上的 Marvin Anderson 与德克萨斯州奥斯汀市 Main 大街上的 Marvin Anderson 就不会产生混淆。DNS 采用了同样的工作方式。

对于 Internet，命名层次结构的顶级由一个专门组织负责管理。该组织名为 Internet **名字与数字地址分配机构**（ICANN，Internet Corporation for Assigned Names and Numbers），创建于 1998 年，它是 Internet 成长为全球性并且受到经济关注的部分成熟标志。从概念上讲，Internet 被划分为超过 250 个**顶级域名**（top-level domains），其中每个域涵盖了许多主机。这些域又被进一步划分成子域，这些子域可被再次划分，依此类推。所有这些域可以表示为一棵树，如图 7-1 所示。树的叶子代表没有子域的域（当然要包含机器）。一个叶结点域可能包含一台主机，或者代表一个公司，该公司包含数以千计的主机。

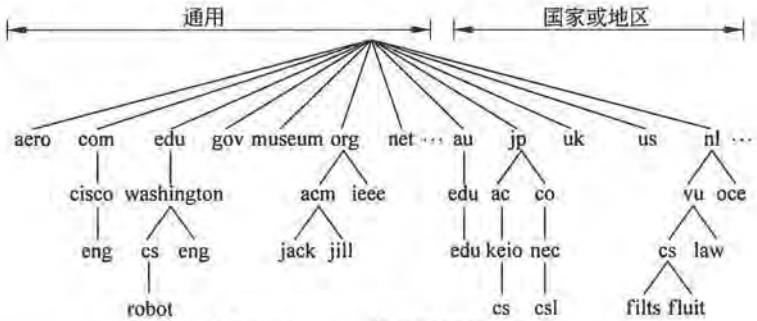

图 7-1 Internet 域名空间的一部分

顶级域名分为两种类型：通用的和国家或地区的。在图中 7-2 中列出的通用域名包括从 20 世纪 80 年代开始的原始域名和向 ICANN 申请引入的新域名。将来或许会增加其他的通用顶级域名。

国家或地区域名包括每个国家或地区，国家或地区的域名由 ISO 3166 文档定义。2010 年推出了使用非拉丁字母的国际化国家或地区域名。这些域名使得说阿拉伯语、西里尔文、中文或其他语言国家的人以自己的母语来命名他们的主机。

获得一个二级域名容易得多，比如 name-of-company.com。顶级域名由 ICANN 委任的**注册机构**（registrar）负责运行。要获得一个域名，只要到相应的注册机构（在这种情况下是 com），检查所需的名字是否可用，并且不是别人的商标。如果没有问题，请求注册域名的一方向管理域名的注册方支付一小笔年费，即可得到心仪的域名。

然而，随着 Internet 变得越来越商业化和越来越国际化，它也变得更具争议性，特别是在有关域名的事务上。这样的争议包括 ICANN 本身。例如，×××域名创造了好几年，法院已经审理要解决它。自愿将成人内容放在自己的域名空间中到底是好事还是坏事（有些人不希望成人内容对 Internet 上的所有人都开放，而其他人则想把成人内容放在一个域名中，以便让保姆过滤器很容易地找到并阻止儿童看到）？有些域名可以自我组织，而另一些则对谁可以得到一个该域名空间中的名字有限制，正如图 7-2 中所指出的那样。但是，什么样的限制才是适当的呢？我们用 pro 域名作为例子说明。这是一个分配给合格的专业人员所用的名字。但问题是谁才是专业的呢？显然医生和律师是专业人士，但自由摄影师、钢琴教师、魔术师、管道工、理发师、灭虫者、纹身艺术家和雇佣军等算专业人士吗？这些职业有资格当选吗？按照谁的标准呢？

域	预定使用	启用时间	限制否
com	商业	1985	不限
edu	教育科研	1985	限
gov	政府	1985	限
int	国际组织	1988	限
mil	军事	1985	限
net	网络提供商	1985	不限
org	非营利组织	1985	不限
aero	航空运输	2001	限
biz	公司	2001	不限
coop	合作	2001	限
info	信息	2002	不限
museum	博物馆	2002	限
name	人	2002	不限
pro	专业	2002	限
cat	加泰罗尼亚	2005	限
jobs	就业	2005	限
mobi	移动设备	2005	限
tel	联络资料	2005	限
travel	旅游业	2005	限
×××	色情业	2010	不限

图 7-2 通用的顶级域名

域名还与金钱挂钩。图瓦卢（国）将它的域名 tv 出售，获得租赁权 50 万美元，就是因为这个国家的域名非常适合广告电视网站。实际上，com 域名下几乎采用了每一个普通词（英文），而且这些词有最常见的拼写错误。家居用品、动物、植物以及身体各部位等名字都被人尝试注册了域名。注册所有人一转身就以高得多的价格把域名出售给感兴趣的人，而所有的一切都只是一个名字而已。这就是所谓的**域名抢注**（cybersquatting）。当 Internet 时代来临时，许多公司起跑得比较缓慢，以至于突然发现显而易见应该是自己的域名已经被他人注册了，于是它们试图收购这样的名字。在一般情况下，只要商标没有受到侵犯，而且没有涉及欺诈，那么域名的授予是先到先得的。然而，用以解决域名纠纷的政策仍在不断细化之中。

每个域的名字是由它向上到（未命名的）根节点的路径来命名的，路径上的各个部分用句点（读作“dot”）分开。因此，Cisco 公司的工程部门可能是 eng.cisco.com.，而不是像

UNIX 风格的名字/com/cisco/eng。请注意，这种层次的命名机制意味着 eng.cisco.com.中的 eng 不会与 eng.washington.edu.中的 eng 发生冲突，华盛顿大学的英语系可能会取它作为自己的域名。

域名可以是绝对的，也可以是相对的。绝对域名总是以句点作为结束（例如 eng.cisco.com.），而相对域名则不然。相对域名必须在一定的上下文环境中被解释才有的真正含义。无论是绝对域名还是相对域名，一个域名对应于域名树中一个特定的结点，以及它下面的所有结点。

域名不区分大小写，因此，edu、Edu 和 EDU 的含义都一样。各组成部分的名字最多可以有 63 个字符，整个路径的名字不得超过 255 个字符。

原则上，域名可以被插入到域名树中的通用域名空间或者国家域名空间。例如，cs.washington.edu 完全可以被列在国家域名 us 的下面，从而变成 cs.washington.wa.us。然而，实际上美国的大多数组织都位于某一个通用域名下面，而美国之外的大多数组织则位于其国家域名的下面。没有规则不允许一个组织在多个顶级域下注册域名。大型公司通常就这么做（例如 sony.com 和 sony.nl）。

每个域自己控制如何分配它下面的子域。例如，日本的 ac.jp 和 co.jp 域分别对应于 edu 和 com；但荷兰不做这样的区分，它把所有的组织直接放在 nl 下。因此，以下 3 个域名都表示大学的计算机科学系：

（1）cs.washington.edu（美国华盛顿大学）。

（2）cs.vu.nl（荷兰阿姆斯特丹自由大学）。

（3）cs.keio.ac.jp（日本庆应义塾大学）。

为了创建一个新域，创建者必须得到包含该新域的上级域的许可。例如，如果华盛顿大学建立了一个 VLSI（超大规模集成电路）组，并且希望将它命名为 vlsi.cs.washington.edu，那么这个组必须获得管理 cs.washington.edu 域名的管理员许可。类似地，如果创立了一所新的大学，比如说 University of Northern South Dakota，那么它必须请求负责 edu 域名的管理员将 unsd.edu 分配给它。按照这种方式分配域名就可以避免名字冲突，并且每个域都可以跟踪它所有的子域。一旦创建并注册了一个新域，则该新域就可以创建属于自己的子域，比如 cs.unsd.edu，而无须得到域名树中任何上层域的许可。

命名机制遵循的是以组织为边界，而不是以物理网络为边界。例如，即使计算机科学系和电子工程系在同一幢楼里，并使用同一个 LAN，但它们仍然可以属于完全不同的域。同样地，即使计算机科学系分散在 Babbage Hall 和 Turing Hall 这两幢楼里，但这两幢楼里的主机通常仍属于同一个域。

7.1.2 域名资源记录

无论是只有一台主机的域还是顶级域，每个域都有一组与它相关联的**资源记录**（resource record）。这些记录组成了 DNS 数据库。对于一台主机来说，最常见的资源记录就是它的 IP 地址，但除此以外还存在着许多其他种类的资源记录。当解析器把一个域名传递给 DNS 时，它能获得的 DNS 返回结果就是与该域名相关联的资源记录。因此，DNS 的基本功能是将域名映射到资源记录。

一条资源记录是一个五元组。尽管出于效率考虑资源记录被编码成了二进制的形式，但是大多数说明性资料还是用 ASCII 文本来表示资源记录，每一条记录占一行。在接下来的介绍中，我们将使用如下的格式：

```
Domain_name Time_to_live  Class    Type    Value
```

Domain_name（域名）指出了这条记录适用于哪个域。通常每个域有许多条记录，并且数据库的每份副本保存了多个域的信息。因此 Domain_name 是匹配查询条件的主要搜索关键字。数据库中资源记录的顺序则无关紧要。

Time_to_live（生存期）指明了该条记录的稳定程度。极为稳定的信息会被分配一个很大的值，比如 86 400（1 天时间里的秒数）；而非常不稳定的信息则会被分配一个较小的值，比如 60（1 分钟的秒数）。稍后当我们讨论缓存机制时将再回到这个议题。

每条资源记录的第三个字段是 Class（类别）。对于 Internet 信息，它总是 IN。对于非 Internet 信息，则可以使用其他的代码，但实际上很少见。

Type（类型）字段指出了这是什么类型的记录。DNS 记录有许多类型，图 7-3 列出了最重要的一些类型。

类型	含义	值
SOA	授权开始	本区域的参数
A	主机的IPv4地址	32位整数
AAAA	主机的IPv6地址	128位整数
MX	邮件交换	优先级，愿意接受邮件的域
NS	域名服务器	本域的服务器名字
CNAME	规范名	域名
PTR	指针	IP地址的别名
SPF	发送者的政策框架	邮件发送政策的文本编码
SRV	服务	提供服务的主机
TXT	文本	说明的ASCII文本

图 7-3 主要的 DNS 资源记录类型

SOA 记录给出了有关该名字服务器区域（后面将会介绍）的主要信息源名称、名字服务器管理员的电子邮件地址、一个唯一的序号以及各种标志位和超时值。

最重要的记录类型是 A（地址）记录，它包含了某台主机一个网络接口的 32 位 IP 地址。对应的 AAAA，或“quad A”记录包含了一个 128 位的 IPv6 地址。每台 Internet 主机必须至少有一个 IP 地址，以便其他机器能与它进行通信。某些主机有两个或多个网络接口，在这种情况下，它们就有两个或多个 A 或 AAAA 资源记录。因此，对于单个域名的查询可能获得多个地址。

最常用的记录类型是 MX 记录，它指定了一台准备接受该特定域名电子邮件的主机的名字。因为并非每台机器都做好了接收电子邮件的准备，因此必须用一个记录作特别说明。例如，如果有人打算发送电子邮件给某个人，比如 bill@microsoft.com，则发送主机必须找到在 microsoft.com 中愿意接收电子邮件的邮件服务器。MX 记录就是用来提供这样的信息。

另一个重要记录类型是 NS 记录。它指明了一台用于所在域和子域的名字服务器。这是一台拥有一份某域数据库副本的主机。这条记录被用于域名查询处理。稍后我们将对此

做简单的讨论。

CNAME 记录允许创建别名。例如，如果一个人很熟悉 Internet 的常规命名规则，他打算给 MIT 计算机科学系名叫 paul 的人发送邮件，他就猜测此人的邮箱名是 paul@cs.mit.edu。实际上，这个地址不正确，因为 MIT 计算机科学系的域名是 csail.mit.edu。但是，MIT 可以创建一条 CNAME 记录指向人和程序的正确方向，为那些不知情的人提供一项服务。类似下面这样的一条记录就可以完成此任务：

```
cs.mit.edu  86400  IN    CNAME    csail.mit.edu
```

如同 CNAME 一样，PTR 指向另一个名字。但是 CNAME 只是一个宏定义（即可以用另一个串来替代一个串的机制），而 PTR 与 CNAME 不同，它是一种正规的 DNS 数据类型，它的确切含义取决于所在的上下文。实际上，PTR 几乎总是被用来将一个名字与一个 IP 地址关联起来，以便能够通过查询 IP 地址来获得对应机器的名字，这种功能称为逆向查询（reverse lookups）。

SRV 是一个新的记录类型，它把主机标识为域内的一种给定服务。例如，cs.washington.edu 的 Web 服务器可以标识成 cockatoo.cs.washington.edu。这个记录能产生 MX 记录，并执行相同的任务，但只适用于邮件服务器。

SPF 也是一个新的记录类型。它可以让一个域把邮件服务信息进行编码，即域内哪台机器负责把邮件发送到 Internet 其余地方。这条记录有助于接收机器检查邮件是否有效。如果接收到的邮件来自一台通话本身躲躲闪闪的机器，但域名记录说明邮件只能来自域外一台称为 smtp 的机器，那么该邮件就是伪造的垃圾邮件。

列表的最后一行给出了 TXT 记录，这是最初为了允许域以任意方式标识自己而提供的一种途径。如今，它们通常被编码成机器可读信息，一般是 SPF 信息。

最后，资源记录还包括 Value 字段，该字段的值可以是一个数字、一个域名或者一个 ASCII 字符串，其语义取决于记录的类型。图 7-3 中给出了每种主要记录类型的 Value 字段的简短描述。

图 7-4 显示的例子给出了在一个域的 DNS 数据库中可能找到的信息种类。它描述了图 7-1 中 cs.vu.nl 域（假设的）数据库的一部分。该数据库包含 7 种资源记录。

图 7-4 中第一个非注释行给出了该域的一些基本信息，我们以后将不再关心这些信息。接下来的两个表项给出了第一个和第二个投递电子邮件给 person@cs.vu.nl 的地点。首先尝试的是 zephyr（一台特定的机器），如果给它发送失败了，则下一个应该尝试给 top 发送。下一行标识了该域的名字服务器为 star。

接下来是一个空白行，加入空白行的目的是为了增加可读性。再下面的几行给出了 star、zephyr 和 top 的 IP 地址。紧跟这些行后面的是别名 www.cs.vu.nl，因此可以使用这个地址而无须指派一台特殊的机器。创建这个别名的好处是允许 cs.vu.nl 改变它的 WWW 服务器，无须使得人们原来所使用的地址失效。类似地，ftp.cs.vu.nl 也是一个别名。

有关机器 flits 的信息部分列出了两个 IP 地址和处理发送至 flits.cs.vu.nl 电子邮件的三种选择。首先选择的是 flits 本身，但是如果它失效，zephyr 和 top 是第二个和第三个选择。

; cs.vu.nl的授权数据				
cs.vu.nl	86 400	IN	SOA	star boss (9527, 7200, 7200, 241 920, 86 400)
cs.vu.nl.	86 400	IN	MX	1 zephyr
cs.vu.nl.	86 400	IN	MX	2 top
cs.vu.nl.	86 400	IN	NS	star
star	86 400	IN	A	130.27.56.205
zephyr	86 400	IN	A	130.37.20.10
top	86 400	IN	A	130.37.20.11
www	86 400	IN	CNAME	star.cs.vu.nl
ftp	86 400	IN	CNAME	zephyr.cs.vu.nl
flits	86 400	IN	A	130.37.16.112
flits	86 400	IN	A	192.31.231.165
flits	86 400	IN	MX	1 tlits
flits	86 400	IN	MX	2 zepnyr
flits	86 400	IN	MX	3 top
rowboat		IN	A	130.37.56.201
		IN	MX	1 rowboat
		IN	MX	2 zephyr
little-sister		IN	A	130.37.62.23
laserjet		IN	A	192.31.231.216

图 7-4 针对 cs.vu.nl 域名可能的 DNS 数据库一部分

接下来的 3 行包含了一台计算机的典型表项，在这个例子中是 rowboat.cs.vu.nl。它提供的信息包括 IP 地址、主要邮件存放处和次要邮件存放处。 然后是一个针对一台计算机的表项，这台计算机自己不能接收邮件；紧随其后的一个表项可能是指一台连接到 Internet 的打印机。

7.1.3 域名服务器

至少在理论上，一台域名服务器就可以包含整个 DNS 数据库，并响应所有对该数据库的查询。但实际上，这台服务器会因负载过重而变得毫无用处。而且，一旦它停机，则整个 Internet 将会瘫痪。

为了避免由于单个信息源带来的各种问题，DNS 名字空间被划分为一些不重叠的区域（zones）。针对图 7-1 的名字空间，一种可能的划分方法如图 7-5 所示。每个圈起来的区域包含域名树的一部分。

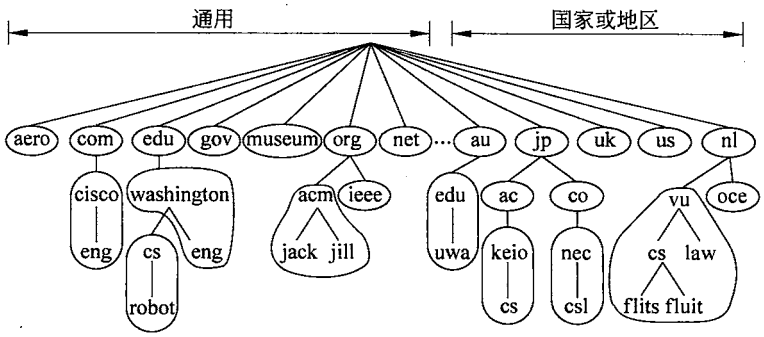

图 7-5 按区域划分（圈起来的）的部分 DNS 名字空间

区域边界应该放置在区域中的什么位置由该区域的管理员来决定。这个决定在很大程度上取决于需要在哪里使用多少个名字服务器。例如，在图 7-4 中，华盛顿大学有一个区

域 washington.edu，它要处理 eng.washington.edu 但不能处理 cs.washington.edu，这是另一个区域，有它自己的域名服务器。当有些系（比如英语系）不希望运行自己的域名服务器，而另外一些系（比如计算机科学系）却希望运行自己的域名服务器时，就有可能作出这样的决定。

每个区域都与一个或多个域名服务器关联。这些服务器是持有该区域数据库的主机。通常情况下，一个区域有一个主域名服务器和一个或多个辅域名服务器。主服务器从自己磁盘的一个文件读入有关域名信息，辅域名服务器从主域名服务器获取域名信息。为了提高可靠性，一些域名服务器可以设置在区域外面。

查询一个名字和找出其对应地址的过程称为域名解析（name resolution）。当解析器需要查询一个域名，它就把该查询传递给一个本地域名服务器。如果需要寻找的域恰好落在该域名服务器管辖下，比如 top.cs.vu.nl 在 cs.vu.nl 的管辖下，则该域名服务器就返回权威资源记录。一个**权威记录**（authoritative record）由管理该记录的权威部门提供，因此总是正确的。权威记录的权威性是相对**缓存记录**（cached record）而言的，缓存的记录有可能过时了。

如果被查询域在远端，比如 flits.cs.vu.nl 要找到华盛顿大学（UW）robot.cs.washington.edu 的 IP 地址，会发生什么？在这种情况下，如果本地没有关于相关域的缓存信息，那么域名服务器启动一次远程查询。该查询遵循如图 7-6 所示的过程。第 1 步，显示查询报文被发送到本地域名服务器。查询中包含了被查询的域名、类型（A）和类（IN）。

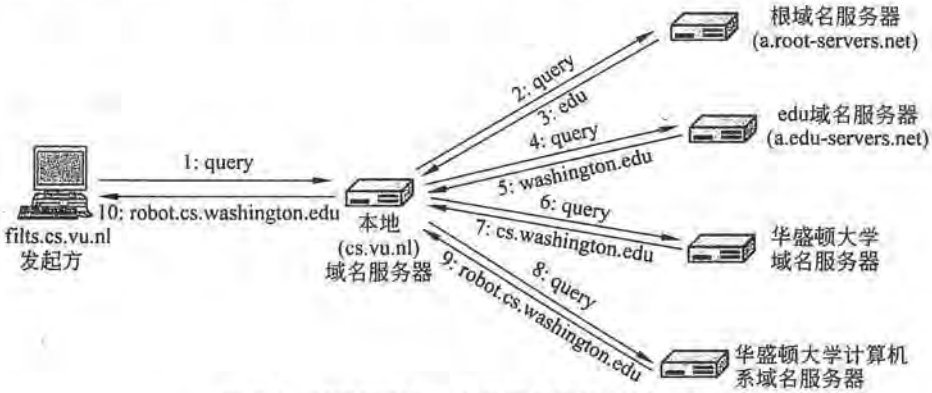

图 7-6 解析器查询一个远程名字的 10 个步骤

下一步是通过请求其中之一的根域名服务器来启动域名层次结构顶部的查询。这些域名服务器包含每个顶级域名的有关信息。这显示在图 7-6 中的第 2 步。为了与一个根服务器取得联系，每个域名服务器必须有一个或多个根域名服务器的信息。这个信息通常放在一个系统配置文件中，在 DNS 服务器启动时把该文件加载到 DNS 缓存。这个文件很简单，只是列出了关于根服务器的 NS 记录和相应的 A 记录。

总共有 13 个根 DNS 服务器，毫无创意地被命名为从 a-root-servers.net 到 m.root-servers.net。每个根服务器可以是逻辑上的一台计算机。然而，由于整个 Internet 依赖于根服务器，因此它们必须是能力超强的，并且被大量复制的计算机。大多数服务器被放置在多个地理位置，查询报文通过选播路由到达其中一台服务器。选播是一种特殊的路由，它能将数据包路由到最近的一个目标地址实例。我们在第 5 章中描述了选播路由。复制技术

能提高域名系统的可靠性和性能。

根域名服务器不可能知道在UW的一台机器的地址，可能还不知道UW的域名服务器。但是，它必须知道edu域的域名服务器，因为cs.washington.edu属于edu域管辖下。在第3步，它返回查询的答案，其中包括了名字和IP地址。

然后本地域名服务器继续尽职地执行任务。它将整个查询发给edu域名服务器（a.edu-servers.net）。该域名服务器返回UW的域名服务器。上述过程显示在图中的第4步和第5步。现在，快接近目标了，本地域名服务器把查询发送给UW的域名服务器（第6步）。如果被查询的域名是英语系，则马上能找到答案，因为UW域包括了英语系。但是，计算机科学系选择运行自己的域名服务器，因此该查询返回的是UW计算机科学系的域名服务器和IP地址（第7步）。

最后，本地域名服务器查询UW计算机科学系的域名服务器（第8步）。这台服务器是cs.washington.edu的权威机构，所以它必定有答案。它返回最终的答案（第9步），即作为响应flits.cs.vu.nl（第10步）转发的本地域名服务器。至此，查询的域名得到了解析。

你可以用标准的工具来探寻整个查询过程，比如安装在大多数UNIX系统中的dig程序。例如，键入：

```
dig@a.edu-servers.net robot.cs.washington.edu
```

向a.edu-servers.net域名服务器发出一个针对robot.cs.washington.edu的查询，并打印出查询结果。这个结果显示了上面例子中第4步获得的信息，你将了解到UW域名服务器的名字和IP地址。

有关这种长距离远程查询的场景，有3个技术要点值得讨论。首先，在图7-6中两个不同的查询机制在工作。当主机flits.cs.vu.nl将查询发送给本地域名服务器后，该域名服务器就代替该主机处理域名解析工作，直到它返回所需的答案。这里的答案必须是完整的，它不能返回部分答案。虽然部分答案可能也会有所帮助，但不是查询应该得到的结果。这个机制称为**递归查询**（recursive query）。

另一方面，根域名服务器（和每个后续的域名服务器）并不是递归继续查询本地域名服务器。它只是返回一个部分答案，并移动到下一个查询操作。本地域名服务器负责继续解析，具体做法是发出进一步的查询报文。这个机制称为**迭代查询**（iterative query）。

一次域名解析可以涉及这两种机制，如同该例子所表明的那样。递归查询似乎总是最可取的，但许多域名服务器（尤其是根服务器）并没有采用这种处理方式，它们太忙了。迭代查询则将重负压在了查询发起方。本地域名服务器支持递归查询的理由是它负责为其域内的主机提供服务。这些主机不必配置成运行一个完整的域名服务器，它们只要刚好能到达本地域名服务器即可。

值得讨论的第二点是缓存。所有的查询答案，包括所有的部分答案都会被缓存。这样，如果另一台cs.vu.nl主机需要查询robot.cs.washington.edu，那么答案早就有了。更妙的是，如果在同一个域中的一台不同主机需要查询另一台主机，比如galah.cs.washington.edu，那么此次查询可直接发送到权威的域名服务器。类似地，针对washington.edu其他域的查询也可以从washington.edu域名服务器直接开始。这种使用缓存答案的做法可大大降低一次查询的步骤，并能提高查询性能。事实上，我们在例子中勾勒出的原始场景是最坏的情况，

通常发生在没有任何缓存有用信息时。

然而，高速缓存的答案不具权威性，因为在 cs.washington.edu 所做的信息变化将不会传播到世界上所有可能知道它的缓存。出于这个原因，缓存的表项不应该生存得太长。这就是为什么在每条资源记录中要包含 Time_to_live 字段的原因。它告诉远程域名服务器可缓存本记录多长时间。如果某台机器已经多年使用相同的 IP 地址，将它的信息缓存 1 天可能是安全的。而对于波动比较大的信息，几秒钟或一分钟后清除掉记录的做法或许更安全。

第三个讨论的问题关于查询和响应使用的传输层协议。域名系统采用的是 UDP。DNS 消息通过 UDP 数据包发送，格式非常简单，只有查询和响应；域名服务器可用此数据包继续进行解析操作。我们不讨论这种报文格式的细节。如果在很短的时间内没有响应返回，DNS 客户端必须重复查询请求；如果重复一定次数后仍然失败，则尝试域内另一台域名服务器。查询过程这样设计的主要目的是为了应付出现服务器关闭以及查询或响应包丢失的情况。每个查询报文都包含 16 位的标识符，这个标识符将被复制到响应包中，以便域名服务器将接收到的答案与相应的查询匹配，即使它同时发出了多个查询也不会弄混查询结果。

即使 DNS 的目的很简单，人们应该明确这是一个庞大而复杂的分布式系统，它由数百万计的域名服务器组成，这些服务器要一起协同工作。DNS 在人类可读域名和机器 IP 地址之间形成了一个关键的衔接。为了提高性能和可靠性，它还利用了复制和缓存机制，同时设计得具有很强的鲁棒性。

我们没有涉及任何安全，但正如你可能想象的那样，如果心怀恶意，那么拥有改变域名到地址映射的能力就可能造成灾难性的后果。出于这个原因，研究者们已经开发了专用于 DNS 的安全扩展，这个安全机制称为 DNSSEC。我们将在第 8 章介绍这些内容。

应用程序对域名的使用方式希望更加灵活化，这样的应用需求逐渐显露。例如，首先对内容进行命名，然后解析出拥有该内容的附近一个主机的 IP 地址。这种映射模式特别符合搜索一个电影并下载它的应用模式。这时，人们关注的是电影本身，而不是某个拥有电影拷贝的计算机，因此解析所要的全部结果只是附近具有该影片拷贝的任何一台计算机 IP 地址。内容分发网络就是完成这个映射的一种实现方式。我们将在 7.5 节介绍如何利用本章后面的 DNS 来构建这样的网络。

7.2 电子邮件

电子邮件或者更常用的 E-mail，已经存在 30 多年了。由于比纸质信件更快更便宜，电子邮件成为自早期 Internet 出现以来最广泛的应用。在 1990 年以前，它主要被用于学术界。在整个 20 世纪 90 年代，它变得普及起来并呈指数形式增长，以至于现在每天发送的电子邮件数量远远超过了**传统的纸质邮件**（snail mail）数量。其他形式的网络通信，比如即时消息和 IP 语音在近 10 年也有了极大的发展，但是电子邮件仍然是 Internet 通信的主要负载。电子邮件广泛地用于业界公司内部的通信，例如，分散在世界各地的员工们就一个复杂项目进行协同。不幸的是，像纸质信件一样，大多数电子邮件都是邮寄宣传品或者**垃圾邮件**（spam），某些甚至 10 封邮件里面高达 9 封是垃圾邮件（McAfee，2010）。

与大多数其他通信方式一样，电子邮件有它自己的约定和风格。特别是它非常不拘小