第一章 绪 论

在这一章，我们首先对本教程要讨论的计算机网络安全问题做一概要性的阐述，然后在第 1.2 节汇集因特网 TCP/IP 协议最重要的技术性内容。熟悉 TCP/IP 协议的读者可以跳过第 1.2 节，在需要的时候再回到这里查阅细节。

1.1 网络安全概论

当前，没有人能否认计算机网络—最杰出的代表就是因特网—对人类社会带来的伟大进步。然而具有讽刺意味的是，同样也没有人敢于轻信计算机网络—因特网仍然是其最突出的例子—提供了一个值得信任的环境。计算机网络安全—当前其最主要的涵义就是因特网安全—所要解决的根本问题因此可以表述为：如何在一个不可信任的环境之下实现可信任的通讯？更进一步，如何在一个不可信任的环境之下实现可信任的计算？

以上目标或许永远可望而不可及，实际情形更可能是在老问题不断被解决的同时，新问题和新挑战又不断被提出，这或许正是科学与技术发展最引人入胜之处。就目前而论，我们可以列举出网络环境中一些典型也是有趣的安全问题如下：

A 向 B 声称自己是“A”，B 如何验证“A”确实是 A？

A、B 彼此已确信对方的身份，接下来如何进行保密通讯？这里“保密”的含义是指除 A、B 之外的任何第三方都不能获得 A、B 之间传输的真正的消息。

A 需要访问系统 S 拥有的对象 O，S 如何保证 A 的这一访问是合法的？即使对合法访问，如何保证这一访问不会导致额外的副作用？

A 拟向 B 购买一项 B 声称“只有他自己才知道答案”的秘密(信息)，A 如何相信 B 所言属实？B 又如何在不泄露这一秘密的情况下使 A 相信自己？更进一步，如果 B 持有多项这类秘密，A 却不希望 B 得知自己究竟对哪条秘密感兴趣，A 应该如何与 B 完成这笔交易？

A 通过网络发行其拥有产权的信息产品，A 如何保证该信息仅仅被合法的用户所接收？如果合法用户出现盗版行为(背叛)，A 如何毫不含糊地识别出背叛者？更有甚者，如果多个合法用户合谋背叛，A 如何能肯定地识别其中至少一个背叛者？

A 将自己的信息委托存储于一个确实值得信任的服务提供者 B—即使如此也不意味着

A 愿意 B 知晓自己的隐私，这里所谓“值得信任”仅仅意味着 A 相信 B 不会主动破译这些秘密—那么，A 如何在需要的时候以不泄露任何信息的方式处理这些(以密文形式)存储于 B 的秘密？

投标者 A、B 如何在不泄露自身竞价的情况下验证彼此之间竞价的相对大小，从而公平竞争？

…… ……

这些问题还可以列举很多，其中 A、B 可以是人类用户、进程、信用卡、读卡器、*Web* 服务期、数据库服务器、无线传感器网络中的传感器节点、无线/移动网络中的移动终端等等；以上问题还可以发生于多个参与者之间而非仅限于两个参与者的情形，凡此种种。固然，我们急于知道如何解决这些问题，但实际上，网络安全技术发展的历史表明我们的首要问题应该是“如何准确表达一个安全问题”，进一步，“什么是一个安全问题的正确的解决方案”，更进一步，“如何证明一个解决方案是正确的”，最后才是“如何构造一个正确而且实用的解决方案”。经验证明大量的解决方案并不正确，或无法证明其正确，或虽然正确但缺乏实用性。实际上，既经得起严格的理论验证同时又实用有效的安全解决方案在目前并不多，从这一意义上讲，网络安全技术还有大量的问题有待深入研究和发展。

但另一方面，网络安全技术，连同其他计算机安全技术(特别是计算机密码学)的发展，确实已经积累起许多有价值的成功的思想和方法。同时，网络安全技术也必须适应当前计算机技术的主流发展方向，例如复杂分布式系统的基于组件重用的构造与开发方法，面向服务的松偶合计算架构等，这些都使得网络安全越来越成为一个高度复杂和综合的技术领域。

在当代网络安全领域丰富多彩的各种问题中，这本教程只选择了几类最典型的网络安全问题进行阐述。虽然这些问题都具有引导初学者入门的性质，但作者并不希望因此而使读者距离当代网络安全领域的前沿过远，为此我们对每一问题的讨论都尽量达到一定深度。第一部分从典型的入侵机制入手，详细阐述了几类典型的溢出攻击机制，目的是为了引出对入侵检测系统和防火墙系统这几类重要的安全系统的讨论，使读者在理解这些反入侵机制之前能对攻击行为本身有一个较深入的认识。入侵检测系统的功能在于识别恶意入侵行为，其中基于主机的入侵检测系统(HIDS)运行于工作站或服务器，通过识别进程的运行行为检测是否发生入侵；基于网络的入侵检测系统(NIDS)运行于边界网关，通过分析 IP 分组流检测是否发生入侵，防火墙则在最基本的意义上可以解释为通过对 IP 分组流进行过滤而实现安全控制的设备。第四章重点阐述 HIDS，比较详细地阐述了 *Wagner*(2000)所实现的基于程序正常行为规范进行检测的 HIDS 和 *Garfinkel* 等(2003)所实现的基于虚拟机、可集成多种检测策略的

HIDS，这是目前很有代表性的工作。第五章重点阐述防火墙和 NIDS，对防火墙的讨论以 *Linux* 平台上的软件防火墙 iptable 和 Cisco 安全路由器为实例详细讨论了其典型应用，对 NIDS 的讨论则以当前最有典型代表意义的 *Snort* 和 *Bro* 系统为实例，特别是 *Bro*，较详细阐述了其工作机理和所开发出来的先进技术。

本书第二部分以访问控制和安全策略为核心内容。安全策略广泛应用于各类安全系统，目的在于明确规范系统的行为。正是依据安全策略，一个系统的行为才有可能被明确界定为“安全”或“不安全”。安全策略也是实施访问控制的依据，后者目的就在于限定特定的主体只能以特定的权限访问特定的对象。事实上，访问控制是几乎所有安全系统都具有的基本安全机制，为此第六章除阐述访问控制的通用模型之外，还特别讨论了 *Unix/Linux*、*Windows 2K*、Java2 以及数据库系统中常见的访问控制机制。第七章阐述适合于分布式系统实现的安全策略模型，特别是详细阐述了 *Minsky* 等学者于九十年代末所发展的 *LGI* 模型及其对几类典型安全策略的实现。*LGI* 模型看似比较形式化，但实际上是一个概念非常直观的算法模型，特别适合于实现许多用普通方法不容易实现的安全策略，这一点读者仔细学习之后就能有所体会，而且这一章所给出的例子都具有实际应用价值。

本书第三部分以网络安全协议为核心内容。第八章概要阐述作为计算机密码学理论基础的初等数论知识和一些典型的、目前在 IT 业界实际应用的基本密码方案，其中是对混合加密方案(*Fujisaki-Okamoto* 方案、*REACT* 方案等)的介绍是目前其它计算机密码学教科书很少讨论、但非常有用的内容，当今许多加密方案(如欧盟著名的 *NESSIE PSec*1/2/3 方案)实际上正是这类混合方案的衍生实例。广播加密方案也是许多教科书没有讨论、但特别有用的主题，例如它可以用于知识产权保护，本章对此进行了概要阐述。考虑到基于身份公钥加密技术(*IBE*)的巨大应用潜力，该章还概要介绍了著名的 *Boneh-Franklin* 方案(2001)，包括以习题的形式介绍了 *IBE* 的几个有趣的应用。第九章从反面讨论了许多协议设计失败的例子，一方面是为后面阐述正确的协议设计建立依据，同时也为了使读者深切体会到对安全的基本方案的堆积丝毫不意味着能产生出安全的整体方案，恰恰相反，无论基本方案多么安全，不正确的协议结构可以导致完全不安全的后果！在此基础上，第十章到第十三章阐述了丰富的网络安全协议的实例，分别涵盖身份鉴别、基于口令的密钥交换、带身份认证的 2-方密钥交换和多方密钥交换协议，所引实例都是目前应用广泛(*SIGMA/IKE*、*TLS/SSL* 协议等)或有优良潜力的协议(如 *TESLA* 协议)。这一部分对基于口令的密钥交换协议、群钥交换协议及适用于无线传感器网络的密钥分发协议的讨论是至今其它教科书所没有的。当然，限于本书的性质和程度，我们不是从严格证明的角度进行讨论，而是从程序实现的角度对协议进行详细描述，

通过直观的讨论向读者解释这些协议为什么能够抵抗第九章中的那些攻击。

第四部分是前面几部分内容的综合应用和提升，包括网络安全协议的分析与验证技术(第十四章)和更高级的网络安全应用(第十五章)。这些内容都属于网络安全领域的当代研究前沿，但也属于初学者经过一定努力后可以理解的范畴，特别是第十五章的内容代表着新一代网络环境中的全新类型的安全服务。

1.2 因特网及 TCP/IP 协议

TCP/IP 协议是因特网的核心，分别工作于 OSI 模型的传输层和网络层。这一节概要总结因特网的 TCP/IP 协议，一方面帮助读者回顾一些重要的基本概念和技术内容，同时也作为今后论述的基础。

1.2.1 IP 协议

IP 协议的目的是建立一个统一的逻辑通讯层，使得所有计算机设备能以统一的方式标识自身、任何两台计算机之间—无论是分别处于互不兼容的局域网上、还是中间需要跨越多个类型迥异的物理网络—都可以相互通讯，而与其实际所处网络的数据链路层机制无关。换句话说，通过 IP 协议统一实现网络层的逻辑功能，这些高级功能屏蔽了各种具体的数据链路层的功能特点，将其变换为统一的网络层服务，作为其结果的逻辑网络，就是因特网。

因此，在体系结构上，因特网是各种局域网和广域网通过网络互连设备—路由器(*router*)—互相连接起来的网际，即网络的网络(图 1-1)。

在因特网上，所有数据以统一的形式，即 IP 分组，从一台计算机传输到另一台计算机。每个 IP 分组明确标识出其源计算机和目标计算机。路由器(实际上是一种专用计算机)用来连接多个网络，对 IP 分组起着“中转”作用，即对从邻接网络来的每个 IP 分组，路由器能按照该 IP 分组所标识的目标计算机找到正确的路径，将该 IP 分组“中转”到该路径的“下一站”。即使连接源计算机和目标计算机的路径发生变化，甚至原来的路径因某段链路发生故障而失效，路由器仍然有能力自动“发现”一条新的路径，将 IP 分组送达目的地。因此，路由器是实现因特网的一个重要组成部分。

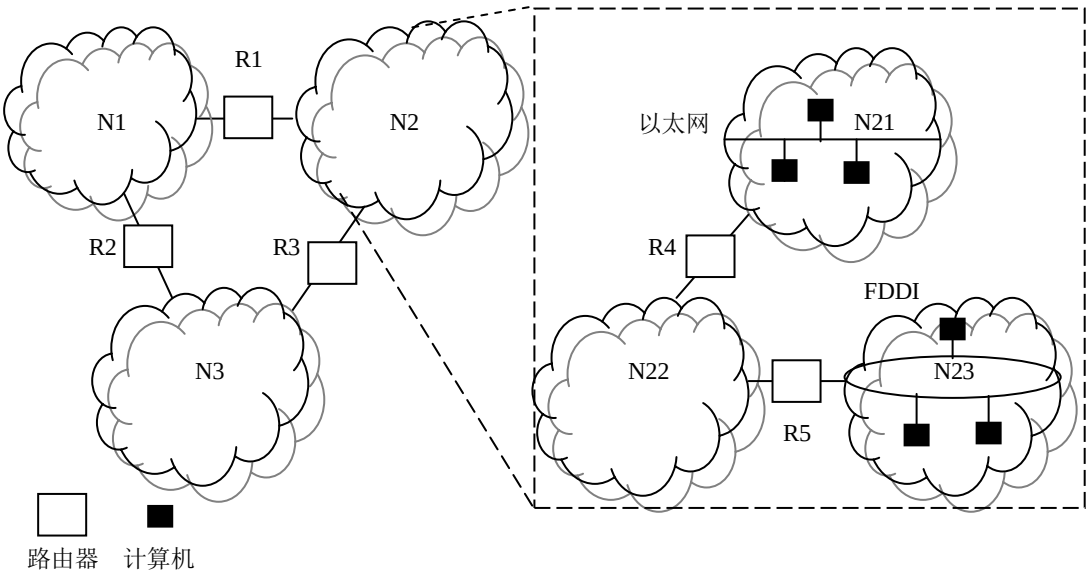


图 1-1 网络互联

IP 协议有以下几个最显著的特点，事实上也是 IP 协议的主要设计原则：

非可靠的分组传输(*unreliable delivery*)

分组交换(*packet switch*)

尽力而为式的服务(*best-effort delivery*)

IP 最基本、最本质的功能是将每个 IP 分组传输到目标计算机，无论源、宿计算机在网络上处于何种位置。但分组在传输过程中，会因为各种设备或网络故障而发生差错，典型的差错包括分组丢失（由于短暂的链路干扰、路由器或接收方计算机上的接收缓存空间溢出，等）、分组错序（一组分组的到达顺序与发送顺序不同）、分组延迟超界，等等。所谓非可靠的分组传输，是指 IP 本身并不试图纠正这些错误，甚至在一些情况下不试图发现这些错误（如延迟超界）。换句话说，IP 仅仅关心如何将 IP 分组送达目的地，但并不保证分组一定能没有任何差错地到达。

既然因特网最终所追求的是高可靠的通讯服务，这样一种网络层服务与因特网最终所追求实现的目标似乎有矛盾。问题的答案在于，从整体上，因特网不是仅仅依靠网络层—IP 协议—来达到最终目的，而是依靠传输层（见下文对 TCP 的概述）与网络层相互配合来达到的。IP 协议所不追求的可靠性传输通过传输层协议都得到了补偿，后者的最大特点之一就是可靠传输。图 1-2 是完整的因特网协议体系结构模型。

IP 协议的第二个特点，分组交换，指 IP 在实际发送、接收、特别是中转 IP 分组时，是将每个 IP 分组单独作为一个独立单元进行处理。 特别是，路由器单独为每个 IP 分组检索路由，而不考虑这些分组之间可能存在的任何关系（例如，不考虑这些分组是否来自于或要去到同一个目标计算机、不考虑这些分组是否属于同一对计算机之间的同一个会话连接，等），因此，甚至一组源、宿地址完全相同的 IP 分组也可能实际上经过完全不同的路由在因特网上传输。这样做的最大优点是利于提高网络的利用率，且分组的转发机制简单，但牺牲了一定的性能。

IP 协议的第三个特点，尽力而为式服务，指 IP 协议在实现分组的传输功能时，并不保证该分组一定可靠、无误地到达对方(即使到达，也不保证内容完整、顺序正确)。

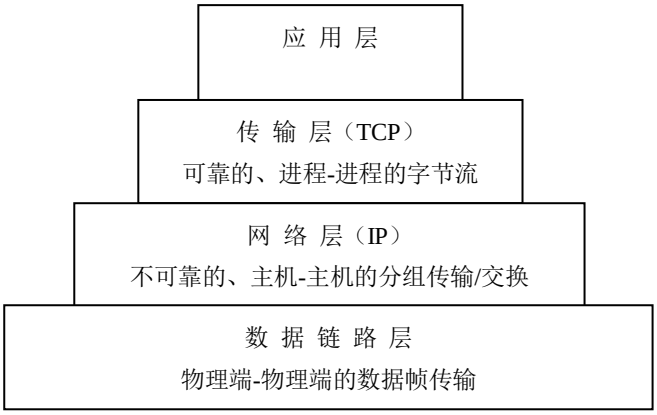


图 1-2 因特网协议栈

在实际发生数据传输时，IP 分组总是由特定的数据链路帧所承载，一段链路、一段链路地在因特网上传输。两者的组合关系如图 1-3 所示。帧首部中有特定的域指示当前所承载的是什么类型的数据，例如，若图 1-3 中是一个承载 IPv4 分组的以太网帧，则该帧首部中的类型域值为 0800（对 IPv6 则为 86DD，均为十六进制表示）。

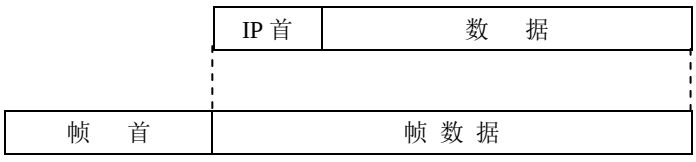


图 1-3 数据链路层的帧对 IP 分组的承载关系

IP 分组的格式如图 1-4 所示[1]。数据部分的格式取决于 IP 分组所承载的高层协议是什么，而与 IP 本身无关。IP 首部包含的大部分域用来表示对 IP 分组传输的控制信息，在传输过程中，路由器正是依据这些域的值决定分组应如何被转发。下面分别介绍各个域的涵义与用途。

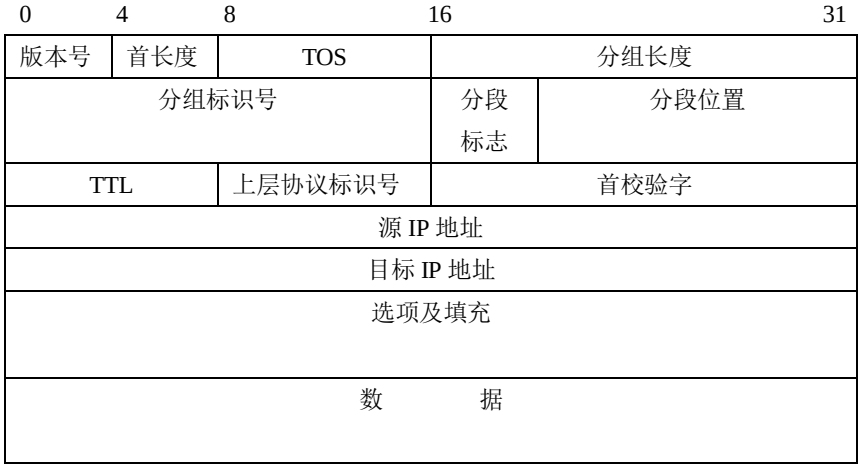


图 1-4 IP 分组的格式

版本号：4 位域，对 IPv4 就是 4，事实上这也是唯一合法的值。任何路由器或计算机在接收到 IP 分组时第一步就必须检查该域的值是否合法。

首长度：4 位域，表示 IP 首部的长度，注意该域值以 4 字节为单位，例如，若该域值=6，表示 IP 首部长度为 24 字节。注意一个 IP 首部除了选项部分之外，其它部分（从版本号直到目标 IP 地址）是必须存在的，这部分总长度为 20 字节。由于 4 位首长度域的最大值=15，最多能表示一个 15×4=60 字节长度的 IP 首部，因此 IP 选项部分最多只能有 60-20=40 字节。

TOS：8 位域，表示该 IP 分组所要求的服务等级和最优指标。具体地说，其低 3 位表示 IP 分组在路由器上被调度时的优先等级，其它四位表示所要求的最优指标，分别是 D-位、T-位、R-位和 C-位，若置位则分别表示要求传输延迟最短、负载最小、可靠性最高和费用最低。TOS 域的最高位没有定义。

分组长度：16 位域，域值表示完整 IP 分组（包括首部及数据）的总长度，并以字节为单位（注意与首长度域的单位不同），因此一个 IP 分组的最大长度是 $2^{16}-1=65535$。

分组标识号（16 位域）、分段标志（3 位域）和分段位置（13 位域）：用于 IP 分组的切割和重组，具体用法在下一小节描述。

[1] 这是 IPv4 分组的格式。今后除非特别声明，我们总是就 IPv4 进行论述。IPv6 与 IPv4 两者在安全能力方面并没有本质差别。

TTL：8 位域，表示一个 IP 分组的寿命。在 IP 分组被发送时，发送方为该域设置一个初始值，随后 IP 分组每经过一个路由器，该域的值被路由器减 1，而一个 TTL-域值为 0 的 IP 分组将被接收到它的任何设备所废弃，以此来避免因特网上的 IP 分组垃圾灾害。

上层协议号：8 位域，表示该 IP 分组的数据部分所属的协议，即目标计算机应如何解释 IP 数据。目标计算机上的 IP 协议软件正是依据该域值决定将 IP 分组中的数据提交给哪个协议软件继续处理。一些常见的情形如下：

表 1-1 常见的上层协议号及其涵义

上层协议号	数据部分应被哪个协议解释	说 明
0、255	保 留	
1	ICMPv4	差错报告协议，见 3.4.2
2	IGMP	组群管理协议
6	TCP	传输控制协议(传输层协议)
17	UDP	用户数据报协议(传输层协议)
51	AH	身份验证协议(IPSec)
52	ESP	数据加密协议(IPSec)
89	OSPF	开放最短路由发现协议

首校验字、IP 选项：首校验字对 IP 首部提供校验，具体计算方法及各种 IP 选项的涵义与表示方法读者都可以参考本章最后所列举的参考书，不在此赘述了。

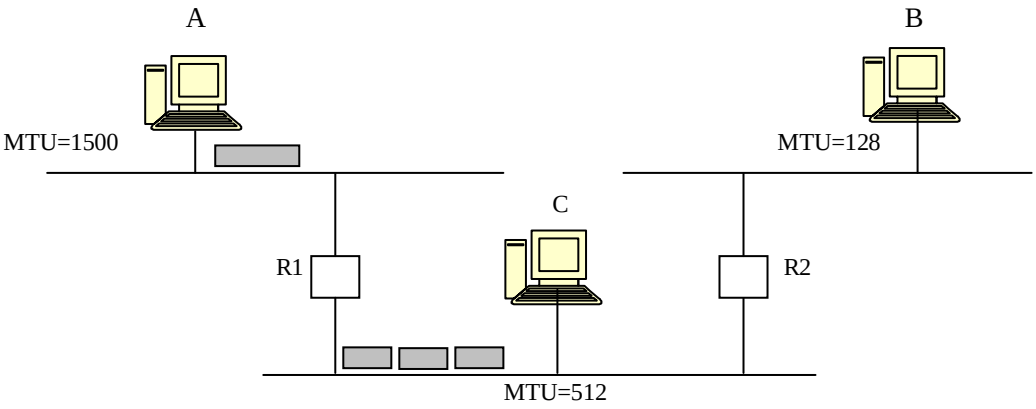
在实际发生数据传输时，IP 分组总是由特定的数据链路帧所承载，一段链路、一段链路地在因特网上传输，两者的组合关系如图 1-4 所示。然而，不同类型的链路，其数据帧所能承载的数据大小是不同的，例如以太网链路帧最大可达 1500 字节，FDDI 的帧最大可达 4470 字节，而某些链路帧最大仅 128 字节，凡此种种。每种类型链路的帧所能承载的最大数据长度，称为这种链路的最大传输单位（MTU：Maximum Transfer Unit），是每种类型链路的基本参数。

因此一个自然出现的问题就是：IP 分组在传输过程中如何跨越 MTU 值不同的链路？

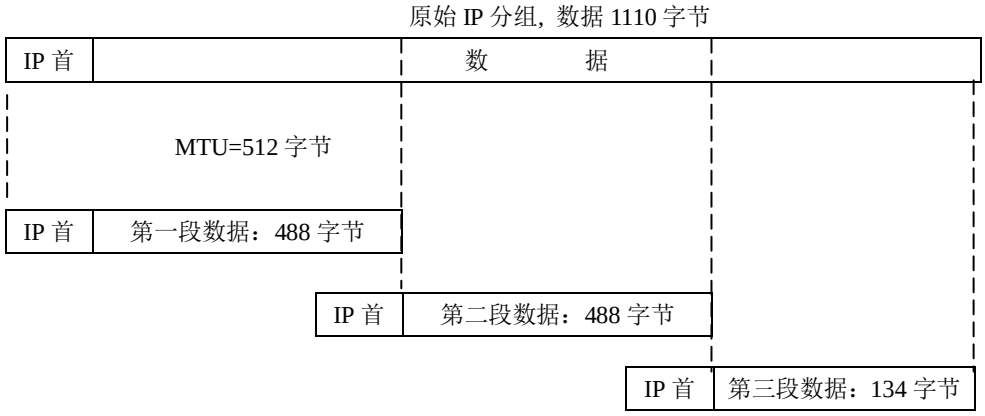
IP 解决这一问题的方式简洁明了：在 IP 分组从大 MTU 的链路向小 MTU 的链路传输前，对 IP 分组进行切割，使切割出的每个新 IP 分组的大小与小 MTU 匹配，同时在每个新 IP 分组的首部加入必要的信息，使得最终能重新组装出原来的 IP 分组。用来存储这些切割-重组信息的首部域，就是上小节未详细解释的分组标识号域（16 位）、分段标志域（3 位）和分段位置域（13 位，以 8 字节为单位），下面以一个例子来解释 IP 分组的切割-重组过程是如

何发生的，以及在这一过程中以上三个域是如何起作用的。

图 1-5(a)是一个包含三个网段的网络，每个网段的 MTU 标注在图中，计算机 A 发送出一个含 1110 字节数据的 IP 分组，目标计算机为 B。该 IP 分组总长度=IP 首部长度+IP 数据长度=20 字节+1110 字节=1130 字节。显而易见，这样一个 IP 分组在经过路由器转发时必须被切割，图 1-5(b)是实际被切割出来的三个新 IP 分组及其首部中相应域的值。



a)



b)

设原始 IP 分组的分组标识号为 X，分段标志域的 DF 位=0，则每个新 IP 分组的首部域值如下：

第一个 IP 分组：分组标识号=X，分段标志域的 MF 位=1，分段位置=0，分组长度=508

第二个 IP 分组：分组标识号=X，分段标志域的 MF 位=1，分段位置=61，分组长度=508

第三个 IP 分组：分组标识号=X，分段标志域的 MF 位=0，分段位置=122，分组长度=154

图 1-5 IP 分组的切割

a) 发生 IP 分组切割的网络 b) 在 R1 上发生的 IP 分组切割过程

请读者仔细验算图 1-5(b)中的数值。从这个例子我们可以总结出 IP 分组切割的规则：

计算机发送每个 IP 分组时，给 IP 分组一个唯一的 16 位分组标识号，如果该分组被切割，则所有切割出的 IP 分组都继承该标识号。接收方正是根据这一分组标识号识别出哪些 IP 分组是从同一个 IP 分组切割出来的。

计算机发送每个 IP 分组时，给 IP 分组首部域中分段标志域的 DF 位（该 3 位域的中间一位）赋值，仅当该值为 0 时，IP 分组在途中允许被切割，否则，当需要切割而 DF 位为 1 时，该 IP 分组被废弃并以 ICMP 协议向分组的始发方报告一个错误。

当切割发生时，切割出的最后一个 IP 分组的首部域中分段标志域的 MF 位（该 3 位域的最低一位）赋值为 0，其它分组的 MF 位为 1，由此，重构分组的计算机能准确判定哪一个分组是切割出的最后一个分组。

因为 IP 是分组交换机制，因此每个切割出的分组都被独立地传输，而且在后续的传输过程中还可能被继续切割。但读者通过仔细验算图 1-5 中的例子，不难理解无论切割实际发生多少次，原始 IP 分组最终都能被完整、准确地恢复出来。

IP 分组切割的最后一条规则是，IP 分组的重构过程永远只在目标计算机上发生。

关于 IP 协议的最后一个要点是：要使网络互联能真正工作，仅有 IP 协议是不够的，还需要一些辅助性协议，其中最重要的就是地址解析协议 ARP 和差错报告协议 ICMP，前者用于自动将 IP 地址翻译为 MAC 地址，后者用来探测和报告网络中的各种状态，特别是故障状态。事实上，这两个协议是 IP 协议族不可或缺的组成部分，IP 协议族实际上是 IP+ICMP+ARP+各种路由协议。

1.2.2 TCP 协议

因特网的传输层把网络层所提供的主机到主机之间的通信服务延伸为进程到进程之间的通信服务，并且能够满足可靠的、支持流量控制的、支持拥塞控制的等各种服务需求，从而为程序员在应用层开发各种各样的网络应用程序提供了可能。

TCP 是因特网传输层最重要的协议之一，具有所谓面向连接式的通讯模式，即在真正的数据通讯发生之前，双方进程首先建立逻辑连接以约定初始状态，以控制此后发生的数据通讯状态的迁移，直到数据通讯不再发生时，双方明确关闭该逻辑连接。TCP 将所传输的所有数据当作有顺序的字节流，对每个字节赋予一个唯一的 32 位编号，即所谓序列号(相邻字节的序列号连续编列)，同时辅以发送-应答机制，以此来保证数据可靠传输。

TCP 的信息传输单元通常称为段（Segment），它由段首部和负载组成，其中段首部的各

个域涵义如下(图 1-6)：

源/目的端口号：源或目的 TCP 端口号，各占用 2 个字节，取值范围是 0 — 65535。

序列号：本段 TCP 负载中第一个字节的编号，占用 4 个字节，取值范围是 0 — 2^{32}-1。

应答号（或称确认号）：接收方期待收到的下一个字节的序列号，占用 4 个字节，取值范围是 0 — 2^{32}-1。需要注意的是，TCP 接收方对已正确、完整收到的数据流进行确认的方法是指出期待收到的下一个字节的序列号，而不是指出已正确收到数据最后一个字节的序列号。

首部长度：段首部的长度，占用 4 个比特。注意长度单位是 4 个字节，这意味着若一个 TCP 段中该域的值是 5，则该 TCP 段首部的长度为 20 字节。因为本域的最大值是 15，故 TCP 段首部的最大长度为 60 字节。

保留域：在最初设计 TCP 时预留出来以便将来使用的 6 个比特。到目前为止，虽然已经出现了很多对这 6 个比特使用方法的建议，但还没有形成广为接受的标准。今天大多数 TCP 的实现都把这 6 个比特置为 0。

六个标志比特，它们的名称和含义分别如下：

(1)URG 位：置 1 则表示该段中含有紧急数据，并且“紧急指针”域有效；紧急数据是要求接收方必须立即处理的数据，例如在远程登录中用于中断一个程序运行的“Ctrl-C”。TCP 的紧急指针机制目前几乎很少使用。

(2)ACK 位：置 1 则表示“应答号”域有效，即该段将起到应答的作用。

(3)PSH 位：置 1 则表示要求接收方 TCP 实体将该段中的负载立即提交给应用层；否则接收方 TCP 实体可能会将该段负载中的数据暂时放在缓冲区中。

(4)RST 位：置 1 则表示发送方向对方报告一个异常，并要求重置连接。

(5)SYN 位：仅在 TCP 连接建立的过程中被使用(置 1)，表示发送方对其第一个数据字节所赋予的序列号等于本段的“序列号”域值+1。

(6)FIN 位：仅在 TCP 关闭连接的过程中被使用(置 1)，表示发送方要求单方向关闭连接，即发送方的所有数据已经传输完毕。

窗口大小：表示发送方当前所剩余的接收缓冲区的大小，取值范围是 0 — 65535，单位是字节。本域被用来做流量控制，它使对方随后所传输的每段负载的数据量不会超过本域所指定的值，从而防止因为段负载过大造成己方缓冲区的溢出。

TCP 的校验和域与 IP 类似，能力很弱，而紧急指针域目前很少实际使用，这里不再赘述。

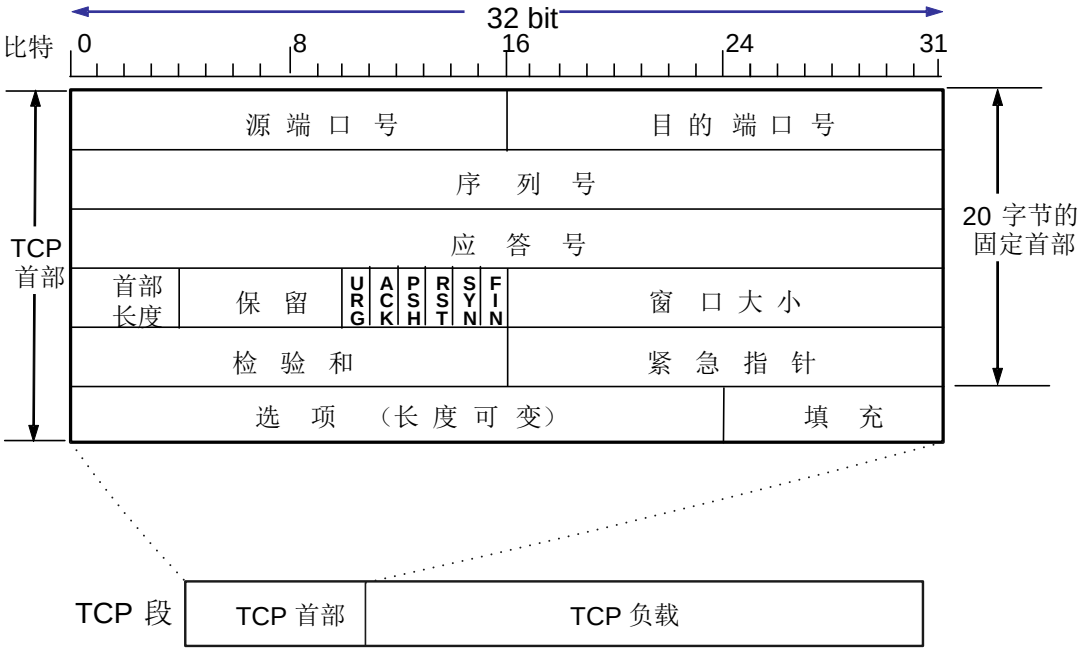


图 1-6 TCP 段格式

选项：用于携带 TCP 协议的一些附加信息，长度可变(0 — 40 字节)。如果选项域的长度不是四个字节的倍数，则用填充域将它凑齐，这是因为在计算机中，一般都是以四个字节为单位对数据进行存取的。到目前为止，已经提出了很多 TCP 选项的建议。这里只介绍如下最常见三种：

(1)最大段长度（Maximum Segment Size, 缩写为 MSS）：用于协商双方所能接受的最大段长度，一般是 1460 字节，这是因为以太网所允许的最大负载长度为 1500 字节，而一般 IP 首部会占去 20 字节，TCP 首部会再占去 20 字节。请读者注意，最大段长度仅指 TCP 负载的最大长度，而非整个 TCP 段的最大长度。

(2)窗口规模（Window Scale）：“窗口大小”域的单位，即窗口大小的值乘以窗口规模的值才是真正的发送方所能进一步接收的数据的字节数。这一选项是针对当前 TCP 的缓冲区不断增大的现象提出来的。关于它的详细描述，请参见 RFC 1323。

(3)选择应答（Selective Acknowledgments, 缩写为 SACK）：指出该 TCP 对滑动窗口协议的实现是否支持选择应答，以及携带用于选择应答的信息。在缺省情况下，TCP 对滑动窗口协议的实现都使用回退 N，但为了提高吞吐量和信道利用率，目前对 TCP 的实现一般都支持选择应答。选择应答是选择重传的一个变种，它允许 TCP 在一个应答段中对多个段进行应答。关于它的更多细节可参考本章后面列举的著作。

跟在 TCP 首部后面的是 TCP 的负载，它用来携带应用层的数据。它的最大长度一般为 MSS 所限定，上文已经提到过 MSS 的常见取值是 1460 字节。

TCP 是一个面向连接的协议，因此使用 TCP 来进行通信需要有连接建立和连接终止的过程。下面我们首先分别对它们进行讲述。在此之后，我们将给出 TCP 在管理一个连接的过程中的状态迁移模型，以及在这些状态之间进行转换的条件。

建立连接

TCP 的连接建立是一个三次握手过程，即需要经过三步 TCP 段的交换（如图 1-7 所示）。

第一步，客户机发送一个标志位 SYN 置 1 的 TCP 段给服务器（简称为 SYN 段，在该段的 6 个标志位中，只有 SYN 被置 1）。在这个 SYN 段中，序列号域含有客户机将要使用的初始序列号 x，负载域则不含任何数据。该段的意思是对服务器说“我想和你建立一个连接，在这个连接上，我将使用初始序列号 x。”SYN 段中的初始序列号 x 随机选取，并非简单的使用 0。这样做的原因有两个：一是使服务器能够判断重传的 SYN 段，二是出于安全的考虑，防止攻击者进行所谓“连接劫持”。

第二步，服务器在收到 SYN 段后，会发送一个标志位 SYN 和 ACK 都置 1 的 TCP 段给客户机（简称 SYN-ACK 段）。在这个段中，序列号域含有的是服务器将使用的初始序列号 y，应答号域含有的是对客户机初始序列号的应答 x+1。它的意思是对客户机说：“我同意建立连接，已经知道你将使用初始序列号 x，我将使用初始序列号 y。”在返回 SYN-ACK 段之后，服务器会在内存中建立起关于这个连接的数据结构，用来维护这个连接上的各种参数和状态[2]。

第三步，客户机在收到 SYN-ACK 段后，会发送一个标志位 SYN 置 0 和 ACK 置 1 的 TCP 段给客户机(简称 ACK 段)。在这个段中，序列号域含有的是客户机将使用的下一个序列号 x+1，应答号域含有的是对服务器初始序列号的应答 y+1。它的意思是对服务器说：“我已经知道你同意建立连接和你将使用初始序列号 y。”在返回 ACK 段之后，客户机也会在内存中建立起关于这个连接的数据结构，用来维护这个连接上的各种参数和状态。

经过以上三步之后，TCP 连接被成功地建立起来。在随后的数据通信过程中所有 TCP 段的 SYN 都将被置 0，即 SYN 段和 SYN-ACK 段只可能出现在连接建立的过程里；每个 TCP 段的 ACK 都将被置 1，即都将包含对当前已经正确收到数据的应答。另外，我们把上面过程中首先发出 SYN 段那一方的行为称为主动打开（Active Open），被动等待 SYN 段那

[2]由于服务器的这种行为, SYN 风暴攻击(SYN Flooding Attack, 即向服务器发送大量的 SYN 段)是对服务器进行攻击的常用手段, 使服务器很快耗尽内存。

一方的行为称为被动打开（Passive Open）。在客户机/服务器通信模型中，客户机永远是主动打开的一方，服务器永远是被动打开的一方。

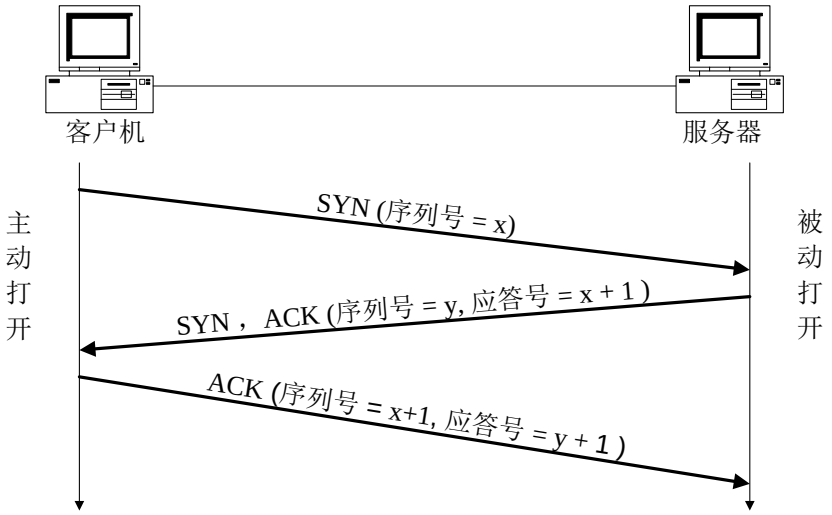


图 1-7 TCP 连接建立

关闭连接

TCP 连接上的数据传输可以在两个方向上同时进行。相应的，TCP 的连接关闭要求在两个方向上分别终止数据传输。总体来讲，TCP 的连接关闭一般要经过四步的 TCP 段交换，可以由客户机或服务器中的任意一方首先发起。它的具体步骤在图 1-8 中给出（该图假设主机 A 为连接关闭的首先发起方）。

第一步，主机 A 向主机 B 发送一个标志位 FIN 置 1 的 TCP 段（后文将简称 FIN 段），这里假设该段序列号域的值为 x。该段的意思是对 B 说:“我没有数据要发送了,将关闭 A→B 方向数据传输。”

第二步，主机 B 向主机 A 发送一个标志位 ACK 置 1 的 TCP 段，该段应答号域的值为 x+1。该段的意思是对 A 说：“我知道了。”

第三步，当主机 B 也已经传输数据完毕之后，主机 B 向主机 A 发送一个 FIN 段，这里假设该段序列号域的值为 y。该段的意思是对 A 说：“我也没有数据要发送了，将关闭 B→A 方向数据传输。”

第四步，主机 A 向主机 B 发送一个标志位 ACK 置 1 的 TCP 段，该段应答号域的值为 y+1。该段的意思是对 B 说：“我知道了。”

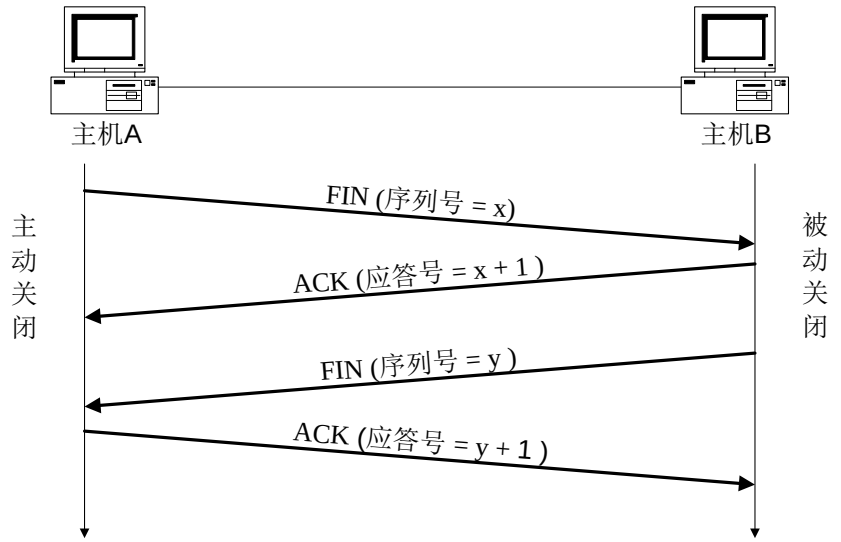


图 1-8 TCP 连接关闭

在经过以上四步之后，两个方向上的数据传输都被终止，TCP 连接也就被关闭了，此时主机 A 和主机 B 都将释放它们用于存放该连接数据结构的内存。在上面过程中，我们把首先发出 FIN 段那一方的行为称为主动关闭(Active Close)，随后发出 FIN 段那一方的行为称为被动关闭(Passive Close)。在客户机/服务器通信模型中，客户机和服务器都可以成为主动关闭的一方。

连接状态及状态迁移

TCP 对连接建立和连接终止这两个过程的管理通过一个有限状态自动机来实现。在这个 FSM 中，一共有 11 个状态，它们的名称和含义如表 1-2 所示。

图 1-9 描述了 TCP 在这 11 个状态之间进行转换的条件和所发生的动作。该图比较复杂，包含了 TCP 在所有正常和异常情况下对连接状态的管理。对于一般的应用来说，只需要掌握在正常情况下 TCP 连接状态转换的情况就可以了，即掌握图中如下两条线：

(1) 粗实线：该线描述了在正常情况下主动打开连接和主动关闭连接的过程，其具体状态转换步骤如下：

表 7-1 TCP 连接状态及其含义

状态	含 义
CLOSED	初始状态，没有连接存在
LISTEN	等待 SYN 段到达
SYN_RCVD	收到 SYN 段之后，并且已发送 SYN-ACK 段
SYN_SENT	发送 SYN 段之后，等待 SYN-ACK 段
ESTABLISHED	连接已经建立，进行数据传输的状态
FIN_WAIT_1	主动发出 FIN 段之后，等待 ACK 段
FIN_WAIT_2	收到对应于主动 FIN 段的 ACK 段，等待被动 FIN 段
TIME_WAIT	收到被动 FIN 段，并回答 ACK 段之后
CLOSING	双方同时发起连接关闭
CLOSE_WAIT	收到主动 FIN 段，并回答 ACK 段之后
LAST_ACK	被动发送 FIN 段之后

① 连接状态从 CLOSED 状态出发，当发送 SYN 段之后，进入 SYN_SENT 状态；

② 若收到 SYN-ACK 段，则返回 ACK 段并进入 ESTABLISHED 状态，此时连接已经建立，可进行数据传输；

③ 若数据传输完毕，则主动发送 FIN 段并进入 FIN_WAIT_1 状态；

④ 如收到对所发送 FIN 段的应答，则进入 FIN_WAIT_2 状态；

⑤ 若进一步收到对方的 FIN 段，则返回 ACK 段并进入 TIME_WAIT 状态；

⑥ 在 TIME_WAIT 状态等待两倍最大段寿命[3]后，重新回到 CLOSED 状态。

TCP 在主动关闭一方引入 TIME_WAIT 状态的原因是为了防止被动关闭方没有收到应答其 FIN 段的 ACK 段，从而再次发送 FIN 段。主动关闭方将在其 TIME_WAIT 状态当中，对可能收到的重传的 FIN 段进行处理。若读者使用“netstat –an”命令显示当前主机 TCP 连接状态，可能会发现有的 TCP 连接处于 TIME_WAIT 状态，而且会停留大约 1 到 2 分钟的时间，就是因为以上原因。

(2)粗虚线：该线描述了在正常情况下被动打开连接和被动关闭连接的过程，其具体状态转

[3] 最大段寿命(Maximum Segment Lifetime, 缩写为MSL)是TCP的参数之一, 它指一个TCP段可能在网络上停留的最长时间, 其取值视 TCP 的不同实现而有所不同, 一般的取值是 30 秒, 1 分钟或 2 分钟。

换步骤如下：

① 连接状态从 CLOSED 状态出发，当需要在某个端口上等待连接请求的时候，进入 LISTEN 状态；

② 若收到 SYN 段，则返回 SYN-ACK 段并进入 SYN_RCVD 状态；

③ 若收到 ACK 段，则进入 ESTABLISHED 状态，此时连接已经建立，可进行数据传输；

④ 若收到对方主动的 FIN 段，则返回 ACK 段并进入 CLOSE_WAIT 状态；

⑤ 若己方数据传输完毕，则发送被动的 FIN 段并进入 LAST_ACK 状态；

⑥ 若收到应答 FIN 段的 ACK 段，则重新回到 CLOSED 状态。

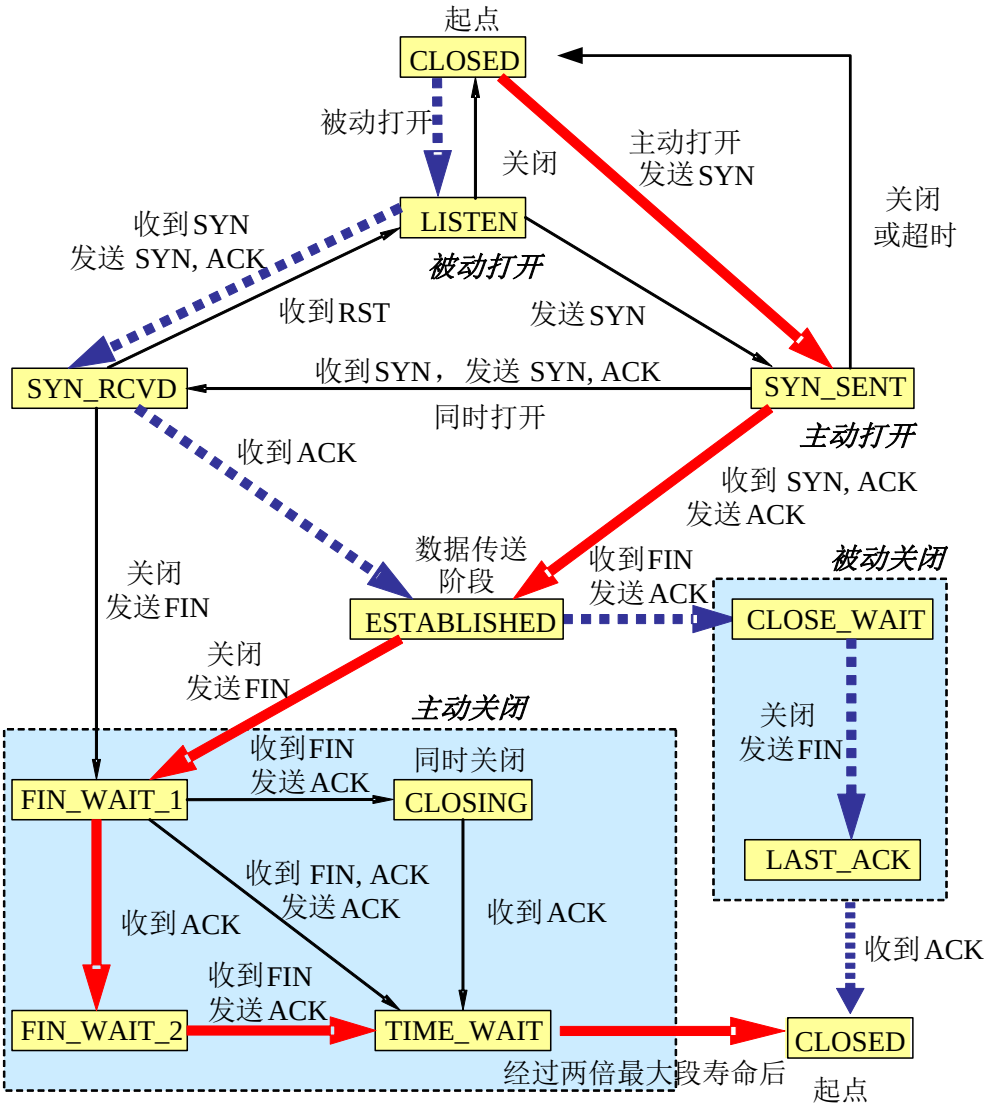


图 1-9 TCP 连接状态转换图

(粗实线或粗虚线描述正常情况，细实线描述异常情况)

最后概要阐述 TCP 的数据可靠传输机制。可靠传送是 TCP 提供的最主要功能，其实现方法是著名的滑动窗口机制。因为在 TCP 早期的时候主机的处理速度不足而且内存有限，所以那时的 TCP 使用回退 N-滑动窗口协议。但在今天，由于主机的处理速度已经很快而且内存很丰富，再加上人们非常在意一个协议的吞吐量和信道利用率，所以目前 TCP 的实现一般都支持选择应答，这是一种选择重传-滑动窗口协议的具体实现方法。

选择应答的基本思想是允许接收方在一个应答段中对已经收到的多个数据段进行应答，从而实现选择重传。选择应答是依赖于在 TCP 段首部的选项域中加入额外的信息来实现的(若仅使用 TCP 段固定首部中的应答号域，则只能应答一个段)。目前，MS Windows 和 Linux 内核对 TCP 的实现都支持选择应答。选择应答的具体实现比较复杂，这里就不给出详细的描述了，感兴趣的读者可以去阅读标准文档 RFC 2018。

TCP 除使用选择应答来提高吞吐量和信道利用率之外，还采用了下面两种技术来减少应答段的数量。一是捎带应答(Piggybacking)，即将应答段和数据段一起发送。这一点可以从 TCP 的段格式中清楚地看到，即 TCP 的段首部中同时含有用于数据的序列号域和用于应答的应答号域。二是累积应答(Cumulative Acknowledgments)，即并非对每个数据段都单独作出应答，而是在连续收到两个数据段之后，一起作出应答。为了实现累积应答，RFC 1122 对 TCP 产生应答段的方法作出了如下规定，见表 1-3。

表 1-3 TCP 应答段产生方法

事 件	产生应答段方法
一个段按顺序到达，并且前面的段已经被应答	推迟应答，最多推迟 500 毫秒；如果在 500 毫秒之内没有下一个段到达，则发送应答。
一个段按顺序到达，并且前面的段还没有被应答	立刻发送一个累积的应答
一个段没有按顺序到达，即含有比期待序列号更高的序列号，从而造成序列号空隙	立即重传上次发出的应答段，再次指出所期待的序列号
一个从低端填补序列号空隙的段到达	立刻发送应答

1.3 客户/服务器系统和对等系统

因特网上大量现有的分布式系统为客户/服务器模式，图 1-10 为一个典型的网络客户/服务器进程相互作用形式，服务器为 *Unix* 上的 *Telnet daemon*(守互进程)。大多数服务器在其运行平台上都具有特权身份，这使得服务器成为绝大多数网络攻击的对象，尽管这种攻击并不总是容易。另一方面，当代对等网络(即 P2P 网络，更准确的称呼应该是 P2P 式应用)也已经大大发展，这类系统中的各个成员在能力和权限方面彼此接近，特别是容易互相访问以共享资源(这是 P2P 系统的本质特点之一)，因此在这类系统中的恶意行为可能比客户/服务器更难以抵抗。

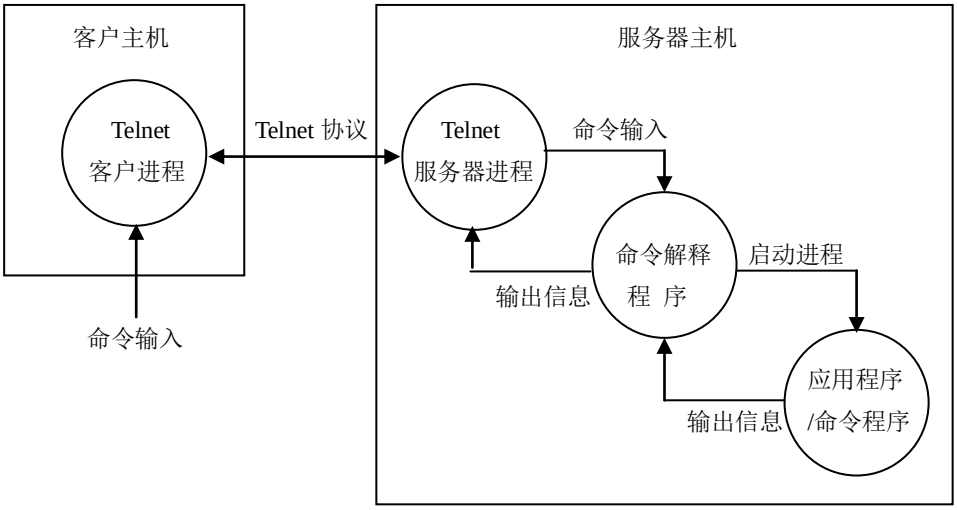


图 1-10 一个典型的客户/服务器系统

1.4 小结与进一步学习的指南

计算机安全(不限于网络安全)方面面向初学者的内容丰富的著作可以列举以下几部：

(德)D.Gollman 著 计算机安全学, 张小松 译, 北京: 机械工业出版社, 2008

(美)B.Schnerier 著 应用密码学, 吴世忠 等译, 北京: 机械工业出版社, 2000

M.Bishop *Computer Security: The Art and Science*, Addison-Wesley Inc. 2002(影印版由清华大学出版社出版)

第一部著作篇幅紧凑，但许多论题讨论得过于简略；后两部著作篇幅较大，更适合作为

学习参考书。注意 Gollman 和 Bishop 的著作都阐述了本书没有讨论的安全系统评估技术，特别是 Bishop 的著作详细讨论了美国联邦政府和欧洲共同体制定的信息系统安全评估准则与规范。

如果局限于 *Unix/Linux* 平台，则下面这部是公认的百科全书式的著作，从实用的角度对 *Unix/Linux* 环境的网络安全问题进行了详尽的阐述：

S.Garginkel, G.Spafford *Practical Unix and Internet Security*(3rd Ed), O'Reilly Media, Inc., 2003

计算机网络集中当代信息技术的精华，因特网则是体现这一事实的伟大成就。计算机网络在今天仍然在快速发展，其具体的技术内涵非常丰富。深入理解当代计算机网络技术对正确理解和熟练掌握网络安全技术是必备的基础，读者可以参考以下这些公认的优秀著作：

(美)Comer D.E.著 用 TCP/IP 实现网络互联 共 3 卷, 谢希人 等译, 北京: 电子工业出版社, 1999

(美)Stevens R 著. TCP/IP 详解 共 3 卷, 谢希仁 等译，北京：机械工业出版社，2000

Kurose J.F. and Ross K.W. *Computer Networking：A Top-down Approach Featuring the internet*, Addison-Wesley, 2003, 3rd Edition(影印版由高等教育出版社出版)

无线移动因特网是当代特别迷人的领域，也是因特网的未来，下面是关于无线移动网络的较全面教科书：

Agrawal D.P. and Zeng Q.A. *Introduction to Wireless and Mobile Systems*, Thompson Learning, 2003(影印版由高等教育出版社出版)

Rappapport T S. *Wireless Communication Principles and Practice*, Prentice-Hall Inc., 1996(影印版由电子工业出版社出版)

网络编程方面的优秀著作是 Stevens R. *Unix Networking Programming*, 2th ed, Vol. 1, socket-APIs and XTI, Prentice Hall, 1997(影印版由清华大学出版社出版)。