明文攻击和选择明文攻击吗？

R5. 考虑一个 8 块密码。这个密码有多少种可能的输入块？有多少种可能的映射？如果我们将每种映射视为一个密钥，则该密码具有多少种可能的密钥？

R6. 假定 N 个人中的每个人都和其他 $N-1$ 个人使用对称密钥密码通信。任两人（i 和 j）之间的所有通信对该 N 个人的组中的所有其他人都是可见的，且该组中的其他人都不应当能够解密他们的通信。则这个系统总共需要多少个密钥？现在假定使用公开密钥密码。此时需要多少个密钥？

R7. 假定 $n=10\,000$、$a=10\,023$ 和 $b=10\,004$。请你使用等同的模算术来心算 $(a \cdot b) \bmod n$。

R8. 假设你要通过加密对应于报文 1010111 的十进制数来加密该报文。该十进制数是什么？

8.3 ~ 8.4 节

R9. 散列以何种方式提供比检验和（如因特网检验和）更好的报文完整性检验？

R10. 你能够“解密”某报文的散列来得到初始报文吗？解释你的答案。

R11. 考虑 MAC 算法（图 8-9）的一种变形算法，其中发送方发送 $(m, H(m)+s)$，这里 $H(m)+s$ 是 $H(m)$ 和 s 的级联。该变形算法有缺陷吗？为什么？

R12. 一个签名的文档是可鉴别的和不可伪造的，其含义是什么？

R13. 公钥加密的报文散列以何种方式比使用公钥加密报文提供更好的数字签名？

R14. 假设 certifier. com 生成一个用于 foo. com 的证书。通常整个证书将用 certifier. com 的公钥加密。这种说法是正确还是错误？

R15. 假设 Alice 有一个准备发送给任何请求者的报文。数以千计的人要获得 Alice 的报文，但每个人都要确保该报文的完整性。在这种场景下，你认为是基于 MAC 还是基于数字签名的完整性方案更为适合？为什么？

R16. 在某端点鉴别协议中，使用不重数的目的是什么？

R17. 我们说一个不重数是一个在生存期中只使用一次的值，这意味着什么？其中是指谁的生存期？

R18. 基于 HMAC 的报文完整性方案易受重放攻击影响吗？如果是，能够在方案中综合一个不重数来去除这种脆弱性吗？

8.5 ~ 8.8 节

R19. 假定 Bob 从 Alice 处接收一个 PGP 报文。Bob 怎样才能确定 Alice（而不是如 Trudy）生成了该报文？PGP 为保证报文完整性使用了 MAC 吗？

R20. 在 SSL 记录中，有一个字段用于 SSL 序号。这种说法是正确还是错误？

R21. 在 SSL 握手中随机不重数的目的是什么？

R22. 假设某 SSL 会话应用了具有 CBC 的块密码。服务器以明文向客户发送了 IV。这种说法是正确还是错误？

R23. 假设 Bob 向 Trudy 发起一条 TCP 连接，而 Trudy 正在伪装她是 Alice。在握手期间，Trudy 向 Bob 发送 Alice 的证书。在 SSL 握手算法的哪一步，Bob 将发现他没有与 Alice 通信？

R24. 考虑使用 IPsec 从主机 A 向主机 B 发送分组流。通常，为该流中的每个发送分组将创建一个新 SA。这种说法是正确还是错误？

R25. 假设在图 8-28 中总部和分支机构之间通过 IPsec 运行 TCP。如果 TCP 重新传输相同的分组，则由 R1 发送的两个对应的分组将在 ESP 首部中具有相同的序号。这种说法是正确还是错误？

R26. IKE SA 和 IPsec SA 是相同的东西。这种说法是正确还是错误？

R27. 考虑 802.11 的 WEP。假定数据是 10101100 并且密钥流是 11110000。相应的密文是什么？

R28. 在 WEP 中，在每个帧中以明文发送 IV。这种说法是正确还是错误？

8.9 节

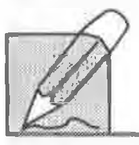
R29. 状态分组过滤器维护两个数据结构。给出它们的名字并简单地讨论它们做些什么。

R30. 考虑某传统的（无状态的）分组过滤器。该分组过滤器可能基于 TCP 标志位以及其他首部字段过滤分组。这种说法是正确还是错误？

R31. 在传统的分组过滤器中，每个接口能够具有自己的访问控制表。这种说法是正确还是错误？

R32. 为什么应用程序网关必须与分组过滤器协同工作才能有效？

R33. 基于特征的 IDS 和 IPS 检查 TCP 和 UDP 报文段的载荷。这种说法是正确还是错误？



习题

P1. 使用图 8-3 中的单码代替密码，加密报文“This is an easy problem”，并解密报文“rmij'u uamu xyj”。

P2. Trudy 使用了已知明文攻击，其中她知道了 7 个字母的（密文，明文）转换对，减少了 8.2.1 节的例子中将被检查的大约 10^9 数量级的可能替换的数量。请说明之。

P3. 考虑图 8-4 所示的多码代替密码系统。利用报文“The quick brown fox jumps over the lazy dogs”得到的明文编码，选择明文攻击足以破解所有报文吗？为什么？

P4. 考虑图 8-5 中显示的块密码。假设每个块密码 T_i 只是反转了 8 个输入比特的次序（例如，使得 11110000 变为 00001111）。进一步假设 64 比特置乱函数不修改任何比特（使得第 m 个比特的输出值等于第 m 个比特的输入值）。

a. 对于 $n=3$ 和初始 64 比特输入等于 10100000 重复了 8 次，输出的值是多少？

b. 重复（a），但此时将初始 64 比特的最后一个比特从 0 变为 1。

c. 重复（a）和（b），但此时假定 64 比特的置乱函数反转了 64 比特的次序。

P5. 考虑图 8-5 中的块密码。对于给定的“密钥”，Alice 和 Bob 将需要 8 个表，每张表 8 比特乘以 8 比特。对于 Alice（或 Bob）来说，要存储所有 8 张表，将需要多少比特的存储器？这个数如何与一个全表 64 比特的块密码所需的比特数进行比较？

P6. 考虑在表 8-1 中的 3 比特块密码。假定明文是 100100100。

a. 初始假设未使用 CBC。生成的密文是什么？

b. 假设 Trudy 嗅探该密文。假设她知道正在应用无 CBC 的一个 3 比特块密码（但不知道特定的密码），她能够推测到什么？

c. 现在假设使用 CBC，其中 IV = 111。产生的密文是什么？

P7. 如题：

a. 使用 RSA，选择 $p=3$ 和 $q=11$，采用对每个字母独立地加密的方法加密短语“dog”。对已加密报文应用解密算法恢复出原报文。

b. 重复（a），而此时加密“dog”作为一个报文 m。

P8. 考虑具有 $p=5$ 和 $q=11$ 的 RSA。

a. n 和 z 是什么？

b. 令 e 为 3。为什么这是一个对 e 的可接受的选择？

c. 求 d 使得 $de=1 \pmod z$ 和 $d<160$。

d. 使用密钥（n，e）加密报文 $m=8$。令 c 表示对应的密文。显示所有工作。提示：为了简化计算，使用如下事实。

$$[(a \bmod n)\cdot(b \bmod n)] \bmod n = (a\cdot b) \bmod n$$

P9. 在这个习题中，我们探讨 Diffie-Hellman（DH）公钥加密算法，该算法允许两个实体协商一个共享的密钥。该 DH 算法利用一个大素数 p 和另一个小于 p 的大数 g。p 和 g 都是公开的（因此攻击者将知道它们）。在 DH 中，Alice 和 Bob 每人分别独立地选择秘密密钥 S_A 和 S_B。Alice 则通过将 g 提高到 S_A 并以 p 为模来计算她的公钥 T_A。类似地，Bob 则通过将 g 提高到 S_B 并以 p 为模来计算他的公钥 T_B。此后 Alice 和 Bob 经过因特网交换他们的公钥。Alice 则通过将 T_B 提高到 S_A 并以 p 为模来计算出共享密钥 S。类似地，Bob 则通过将 T_A 提高到 S_B 并以 p 为模来计算出共享密钥 S'。

a. 证明在一般情况下，Alice 和 Bob 得到相同的对称密钥，即证明 $S=S'$。

b. 对于 $p=11$ 和 $g=2$，假定 Alice 和 Bob 分别选择私钥 $S_A=5$ 和 $S_B=12$，计算 Alice 和 Bob 的公钥 T_A 和 T_B。显示所有计算过程。

c. 接着（b），现在计算共享对称密钥 S。显示所有计算过程。

d. 提供一个时序图，显示 Diffie-Hellman 是如何能够受到中间人攻击的。该时序图应当具有 3 条垂直

线，分别对应 Alice、Bob 和攻击者 Trudy。

P10. 假定 Alice 要与采用对称密钥密码体制的 Bob 使用一个会话密钥 K_S 通信。在 8.2 节中，我们知道了如何使用公开密钥密码从 Alice 向 Bob 分发该会话密钥。在本习题中，我们探讨不使用公开密钥密码而使用一个密钥分发中心（KDC）分发会话密钥的方法。KDC 是一个与每个注册用户共享独特的秘密对称密钥的服务器。对于 Alice 和 Bob 而言，$K_{A\text{-}KDC}$ 和 $K_{B\text{-}KDC}$ 表示了这些密钥。设计一个使用 KDC 向 Alice 和 Bob 分发 K_S 的方案。你的方案应当使用三种报文来分发会话密钥：一种从 Alice 到 KDC 的报文；一种从 KDC 到 Alice 的报文；最后一种是从 Alice 到 Bob 的报文。第一种报文为 $K_{A\text{-}KDC}(A, B)$。使用标记 $K_{A\text{-}KDC}$、$K_{B\text{-}KDC}$、S、A 和 B 回答下列问题。

a. 第二种报文是什么？

b. 第三种报文是什么？

P11. 计算一个不同于图 8-8 中的两个报文的第三个报文，使该报文具有与图 8-8 中的报文相同的检验和。

P12. 假定 Alice 和 Bob 共享两个秘密密钥：一个鉴别密钥 S_1 和一个对称加密密钥 S_2。扩充图 8-9，使之提供完整性和机密性。

P13. 在 BitTorrent P2P 文件分发协议中（参见第 2 章），种子将文件分割为块，并且对等方彼此重分发这些块。不使用任何保护，一个攻击者能够容易地通过假冒善意的对等方并向洪流中的一部分对等方发送假冒块来实施破坏。这些未被怀疑的对等方则重新向其他对等方发送这些假冒块，其他对等方则将再次向甚至更多的对等方重新分发这些假冒块。因此，对于 BitTorrent 来说，采用一种机制使对等方能验证一个块的完整性，从而使得假冒块无法分发，这是至关重要的。假设当某对等方加入一个洪流时，它初始从一个完全受信任的源得到一个 .torrent 文件。描述允许对等方验证块完整性的一个简单的方案。

P14. OSPF 路由选择协议使用一个 MAC 而不是数字签名来提供报文完整性。你认为选择 MAC 而未选择数字签名的原因是什么？

P15. 考虑图 8-18 中的鉴别协议，其中 Alice 向 Bob 鉴别她自己，我们看来工作正常（即我们没有发现其中有缺陷）。现在假定当 Alice 向 Bob 鉴别她自己的同时，Bob 必须向 Alice 鉴别他自己。给出一个情况，此时 Trudy 假装是 Alice，向 Bob 鉴别她自己是Alice。（提示：该协议运行的顺序，鉴别过程可由 Trudy 或 Bob 发起，能够任意地交织在一起。特别注意 Bob 和 Alice 将使用不重数这样一个事实，如果不小心的话，能够恶意地使用相同的不重数。）

P16. 一个自然的问题是我们能否使用一个不重数的公钥密码来解决 8.4 节中的端点鉴别问题。考虑下列自然的协议：①Alice 向 Bob 发送报文“I am Alice”；②Bob 选择一个不重数并将其发送给 Alice；③Alice使用她的私钥来加密该不重数并向 Bob 发送得到的值；④Bob 对接收到的报文应用 Alice 的公钥。因此，Bob 计算 R 并鉴别了 Alice。

a. 画图表示这个协议，使用本书中应用的公钥和私钥的标记法。

b. 假定未使用证书。描述 Trudy 怎样能够通过拦截 Alice 的报文，进而对 Bob 假装她是 Alice 而成为一名“中间人”。

P17. 图 8-19 显示了 Alice 必须执行 PGP 的操作，以提供机密性、鉴别和完整性。图示出当 Bob 接收来自 Alice 的包时必须执行的对应操作。

P18. 假定 Alice 要向 Bob 发送电子邮件。Bob 具有一个公共 - 私有密钥对（K_B^+，K_B^-），并且 Alice 具有 Bob 的证书。但 Alice 不具有公钥私钥对。Alice 和 Bob（以及全世界）共享相同的散列函数$H(\cdot)$。

a. 在这种情况下，能设计一种方案使得 Bob 能够验证 Alice 创建的报文吗？如果能，用方框图显示 Alice 和 Bob 是如何做的。

b. 能设计一个对从 Alice 向 Bob 发送的报文提供机密性的方案吗？如果能，用方块图显示 Alice 和 Bob 是如何做的。

P19. 考虑下面对于某 SSL 会话的一部分的 Wireshark 输出。

a. Wireshark 分组 112 是由客户还是由服务器发送的？

b. 服务器的 IP 地址和端口号是什么？

c. 假定没有丢包和重传，由客户发送的下一个 TCP 报文段的序号将是什么？

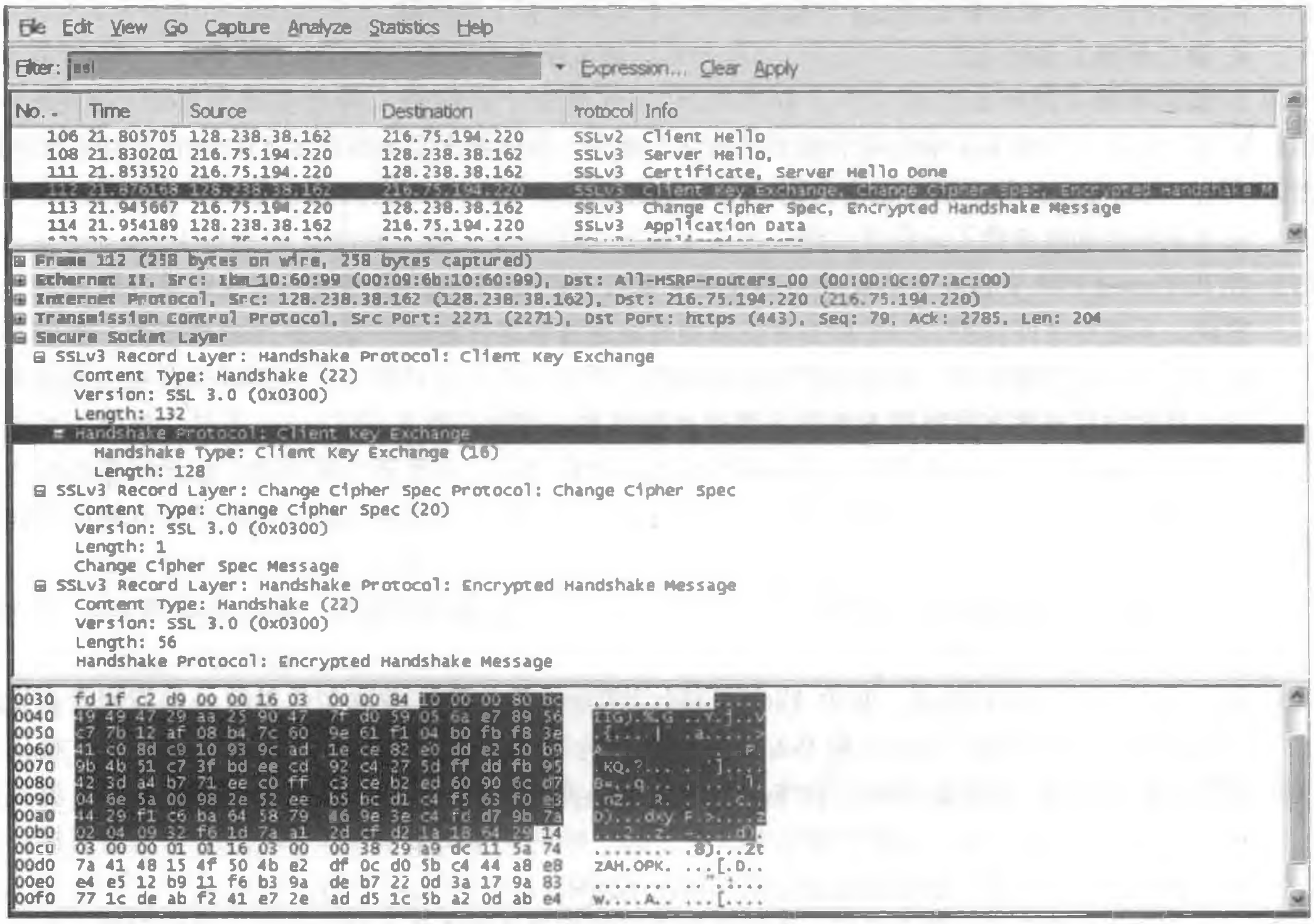
d. Wireshark 分组 112 包含了多少个 SSL 记录？

e. 分组 112 包含了一个主密钥或者一个加密的主密钥吗？或者两者都不是？

f. 假定握手类型字段是 1 字节并且每个长度字段是 3 字节，主密钥（或加密的主密钥）的第一个和最后一个字节的值是什么？

g. 客户加密的握手报文考虑了多少 SSL 记录？

h. 服务器加密的握手报文考虑了多少 SSL 记录？



（Wireshark 屏幕截图的重印获得 Wireshark 基金会的许可）

P20. 8.6.1 节中表明，不使用序号，Trudy（一名中间人）能够在一个 SSL 会话中通过互换 TCP 报文段实施破坏。Trudy 能够通过删除一个 TCP 报文段做某种类似的事情吗？在该删除攻击中，她需要做什么才能成功？它将具有什么影响？

P21. 假定 Alice 和 Bob 通过一个 SSL 会话通信。假定一个没有任何共享密钥的攻击者，在某分组流中插入一个假冒的 TCP 报文段，该报文段具有正确的 TCP 检验和及序号（以及正确的 IP 地址和端口号）。在接收侧 SSL 将接受该假冒分组并传递载荷给接收应用程序吗？为什么？

P22. 下列是有关图 8-28 的判断题。

a. 当在 172.16.1/24 中的主机向一台 Amazon.com 服务器发送一个数据报时，路由器 R1 将使用 IPsec 加密该数据报。

b. 当在 172.16.1/24 中的主机向在 172.16.2/24 中的主机发送一个数据报时，路由器 R1 将改变该 IP 数据报的源和目的地址。

c. 假定在 172.16.1/24 中的主机向在 172.16.2/24 中的 Web 服务器发起一个 TCP 连接。作为此次连接的一部分，由 R1 发送的所有数据报将在 IPv4 首部字段最左边具有协议号 50。

d. 考虑从在 172.16.1/24 中的主机向在 172.16.2/24 中的主机发送一个 TCP 报文段。假定对该报文段的应答丢失了，因此 TCP 重新发送该报文段。因为 IPsec 使用序号，R1 将不重新发送该 TCP 报文段。

P23. 考虑图 8-28 中的例子。假定 Trudy 是中间人，她能够在从 R1 和 R2 发出的数据报流中插入数据报。

作为重放攻击一部分，Trudy 发送一个从 R1 到 R2 发送的数据报的冗余副本。R2 将解密该冗余的数据报并将其转发进分支机构网络吗？如果不是，详细描述 R2 如何检测该冗余的数据报。

P24. 考虑下列伪 WEP 协议。其密钥是 4 比特，IV 是 2 比特。当产生密钥流时，IV 被附加到密钥的后面。假定共享的密钥是 1010。密钥流的 4 个可能输入如下：

101000：00101011010101010010110101001 00…

101001：10100110110010101101001001011 01…

101010：00011010001111000101001010011 11…

101011：11111010100000001010101000101 11…

假定所有报文都是 8 比特长。假定 ICV（完整性检查）是 4 比特长，并且通过用数据的后 4 比特异或数据的前 4 比特来计算。假定该伪 WEP 分组由 3 个字段组成：首先是 IV 字段，然后是报文字段，最后是 ICV 字段，这些字段中的某些被加密。

a. 我们希望使用 IV = 11 和 WEP 发送报文 m = 10100000。在这 3 个 WEP 字段中将有什么样的值？

b. 说明当接收方解密该 WEP 分组时，它恢复报文和 ICV。

c. 假定 Trudy 截获了一个 WEP 分组（并不必要使用 IV = 11）并要在向接收方转发前修改该分组。假定 Trudy 翻转了第一个 ICV 比特。假定 Trudy 并不知道用于任何 IV 的密钥流，则 Trudy 也必须翻转哪些其他比特，使得接收到的分组通过 ICV 检查？

d. 通过修改（a）中 WEP 分组中的比特，解密所生成的分组，并验证完整性检查来评价你的答案。

P25. 对于尽可能限制但能实现下列功能的一台有状态防火墙，提供一张过滤器表和一张连接表：

a. 允许所有的内部用户与外部用户创建 Telnet 会话。

b. 允许外部用户冲浪公司位于 222.22.0.12 的 Web 站点。

c. 否则阻挡所有入流量和出流量。

内部网络为 222.22/16。在你的答案中，假设连接表当前缓存了 3 个从内向外的连接。你需要虚构适当的 IP 地址和端口号。

P26. 假设 Alice 要使用 TOR 类似的服务访问 Web 站点 activist.com。该服务使用两个不串通的代理服务器 Proxy1 和 Proxy2。Alice 首先从某个中央服务器获得对 Proxy1 和 Proxy2 的证书（每个都包含一个公钥）。用 $K_1^+()$、$K_2^+()$、$K_1^-()$ 和 $K_2^-()$ 表示加密/解密时所使用的 RSA 公钥和 RSA 私钥。

a. 使用一幅时序图，提供一个（尽可能简单的）协议允许 Alice 创建一个用于 Proxy1 的共享会话密钥 S_1。$S_1(m)$ 表示为使用共享密钥 S_1 对数据 m 加密/解密。

b. 使用时序图，提供一个（尽可能简单的）协议允许 Alice 创建一个对于 Proxy2 的共享会话密钥 S_2，而不向 Proxy2 透露她的 IP 地址。

c. 现在假设创建了共享密钥 S_1 和 S_2。使用时序图提供一个协议（尽可能简单并且不使用公开密钥密码），该协议允许 Alice 从 activist.com 请求一个 html 页面而不向 Proxy2 透露她的 IP 地址，并且不向 Proxy1 透露她正在访问哪个站点。你的图应当终止在一个 HTTP 请求到达 activist.com。

Wireshark 实验

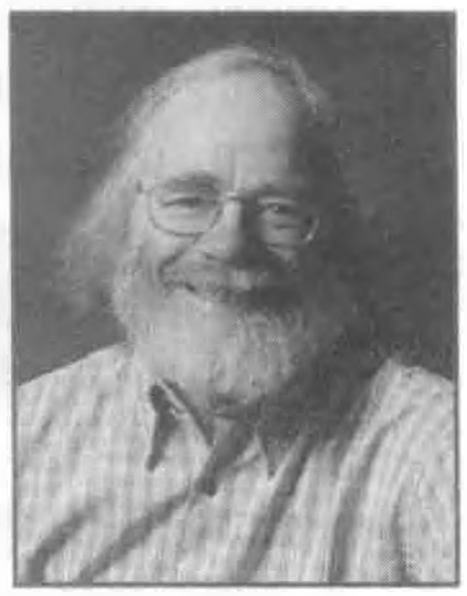
在这个实验中（与本书配套的 Web 站点有可用资源），我们研究安全套接层（SSL）协议。8.6 节讲过，使用 SSL 使得 TCP 连接更为安全，为了使因特网事务安全，实践中广泛应用了 SSL。在本实验中我们关注经 TCP 连接发送的 SSL 记录。我们将试图对每个记录定界和分类，目标是理解每个记录的工作原理和工作过程。我们研究各种 SSL 记录类型以及在 SSL 报文中的字段。通过分析你的主机与某电子商务服务器之间发送的 SSL 记录的踪迹来做这些事情。

IPsec 实验

在这个实验中（与本书配套的 Web 站点有可用资源），我们将探讨如何在 linux 装置之间创建 IPsec SA。你能够用两个普通的 linux 装置做该实验的第一部分，每个装置配有一块以太网适配器。但是对于实验

的第二部分，你将需要 4 个 linux 装置，这些装置每个都具有两块以太网适配器。在该实验的第二部分，你将在隧道模式中使用 ESP 协议创建 IPsec SA。你做实验过程是：先人工创建 SA，然后让 IKE 创建 SA。

人物专访



Steven M. Bellovin

Steven M. Bellovin 在位于新泽西州 Florham Park 的 AT&T 实验研究所的网络服务研究实验室工作多年后，成为哥伦比亚大学的教师。他的研究重点是网络和安全，以及将两者有机结合起来。1995 年，因创立了 Usenet，即第一个连接两个或多个计算机并允许用户共享信息和参与讨论的新闻组交换网络，而被授予 Usenix 终生成就奖。Steven也是国家工程学会的当选成员。他获得了哥伦比亚大学的学士学位和位于 Chapel Hill 的北卡罗来纳大学的博士学位。

- 什么原因使您决定专注于网络安全领域的研究？

听起来可能很奇怪，但是答案却很简单：只是因为感兴趣而已。我以前的背景是从事系统编程和系统管理，这很自然就发展到安全领域了。而且我一直对通信很感兴趣，这可以追溯到我还在上大学时，就兼职做系统编程方面的工作。

我在安全领域的工作持续受到两个因素的激励：一个是希望计算机有用，这意味着它们的功能不会被攻击者破坏，另一个是希望保护隐私。

- 当初您在研发 Usenet 时，您对它的愿景是什么？现在呢？

我们最初将它看作是一种能够在全国范围内讨论计算机科学和计算机编程的手段，考虑了用于事务管理和广告销售等目的的许多本地使用情况。事实上，我最初的预测是，每天从至多 50 ~ 100 个站点有 1 ~ 2 个报文。但是实际增长是与人相关的主题方面，包括（但不限于）人与计算机的相互作用。这么多年来，我喜欢的新闻组有 rec. woodworking 以及 sci. crypt。

在某种程度上，网络新闻已经被 Web 取代。如果现在要我再设计它的话，就会和那时的设计大不相同了。但是它仍然是沟通对某一主题感兴趣的大量读者的一种极好手段，而不必依赖特定的 Web 站点。

- 是否有人给过您专业上的启示和灵感？以什么样的方式呢？

Fred Brooks 教授对我的专业生涯影响重大。他是位于 Chapel Hill 的北卡罗来纳大学计算机科学系的创立者和原系主任，是研发 IBM S/360 和 OS/360 团队的管理者。他也是“The Mythical Man Mouth”（《人月神话》）的作者。最重要的是，他教给我们展望和折中的方法，即如何在现实世界环境中观察问题（不论这个现实世界比理论上的要复杂多少倍），以及在设计一种解决方案时如何平衡竞争各方的利益。大部分计算机工作都是工程性的，正确折中的艺术能够满足许多相矛盾的目标。

- 您对未来的联网和安全性的展望是什么？

到目前为止，我们所具有的安全性大多来自隔离。例如，防火墙的工作是通过切断某些机器和服务实现的。但是我们正处在增加连通性的时代，这使得隔离变得更为困难。更糟糕的是，我们的生产性系统要求的远不止是分离的部件，而需要通过网络将它们互联起来。我们面临的最大挑战之一是使所有都安全。

- 您认为在安全性方面已经取得的最大进展是什么？未来我们还能有多大作为？

至少从科学上讲，我们知道了密码学的原理。这是非常有帮助的。但是多数安全性问题因为其代码错误成堆而成为非常困难的问题。事实上，它是计算机科学中悬而未决的老问题，并且我认为该问题仍会持续。挑战在于弄明白：当我们不得不使用不安全的组件构建安全的系统时，如何才能让系统安全。我们面对硬件故障已经能够解决可靠性问题了；面对安全性问题，我们是否能够做到这一点呢？

- 您对进入因特网和网络安全领域的学生有什么忠告吗？

学习各种安全机制是件容易的事。学习如何“思维多疑”是困难的。你必须记住概率分布在下列场合并不适用，即攻击者能够发现不可能的情况。细节情况不胜枚举。

第9章

Computer Networking: A Top-Down Approach, Seventh Edition

多媒体网络

位于世界各个角落的人们懒散地靠坐在床上或乘坐公共汽车和地铁打发时间时，眼下都使用因特网来按需观看电影和电视节目。因特网电影和电视分发公司（如北美的 Netflix 和 Amazon、中国的“优酷”（Youku）和“看看”（Kankan））实际上已经成为家喻户晓的品牌。而人们不仅观看因特网视频，他们也使用诸如 YouTube 这样的站点来上载和分发用户自己生成的内容，不仅成为因特网视频的消费者，也成为视频的生产者。此外，网络应用如 Skype、Google Talk 和微信（WeChat 在中国极为流行），不仅允许人们经过因特网打“电话”，而且可以用视频和多方会议来强化这些电话。事实上，我们能够确定地预测：到当前年代末，几乎所有的视频消费和语音会话都将在因特网的端到端发生，通常更多是以无线终端方式经蜂窝和 WiFi 接入网与因特网相连接。传统的电话和广播电视正快速落伍。

本章以 9.1 节中的多媒体应用的分类方法开始。我们将看到多媒体应用能够分为*流式存储音频/视频*、*会话式 IP 音频/视频*或*流式实况音频/视频*等几类。我们将看到这些应用类型中的每一类都有自己独特的服务需求，这些需求与传统的弹性应用如电子邮件、Web 浏览和远程注册的需求差异很大。在 9.2 节中，我们较为详细地研究流式视频。我们将探讨支撑流式视频的许多基础原则，包括客户缓存、预取和对可用带宽的适应性视频质量。在 9.3 节中，我们研究会话式语音和视频。这种应用不同于弹性应用，对端到端时延高度敏感，但能够容忍偶尔的数据丢失。此时我们将研究诸如适应性播放、前向纠错和差错掩盖等技术是如何减缓网络引入的丢包和时延的。我们还将以学习案例方式审查 Skype。在 9.4 节中，我们将学习 RTP 和 SIP，这是两个用于实时会话式语音和视频应用的协议。在 9.5 节中，我们将研究网络内部的一些机制，这些机制能用于区分一类流量（如会话式语音这样的时延敏感应用）和其他类型流量（如浏览 Web 网页这样的弹性应用），并且在多类流量中提供区分服务。

9.1 多媒体网络应用

我们将多媒体网络应用定义为任何应用音频或视频的网络应用。在本节中，我们将提供多媒体应用的分类法。我们将看到在该分类法中的每类应用都具有自己独特的服务要求和设计问题集合。但在深入讨论因特网多媒体应用前，考虑音频和视频媒体自身的内在特点是有用的。

9.1.1 视频的性质

视频最为显著的特点或许是它的**高比特率**（high bit rate）。经因特网分发的视频的典型传输速率从用于低质量视频会议的 100kbps 到用于流式高分辨率电影的 3Mbps。为了比较视频带宽需求与其他因特网应用的带宽需求的不同，我们简要地考虑三个不同的用户，他们每人使用了一种不同的因特网应用。第一位用户 Frank，他打算迅速将照片张贴到他的朋友的脸书页面上。我们假设 Frank 每 10 秒查找一次新照片，并且这些照片的平均大小

是200KB。（与以前一样，我们在整个讨论中都简单地假定1KB = 8000比特。）第二位用户Martha正从因特网（“云中”）向她的智能手机流式传输音乐。我们假定Martha正在听许多MP3歌曲，一首接着一首，都以128kbps速率进行编码。第三位用户Victor则正在观看以2Mbps编码的视频。最后，我们假设所有三位用户的会话长度是4000秒（大约67分钟）。表9-1比较了这三位用户的比特率和传输的总字节。我们看到这时流式视频消耗了最多的带宽，其比特率比脸书和流式音乐应用的带宽大10倍。因此，当设计网络视频应用时，我们心中必须记住的第一件事是视频的高比特率需求。鉴于视频的流行性及其高比特率，也许不会对思科公司的以下预测感到惊讶［Cisco 2015］：到了2019年，流式视频和存储视频将大约占全球因特网流量消费的90%。

表9-1　三种因特网应用的比特率需求的比较

	比特率	67分钟传输的字节
Frank脸谱	160kbps	80MB
Martha音乐	128kbps	64MB
Victor视频	2Mbps	1GB

视频的另一种重要特点是它能被压缩，因而要在视频质量与比特率间进行折中。视频是一个图像序列，图像通常以恒定的速率显示，例如每秒24幅或30幅图像。一个没有压缩、数字编码的图像由像素阵列组成，每个像素被编码为一定数量的比特来表示亮度和颜色。在视频中有两种类型的冗余，它们都可以用来进行**视频压缩**（video compression）。空间冗余是给定图像的内部冗余。从直觉上讲，一个主要由空白组成的图像具有高度的冗余，能够有效地压缩而不会明显降低图像质量。时域冗余反映一幅图像和后续图像的重复程度。例如，如果一幅图像和后续图像完全一致，没有理由对后续图像再进行编码；相反，在编码过程中直接指出后续图像是完全一样的则更为有效。今天现成的压缩算法能够将视频压缩为所希望的任何基本比特率。当然，比特率越高，图像质量越好，总体用户视觉体验也越好。

我们也能够使用压缩来生成相同视频的**多重版本**（multiple version），每个版本有不同的质量等级。例如，我们能够使用压缩生成相同视频的三个版本，速率分别为300kbps、1Mbps和3Mbps。用户则能够根据他们的当前可用带宽来决定要观看哪个版本。具有高速因特网连接的用户可以选择3Mbps的版本；使用智能手机经过3G观看视频的用户可以选择300kbps的版本。类似地，在视频会议应用中的视频能被“动态”（on-the-fly）地压缩，以在会话用户之间给定的可用端到端带宽上提供最好的视频质量。

9.1.2　音频的性质

数字音频（包括数字化语音和音乐）的带宽需求比视频低得多。然而，数字音频具有自己独特的性质，当设计多媒体应用时必须考虑这些性质。为了理解这些性质，我们首先考虑模拟音频（由人和乐器所产生）是如何转换为数字信号的：

- 模拟音频信号首先以某种固定速率采样，例如每秒8000个样本。每个采样值是一个任意的实数。
- 然后每个采样值被“四舍五入”为有限个数值中的一个。这种操作被称为**量化**（quantization）。这些有限个数值（称为量化值）通常是2的幂，例如256个量化值。
- 每个量化值由固定数量的比特表示。例如，如果有256个量化值，那么每个值（因此每个音频采样）用一个字节来表示。所有样本的比特表示级联在一起就形成了该信号的数字表示。举例来说，如果一个模拟信号以每秒8000个样值采样，而且每个样本被量化并用8比特表示，则得到的数字信号的速率就为每秒64 000比

特。通过音频扬声器播放，这个数字信号则能够转换回来（也就是解码），形成一个模拟信号。然而，解码后的模拟信号仅是初始信号的近似，并且声音质量也许有明显的下降（例如，高频的声音可能在解码信号中丢失了）。通过增加采样速率和量化值的数量，解码信号能够更好地接近初始的模拟信号。因此（与视频一样），在解码信号的质量和比特率与数字信号存储空间之间存在一种折中。

我们刚才描述的基本编码技术称为**脉冲编码调制**（Pulse Code Modulation，PCM）。语音编码通常采用 PCM，采样速率为每秒 8000 个样本，每个样本用 8 比特表示，得到 64kbps 的速率。音频光盘（CD）也使用 PCM，采样速率为每秒 44 100 个样本，每个样本用 16 比特表示；这样使得单声道速率为 705.6kbps，立体声速率为 1.411Mbps。

然而，PCM 编码的语音和音乐很少在因特网中使用。与视频一样，取而代之的是使用压缩技术来减小流的比特速率。人类语音能被压缩到小于 10kbps 并仍然可懂。一种接近 CD 质量立体声音乐的流行压缩技术是 **MPEG 1 第 3 层**，更通常的叫法是 **MP3**。MP3 编码器通常能够压缩为许多不同的速率；128kbps 是最常使用的编码速率，并且能够产生非常小的声音失真。一种相关的标准是**高级音频编码**（Advanced Audio Coding，AAC），该标准已经随苹果公司而流行起来。与视频一样，能够以不同的比特率生成多重版本的预先录制的音频流。

尽管音频比特率通常比视频的比特率小得多，但用户通常对音频的小失误比视频的小失误更为敏感。例如，考虑在因特网上举行的视频会议。如果视频信号时不时地丢失几秒，该视频会议很可能继续进行而没有太多的用户抱怨。然而，如果音频信号经常丢失，用户就可能不得不中止该会话。

9.1.3 多媒体网络应用的类型

因特网能够支持各种各样有用的和娱乐性的多媒体应用。在本小节中，我们将多媒体应用分为三个大类：①流式存储音频/视频；②会话式 IP 语音/视频；③流式实况音频/视频。如我们很快将看到的那样，这些应用类型中的每种都有自己的服务需求和设计问题的集合。

1. 流式存储音频和视频

为使讨论具体化，我们这里聚焦流式存储视频，它通常结合了视频和音频组件。流式存储音频（例如流式音乐）非常类似于流式存储视频，尽管它的比特率通常要低得多。

在这类应用中，依赖的媒体是预先录制的视频（如电影、电视节目）预先录制的体育赛事或预先录制的用户生成的视频（如常在 YouTube 上看到的那些）。这些预先录制的视频放置在服务器上，用户向服务器发送请求按需观看视频。许多因特网公司今天提供流式视频，包括 YouTube（谷歌）、Netflix、Amazon 和 Hulu。流式存储视频具有三个关键的不同特色。

- *流*。在流式存储视频应用中，客户开始从服务器接收文件几秒之后，通常就开始播放视频。这意味着当客户正在从视频的一个位置开始播放时，与此同时正在从服务器接收该视频的后续部分。这种技术被称为**流**（streaming），它避免了在开始播放之前必须下载整个视频（并且引起一个潜在的长时延）。
- *相互作用*。因为媒体是预先录制的，用户可以对多媒体内容进行暂停、重新配置前进、重新配置倒退、快进等操作。从一个客户提出这种请求到该动作在客户端表现出来，可接受的响应时间应该小于几秒。

- 连续播放。一旦视频开始播放，它应该根据初始记录的时序进行。因此，为了在客户端播放，必须从服务器中及时接收数据；否则，用户经历视频帧停滞（这时客户等待延迟的帧）或帧跳过（这时客户漏掉延迟的帧）。

到目前为止，对流式视频最重要的性能测度是平均吞吐量。为了提供连续的播放，网络为流式应用提供的平均吞吐量必须至少与该流视频本身的比特率一样大。如我们将在9.2节所见，通过使用缓存和预取，即使在吞吐量波动的时候，提供连续播放也是可能的，只要平均吞吐量（在5～10秒区间平均）保持在视频速率之上［Wang 2008］。

对于许多流式视频应用，预先录制的视频被存储起来，并且从CDN而非从单一的数据中心流式播放。也有许多P2P视频流式应用，其中视频被存储在用户主机（对等方）上，不同视频块从可能分布在全球的不同对等方到达。在得知了因特网流式视频的性能后，我们将在9.2节更加深入地研究流式视频，特别关注客户缓存、预取、对可用带宽的适应性质量和CDN分发。

2. 会话式IP语音和视频

在因特网上的实时会话式语音通常称为**因特网电话**（Internet telephony），因为从用户的角度看，它类似于传统的电路交换电话服务。它也常被称为**IP语音**（Voice-over-IP，VoIP）。会话式视频与之类似，除了它包括参与者的语音以及视频外。今天的大多数语音和视频会话式系统允许用户生成具有三个或更多个参与者的会议。会话式语音和视频广泛地应用于今天的因特网中，因特网公司Skype、QQ和Google Talk自称每天都有数亿用户。

在第2章有关应用服务需求的讨论中（图2-4），我们确定了一些轴，服务需求可以根据它们分类。其中的两个轴（即定时考虑和数据丢失容忍度）对会话式语音和视频应用尤其重要。定时考虑是很重要的，因为音频和视频会话式应用是高度**时延敏感**（delay-sensitive）的。对于具有两个或更多个交互讲话者的会话来说，从用户讲话或移动开始到该动作显现在其他端的时延应当小于几百毫秒。对于语音，小于150ms的时延不会被人类听者觉察到，150～400ms的时延能够被接受，当时延超过400ms时，即使不会使对话变得完全无法理解，也会使语音会话变得令人沮丧。

另一个方面，会话式多媒体应用**容忍丢包**（loss-tolerant），即偶尔的丢失只会在音频/视频回放时偶尔出现干扰信号，而且这些丢失经常可以部分或者全部地隐藏。这些时延敏感但容忍丢包的特性明显不同于那些弹性数据应用（如Web浏览、电子邮件、社交网络和远程注册等）的特性。对于这些弹性应用，长时延令人烦恼，但并不是特别有害，然而传输数据的完全和完整性是首要的。我们将在9.3节中更加深入地探讨会话式语音和视频，特别关注适应性播放、前向纠错和差错掩盖是如何减缓网络引入的分组丢失和时延的。

3. 流式实况音频和视频

这种第三类应用类似于传统的电台广播和电视，只是它通过因特网来传输而已。这些应用允许用户接收从世界上任何角落发出的实况无线电广播和电视传输。今天有数以千计、遍及全球的无线电台和电视台正在因特网上广播内容。

实况是类似于广播的应用，它们经常有很多接收相同音频/视频节目的客户。在今天的因特网中，这种应用通常是用CDN来实现的（2.6节）。由于使用流式存储多媒体，网络必须为每个实况多媒体流提供大于该视频消耗速率的平均吞吐量。因为事件是直播的，尽管定时限制没有会话式语音那么严格，但时延也可能成为问题。从用户选择观看一个实况传输到播放开始，能够容忍的时延最多为10秒。我们在本书中将不涉及流式实况媒体，

因为用于流式实况媒体的许多技术（如初始缓存时延、适应性带宽使用和 CDN 分发）都类似于流式存储媒体所使用的技术。

9.2　流式存储视频

对于流式视频应用，预先录制的视频放置在服务器上，用户向这些服务器发送请求按需观看这些视频。用户可能从开始到结束都在观看视频而没有中断它，也可能在视频结束前停止观看它，或者通过暂停、重新定位到后面或前面镜头来与视频交互。流式视频系统可分为三种类型：UDP 流（UDP streaming）、HTTP 流（HTTP streaming）和适应性 HTTP 流（adaptive HTTP streaming）（参见 2.6 节）。尽管在实践中所有这三种系统都在使用，但绝大多数今天的系统应用了 HTTP 流和适应性 HTTP 流。

所有这三种形式的视频流的共同特点是广泛使用了客户端应用缓存，以此来缓解变化的端到端时延和变化的服务器和客户之间可用带宽量的影响。对于流式视频（存储的和实况的），用户通常能够容忍在客户请求某视频与该流视频在客户端播放之间有几秒的初始小时延。所以，当视频开始到达客户时，客户不必立即开始播放，反而能够在应用程序缓存中建立该视频的储备。一旦该客户建立起几秒“已缓存但尚未播放”的视频储备，客户就可以开始视频播放了。这种**客户缓存**（client buffering）具有两种重要的优点。第一，客户端缓存能够吸收服务器到客户时延中的波动。如果某特殊部分的视频数据延迟了，只要它在“接收到但尚未播放”的视频耗尽之前到达，这个长时延将不会被注意到。第二，如果服务器到客户带宽暂时低于视频消耗速率，用户能够继续享受连续的播放，只要客户应用缓存仍没有完全排尽。

图 9-1 显示了客户端的缓存。在这个例子中，假定视频以固定的比特率编码，因此每个视频块包含了能在相同固定时间量 Δ 区间播放的视频帧。服务器在 t_0 传输第一个视频块，在 $t_0+\Delta$ 传输第二个视频块，在 $t_0+2\Delta$ 传输第三个视频块等等。一旦客户开始播放，为了重新产生初始录制视频的定时，每个块应当在前一个块之后播放 Δ 时间单元。第一个视频块于 t_1 时刻到达，第二个视频块于 t_2 时刻到达。第 i 块的网络时延是服务器传输该块

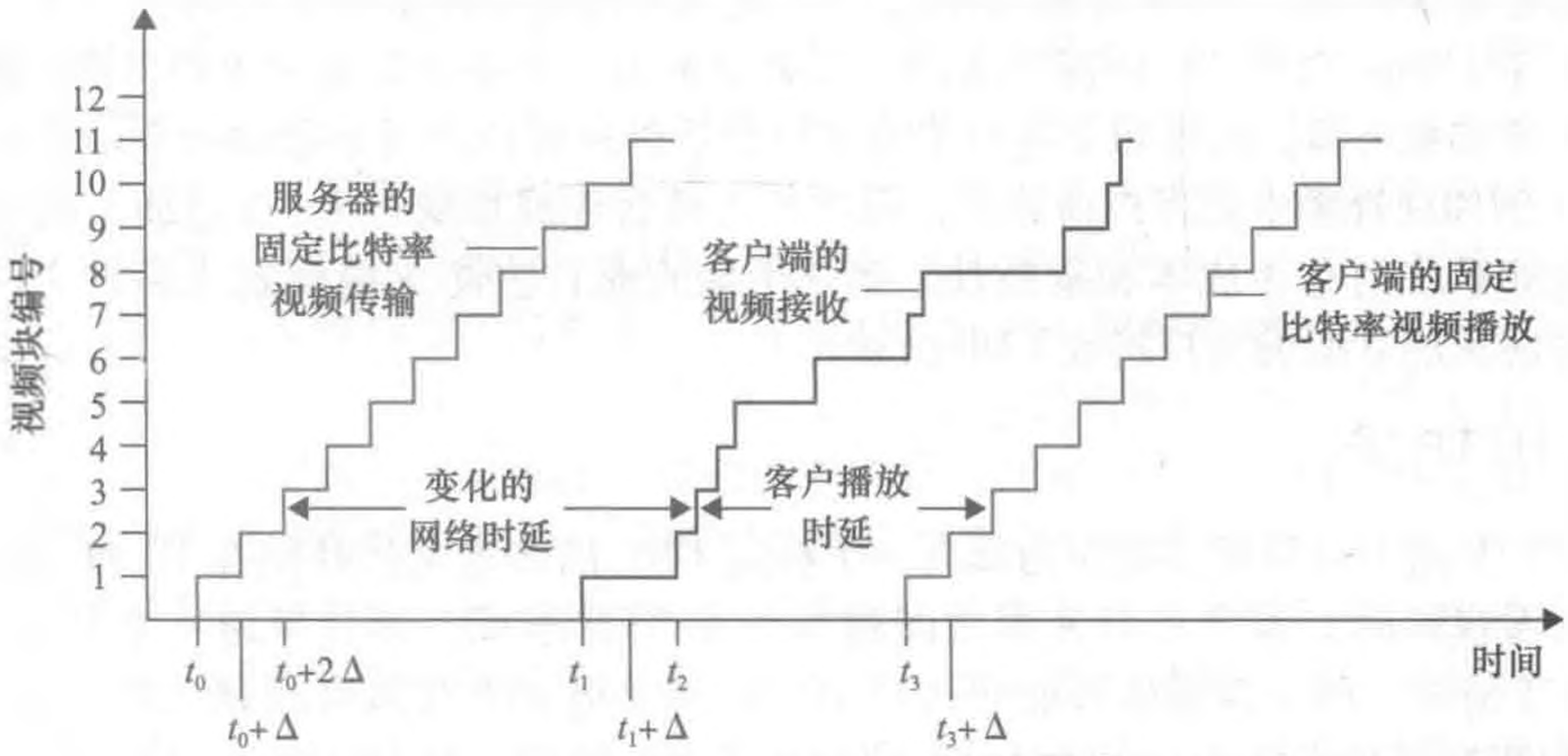


图 9-1　视频流中的客户播放时延

的时间与客户收到该块的时间之间的水平距离；注意到网络时延随视频块不同而变化。在此例子中，如果客户准备当第一块在 t_1 时刻一到达就开始播放，那么第二块将不能在 $t_1+\Delta$时刻及时到达进行播放。在这种情况下，视频播放或将停止运行（等待第二块的到

达）或可能漏掉第二块，即这两种情况都将导致不希望的播放损伤。相反，如果客户将播放延迟到 t_3 时刻开始，这时第一块到第六块都已经到达，所有已经收到的块在它们的播放时间前都能够进行周期性的播放。

9.2.1　UDP 流

我们这里仅简要讨论 UDP 流，在适当时向读者更为深入地介绍这些系统背后隐含的协议。使用 UDP 流，服务器通过 UDP 以一种稳定的速率记录下视频块，用与客户的视频消耗速率相匹配的速率传输视频。例如，如果视频消耗率是 2Mbps，每个 UDP 分组承载 8000 比特视频，则服务器将每隔（8000 比特）/（2Mbps）=4ms 向其套接字发送一个 UDP 分组。如我们在第 3 章所知，因为 UDP 未采用某种拥塞控制机制，所以服务器能够以视频的消耗速率将分组推进网络中，而无 TCP 的速率控制的限制。UDP 流通常使用很小的客户端缓存，空间维持小于 1 秒视频就足够了。

在将视频块传递给 UDP 之前，服务器将视频块封装在运输分组中，该运输分组是专门为传输音频和视频而设计的，使用了**实时传输协议**（Real-Time Transport Protocol，RTP）[RFC 3550] 或某种类似（可能是专用）的方案。我们将在 9.3 节再讨论 RTP，那时我们将在会话式语音和视频系统环境中讨论 RTP。

UDP 流的另一种不同的性质是，除了服务器到客户的视频流外，两者间还并行地维护一个单独的控制连接，通过该连接，客户可发送有关会话状态变化的命令（如暂停、重新开始、重定位等）。这种控制连接在许多方面类似于我们在第 2 章中学习的 FTP 控制连接。在本书配套 Web 网站上更为详细地解释了**实时流协议**（Real-Time Streaming Protocol，RTSP）[RFC 2326]，它是一种用于这样的控制连接的流行开放协议。

尽管 UDP 流已经在多个开源系统和专用产品中得到应用，但它有三个重大不足。首先，由于服务器和控制之间的可用带宽无法预测并且是变化的，恒定速率 UDP 流不能够提供连续的播放。例如考虑以下场景：视频消耗速率为 1Mbps，服务器到客户可用带宽通常超过 1Mbps，但每过几分钟就有几秒时间其可用带宽低于 1Mbps。在这种场景下，以 1Mbps 恒定速率经 RTP/UDP 传输视频的 UDP 流系统很可能将提供不好的用户体验，在可用带宽低于 1Mbps 之后产生停滞或漏帧。UDP 流的第二个缺点是它要求如 RTSP 服务器这样的媒体控制服务器，以对每个进行中的客户会话处理客户到服务器的交互请求和跟踪客户状态（例如在视频中的客户播放点，视频是否被暂停或播放等）。这增加了部署大规模的按需视频系统的总体成本和复杂性。第三个缺点是许多防火墙配置为阻塞 UDP 流量，防止这些防火墙后面的用户接收 UDP 视频。

9.2.2　HTTP 流

在 HTTP 流中，视频直接作为具有一个特定 URL 的普通文件存储在 HTTP 服务器上。当用户要看视频时，客户和服务器之间建立一个 TCP 连接，并且发送一个对该 URL 的 HTTP GET 请求。服务器则尽可能快地在 HTTP 响应报文中发送该视频文件，这就是说，以 TCP 拥塞控制和流控制允许的尽可能快的速率进行处理。在客户端上，字节收集在一个客户应用缓存中。一旦在缓存中字节数量超过了预先设定的阈值，该客户应用程序开始播放，具体而言，它周期性地从客户应用缓存中抓取视频帧，对帧解压缩并在用户屏幕上显示它们。

我们在第 3 章学习过，当通过 TCP 传输一个文件时，由于 TCP 的拥塞控制机制，服

务器到客户的传输速率可能变化很大。特别是，传输速率以与 TCP 拥塞控制相关联的“锯齿”形变化并非罕见。此外，分组也能由于重传机制而被大大延迟。因为 TCP 的这些特点，在 20 世纪 90 年代大多数人关于会话式的看法是流式视频将不可能在 TCP 上很好地工作。然而，随着时间的推移，流式视频系统的设计者知道了当使用了客户缓存和预取（在下面讨论）技术时，TCP 的拥塞控制和可靠数据传输机制并不一定会妨碍连续播放。

在 TCP 上使用 HTTP 也使得视频穿越防火墙和 NAT（它们常常被配置为阻挡 UDP 流量但允许大部分 HTTP 流量通过）更为容易。HTTP 流消除了因需要媒体控制服务器（如 RTSP 服务器）带来的不便，减少了在因特网上大规模部署的成本。由于所有这些优点，今天的大多数流式视频应用（包括 YouTube 和 Netflix）都使用 HTTP 流（在 TCP 上）作为它的底层流式协议。

1. 预取视频

如同我们刚才所学的那样，客户端缓存可用于缓解变化的端到端时延和变化的可用带宽的影响。在前面图 9-1 的例子中，服务器以视频播放的速率传输。然而，对于流式存储视频，客户能够尝试以高于消耗速率的速率下载视频，因此**预取**（prefetching）将来会被消耗的视频帧。该预取的视频当然存储在客户应用缓存中。这样的预取自然伴随 TCP 流出现，因为 TCP 拥塞避免机制将试图使用服务器和客户之间的所有可用带宽。

为了深入洞察预取技术，我们来举个简单的例子。假设视频消耗速率是 1Mbps，而网络从服务器到客户能够以恒定的 1.5Mbps 速率交付视频。客户则不仅能够以非常小的播放时延播放该视频，而且还能够以每秒 500Kb 的量增加缓存的视频数据。以这种方式，如果后来该客户在一段短暂时间内以小于 1Mbps 的速率接收数据，该客户由于在其缓存中的储备将能够继续提供连续的播放。[Wang 2008] 显示了当平均 TCP 吞吐量大致为媒体比特率的两倍时，TCP 流导致最小的饥饿和低缓存时延。

2. 客户应用缓存和 TCP 缓存

图 9-2 说明了客户和服务器之间 HTTP 流的交互。在服务器侧，视频文件中的白色部分已经通过服务器的套接字进行发送，而黑色部分是留下待发送的部分。在“通过套接字的门传送”之后，放置在 TCP 发送缓存中的字节在被传输进因特网之前如第 3 章所描述。在图 9-2 中，因为 TCP 发送缓存显示为满，服务器立即防止从视频文件向套接字发送更多的字节。在客户侧，客户应用程序（媒体播放器）从 TCP 接收缓存（通过其客户套接字）读出字节并将字节放入客户应用缓存中。与此同时，客户应用程序周期性地从客户应用缓存中抓取视频帧，解压缩并显示在用户屏幕上。注意到如果客户应用缓存大于该视频文件，则从服务器存储器到客户应用缓存移动字节的整个过程等价于普通文件经 HTTP 的下载过程，即客户直接将视频用 TCP 允许的尽可能快的速率从服务器中拉出来。

现在考虑在流播放期间当用户暂停视频时将发生的现象。在暂停期间，比特未从客户应用缓存中删除，甚至比特继续从服务器进入缓存。如果客户应用缓存是有限的，它可能最终会变满，这将反过来引起对服务器的“反向压力”。具体而言，一旦客户应用缓存变满，字节不再从客户 TCP 接收缓存中删除，因此它也会变满。一旦客户 TCP 接收缓存变满，字节不再从服务器 TCP 发送缓存删除，因此它也变满。一旦客户 TCP 发送缓存变满，服务器不能向套接字中发送任何更多的字节。因此，如果用户暂停视频，服务器可能被迫停止传输，在这种情况下服务器被阻塞，直到用户恢复该视频。

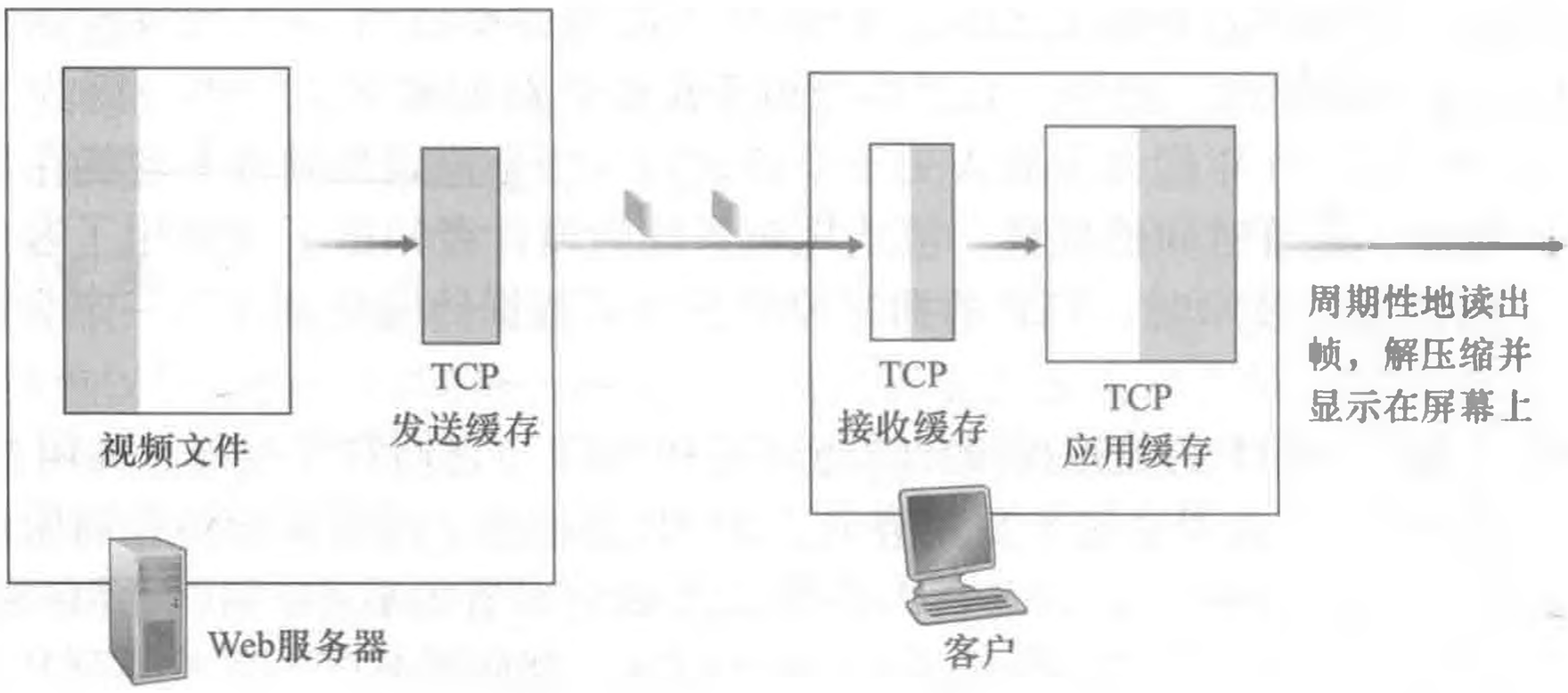


图 9-2 经 HTTP/TCP 的流式存储视频

事实上，甚至在常规的播放过程中（即没有暂停），如果客户应用缓存变满，反向压力将引起 TCP 缓存变满，这将迫使服务器降低其速率。为了决定其产生的速率，注意到当客户缓存删除 f 比特，它在客户应用缓存中产生了 f 比特的空间，这依次允许服务器发送额外的 f 比特。因此，服务器发送速率不能比客户端视频消耗速率更高。因此，当使用HTTP流时，一个满的客户应用缓存间接地对服务器到客户能够发送的视频速率施加了限制。

3. 流式视频的分析

某些简单的建模将有助于洞察由于应用缓存消耗所产生的初始播放时延和停滞。如图 9-3所示，B 表示客户应用缓存的长度（以比特计），Q 表示在客户应用缓存开始播放之前必须被缓存的比特数量。（当然，$Q < B$。）r 表示视频消耗速率，即客户在播放期间从客户应用缓存提取比特的速率。在此情况下，举例来说，如果视频的帧速率是 30 帧/秒，每（压缩）帧是 100 000 比特，则 $r = 3$Mbps。为了从细节看整体，我们将忽略 TCP 的发送和接收缓存。

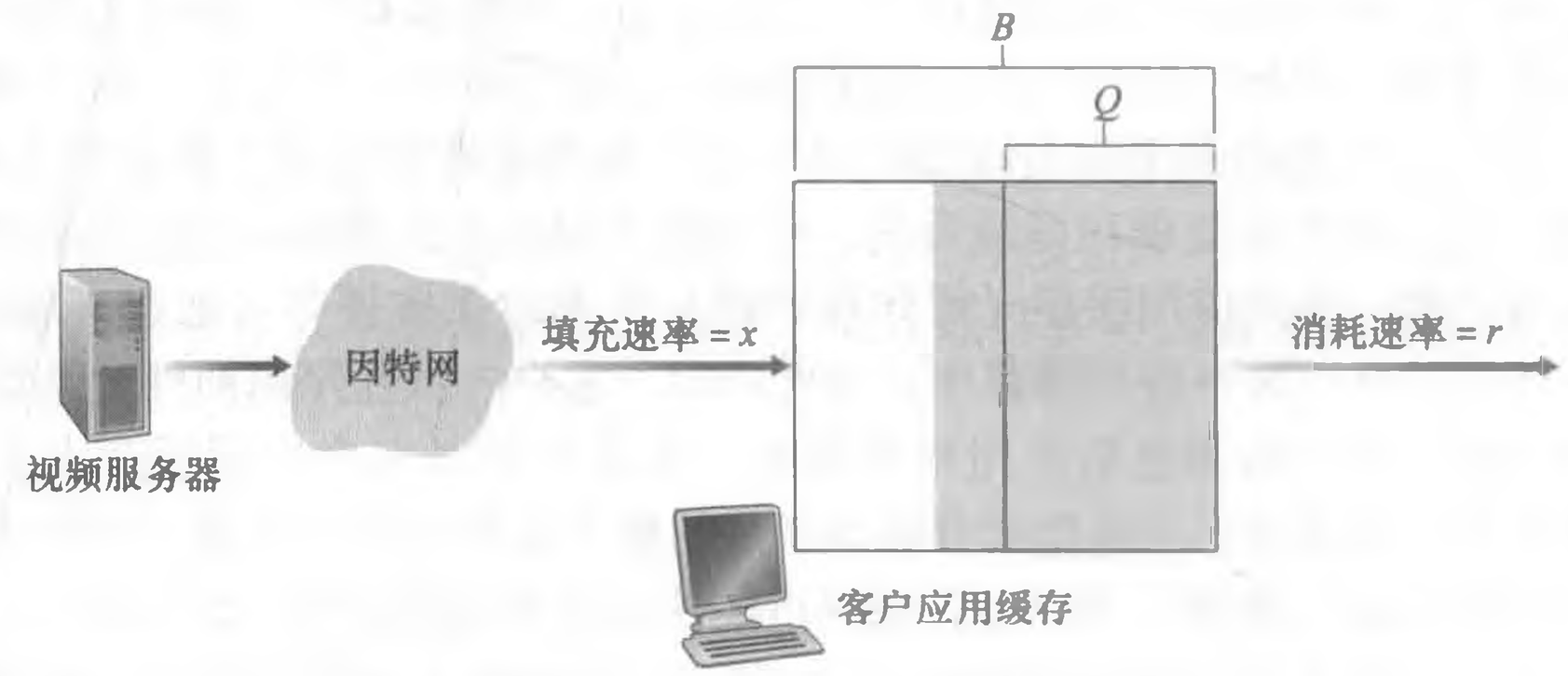


图 9-3 流式视频的客户端缓存的分析

我们假设无论何时客户缓存都为非空，服务器以一种恒定速率 x 发送比特。（这是一种显而易见的简化，因为 TCP 的发送速率由于拥塞控制而变化；在本章后面的习题中我们将考察更为真实的与时间相关的速率 $x(t)$。）假设在时刻 $t = 0$，应用缓存为空，视频开始到达客户应用缓存。我们现在问，在什么时刻 $t = t_p$ 开始播放呢？并且在播放过程中，什么时刻 $t = t_f$ 客户应用缓存变满呢？

首先，我们来确定 t_p，此时 Q 比特已经进入应用缓存并且开始播放。前面讲过比特以速率 x 到达客户应用缓存，并且在开始播放之前没有比特从其缓存中删除。所以，建立 Q 比特所需的时间（初始缓存时延）是 $t_p = Q/x$。

我们现在来决定 t_f，这是客户应用缓存变满的时刻。我们先观察，如果 $x < r$（即如果服务器发送速率小于视频消耗速率），则客户缓存将决不会满！的确，时刻 t_p 开始，缓存将以速率 r 排空并且仅以速率 $x < r$ 填充。最终客户缓存将完全排空，此时当客户缓存等待另一个 t_p 秒来建起 Q 比特的视频时，视频将在屏幕上停滞。所以，当网络中可用速率小于视频速率时，播放将在连续播放期和停滞播放期之间进行变动。在课后习题中，将请你决定每个连续播放期和停滞期的长度，它们都作为 Q、r 和 x 的函数。当 $x > r$ 时，现在我们来决定 t_f。在这种情况下，在时刻 t_p 开始，缓存以 $x - r$ 的速率从 Q 增加到 B，因为比特以速率 r 消耗但以速率 x 到达，如图 9-3 所示。有了这些提示，在课后习题中将请你决定 t_f，即客户缓存变满的时刻。注意到当网络中的可用速率大于视频速率时，在初始缓存时延后，用户将享受连续的播放直到视频结束。

4. 视频的早期中止和重定位

HTTP 流系统经常利用 HTTP GET 请求报文中的 **HTTP 字节范围首部**（HTTP byte-range header），该首部指示了客户当前要从所希望的视频中获取的字节范围。当用户要在视频中及时重定位（即跳跃）到未来点时，这特别有用。当用户重定位到一个新位置时，客户发送一个新 HTTP 请求，用字节范围首部指出服务器应当从文件的哪个字节起发送数据。当服务器接收到该新的 HTTP 请求时，它能够忘记任何较早的请求，而是由字节范围请求中指示的字节开始发送。

在我们讨论重定位主题的时候，我们简要地提及当某用户重定位到视频中的某个未来点或提前终止视频时，某些由服务器发送的已预取但尚未观看的数据将不会被观看，即导致了网络带宽和服务器资源的浪费。例如，假设在视频中的某时刻 t_0 客户缓存充满 B 比特，在此时用户重定位到视频中的某个瞬间 $t > t_0 + B/r$，然后从这点起观察视频直到结束。在这种情况下，缓存中的所有 B 比特将未被观看，用于传输这 B 比特的带宽和服务器资源完全被浪费掉了。在因特网中，有大量的带宽因提前终止而浪费，这些成本可能相当大，特别是对于无线链路［Ihm 2011］。由于这个原因，许多流系统仅使用了长度适当的客户应用缓存，或者将限制在 HTTP 请求中使用字节范围首部预取的视频数量［Rao 2011］。

重定位和提前终止可以与下列做法类比：烹调了很多肉，仅吃了一部分，并将其他都扔掉，因而浪费了食物。因此，下次你因为没有吃完所有晚餐而被父母批评浪费食物时，你能够迅速反驳：当他们在因特网上观看电影并进行带宽重定位时，他们浪费了带宽和服务器资源！但是，别人错了不等于你对了，食物和带宽都不应被浪费！

在 9.2.1 节和 9.2.2 节中，我们分别学习了 UDP 流和 HTTP 流。第三种流是经 HTTP 的动态适应流（DASH），DASH 使用了该视频的多个版本，每个版本是以不同速率压缩而成的。DASH 在 2.6.2 节中进行了详细讨论。分发存储和实况视频经常使用 CDN。CDN 在 2.6.3 节中进行了详细讨论。

9.3 IP 语音

经因特网的实时会话式语音经常被称为**因特网电话**（Internet telephony），因为从用户的视角看，它类似于传统的电路交换电话服务。它通常也被称为 **IP 语音**（Voice-over-IP，