VoIP)。在本节中我们描述 VoIP 所依据的原则和协议。会话式视频在许多方面类似于 VoIP，除了它包括参与者的视频以及他们的语音以外。为了使讨论重点突出且具体，我们这里仅关注语音，而不是语音和视频的结合。

9.3.1　尽力而为服务的限制

因特网的网络层协议 IP 提供了尽力而为的服务。那就是说服务尽全力将每个数据报从源尽可能快地移动到目的地，但是没有就在某些时延界限内使分组到达目的地或丢包百分比的限制做任何承诺。缺失这种保证对实时会话式应用的设计提出了严峻挑战，这些应用对分组时延、时延抖动和丢包非常敏感。

在本节中，我们将讨论加强尽力而为网络上的 VoIP 性能的几种方式。我们的重点将在应用层技术上，即这些技术并不要求在网络核心甚至在端系统的运输层有任何变化。为了使讨论具体，我们将讨论在一个特定的 VoIP 例子环境下尽力而为 IP 服务的限制。发送方以每秒 8000 字节的速率产生字节，且每 20ms 将字节汇聚成块。每个块和一个特殊的首部（在下面讨论）封装在一个 UDP 报文段中（通过一个到套接字接口的呼叫）。因此，一个块中的字节数为（20ms）×（8000 字节/秒）=160 字节，每 20ms 发送一个 UDP 报文段。

如果每个分组以恒定的端到端时延到达接收方，那么分组每隔 20ms 就能周期性地到达接收方。在这种理想的情况下，只要每个块一到达，接收方就能直接播放它。但不幸的是，某些分组可能丢失，大多数分组没有相同的端到端时延，即使在一个轻度拥塞的因特网中也是如此。因此，接收方必须更仔细地判断：①什么时候播放一个块；②如何处理一个丢失块。

1. 丢包

考虑由 VoIP 应用产生的一个 UDP 报文段。这个 UDP 报文段封装在 IP 数据报中。当数据报在网络中徘徊时，在它等待出链路传输时要经过路由器的缓存（即队列）。从发送方到接收方的路径上的一个或多个缓存有可能是满的，不能接纳该 IP 数据报。在这种情况下，这个 IP 数据报就被丢弃了，永远不会到达接收方的应用程序。

通过 TCP（它提供了可靠数据传输）而不是 UDP 发送分组可以消除丢失。然而，重传机制对于诸如 VoIP 这样的会话式实时音频应用，通常认为是不可接受的，因为它们增加了端到端时延［Bolot 1996］。此外，当丢包后，由于 TCP 的拥塞控制，发送方的传输速率可能减少到低于接收方的排空速率，可能导致缓存“饥饿”。这可能会对接收方的语音可理解程度产生严重影响。由于这些原因，几乎所有现有的 VoIP 应用默认运行在 UDP 上。［Baset 2006］报告称 Skype 使用了 UDP，除非用户位于阻碍 UDP 报文段的 NAT 或防火墙之后（这时使用 TCP）。

但是分组的丢失并不一定会造成人们想象中的灾难。实际上，取决于语音是如何编码和传输的以及接收方隐藏丢包的方式，1% ~20% 的丢包率是可以忍受的。例如，前向纠错（FEC）能够有助于隐藏丢包。我们后面可以看到，通过使用 FEC，将冗余信息和初始信息一起传输，以便能够从冗余信息中恢复一些丢失的初始数据。无论如何，如果发送方和接收方之间的一段或多段链路严重拥塞，丢包率超过 10% ~20%（例如在无线链路上），那么无论采取何种措施都无法获得可以接受的声音质量了。显然，尽力而为服务有它的局限性。

2. 端到端时延

端到端时延（end-to-end delay）是以下因素的总和：路由器中的传输、处理和排队时

延，链路中的传播时延和端系统的处理时延。对于实时会话式应用，例如 VoIP，听电话的人对于小于 150ms 的端到端时延是觉察不到的；在 150ms 和 400ms 之间的时延能够接受，但是不够理想；超过 400ms 的时延可能严重妨碍语音谈话的交互性。VoIP 应用程序的接收方通常忽略时延超过特定阈值（例如超过 400ms）的任何分组。因此，时延超过该阈值的分组等效于丢弃。

3. 分组时延抖动

端到端时延的一个关键成分是一个分组在网络路由器中经历的变化的排队时延。由于这些可变的时延，从在源中产生分组到它在接收方收到的这段时间，对于不同的分组可能会有波动，如图 9-1 所示。这个现象称为**时延抖动**（jitter）。举一个例子，考虑在 VoIP 应用中两个连续的分组。发送方在发送第一个分组 20ms 之后发送第二个分组。但是在接收方，这两个分组之间的间隔可能变得大于 20ms。为了理解这一点，假设第一个分组到达路由器的一个几乎为空的队列，但是恰好在第二个分组到达该队列之前，从其他源来的大量分组也到达了相同的队列。因为在这个路由器中第一个分组经受了很小的排队时延，而第二个分组经受了较大的排队时延，这两个连续的分组之间的时间间隔变得大于 20ms 了。两个连续分组的间隔也可能会小于 20ms。为了理解这种情况，再次考虑两个连续的分组。假设第一个分组加入一个有大量分组队列的队尾，并且第二个分组到达这个队列时第一个分组尚未被传输，而且来自其他源的分组尚未到达该队列。在这种情况下，这两个分组发现它们在队列中互相紧挨着。如果在路由器的出链路传输一个分组所需时间小于 20ms，则第一个分组和第二个分组的间隔就变得小于 20ms 了。

这种情况可以与在公路上开车相类比。假设你和你的朋友每人驾驶一辆车从美国的圣地亚哥到凤凰城。并且假设你和朋友有类似的驾驶风格，而且你们都以交通规则允许的 100km/h 的速度驾驶。如果你的朋友在你之前一个小时出发，那么根据干扰车流量的不同，你可能在你的朋友之后一个小时左右到达凤凰城。

如果接收方忽略了时延抖动的存在，一旦该块到达就开始播放，那么在接收方产生的音频质量很容易变得不可理解。幸运的是，时延抖动通常可以通过使用**序号**（sequence number）、**时间戳**（timestamp）和**播放时延**（playout delay）来消除，如下面所讨论的内容。

9.3.2 在接收方消除音频的时延抖动

对于 VoIP 应用，周期性地产生分组，接收方应该在存在随机网络时延抖动的情况下尝试提供播放语音块。这经常通过结合下面两种机制来实现：

- 为每个块预先计划一个**时间戳**（timestamp）。发送方用每个块产生的时刻为它加上时间印记。
- 在接收方**延迟播放**（delaying playout）块。如我们前面在图 9-1 的讨论所见，接收的音频块的播放时延必须足够长，以便大多数分组在它们的预定播放时间之前被接收到。这个播放时延可能在整个音频会话期间是固定的，或者在音频会话生命期中适应性地变化。

我们现在讨论如何结合这三种机制来减轻甚至消除时延抖动的影响。我们研究两种播放策略：固定播放时延和适应性播放时延。

1. 固定播放时延

使用固定播放时延策略，接收方试图在块产生正好 q ms 后播放它。因此如果一个块

在时刻 t 打上时间戳，接收方在时刻 $t+q$ 播放这个块，假设这个块在那个时间已经到达。在预定播放时间之后到达的分组将被丢弃，并被认为已经丢失。

q 选择什么值为好呢？尽管使用更小的 q 值可以获得更令人满意的会话体验，但 VoIP 能够支持高达约 400ms 的时延。另一方面，如果 q 比 400ms 小得多，那么由于网络引入的分组时延抖动会使许多分组可能错过了它们的预定播放时间。概括地说，如果端到端时延经常发生大的变化，用一个大的 q 更好；另一方面，如果时延很小并且时延变化也很小，用一个较小的、可能小于 150ms 的 q 更好。

在图 9-4 中说明了播放时延和丢包之间的折中。该图表示了单个话音突峰期的分组产生和播放的时间。考虑了两种不同的初始播放时延。如最左边的阶梯所示，发送方以规则的间隔（比方说每 20ms）产生一组分组。在这个话音突峰期中的第一个分组在时刻 r 被接收到。如该图所示，由于网络时延抖动，后续分组的到达间隔是不均匀的。

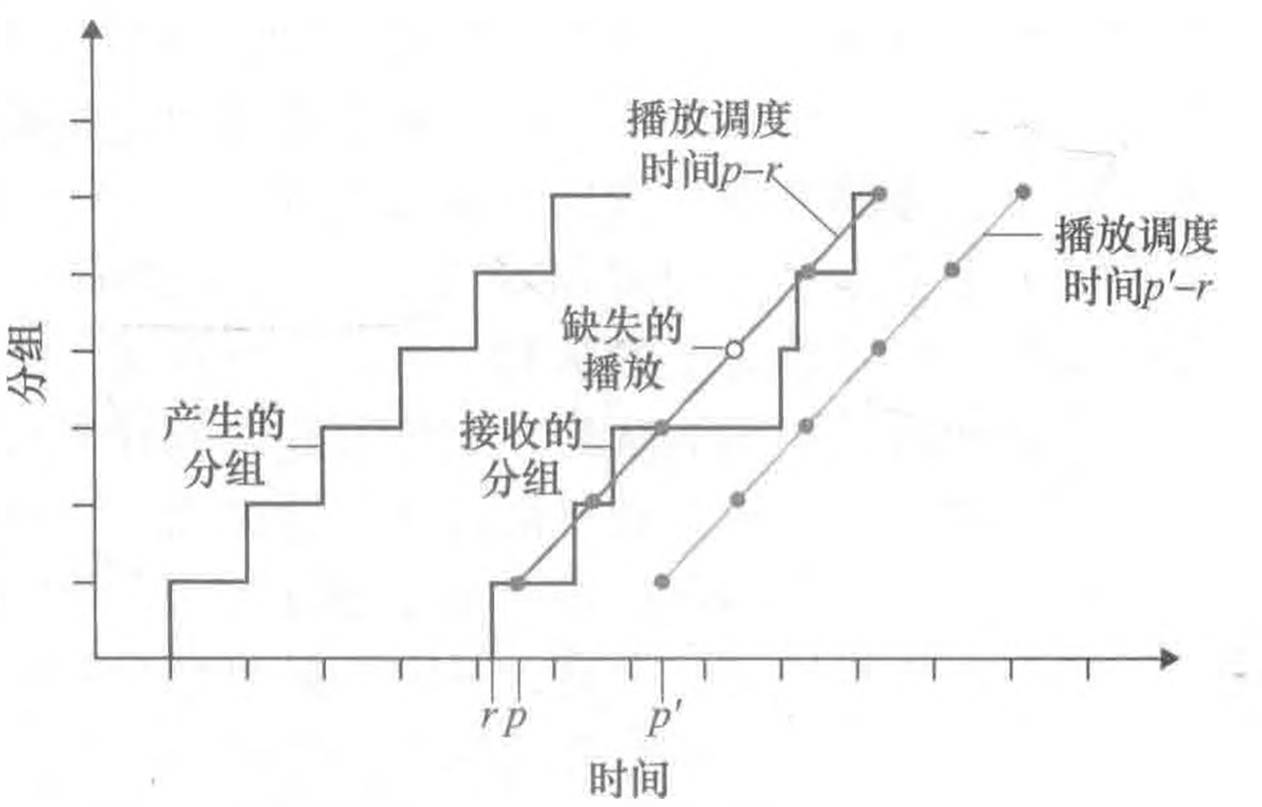

图 9-4　不同的固定播放时延情况下的丢包

对于第一个播放调度时间，固定的初始播放时延设置为 $p-r$。使用这个方案，第四个分组没有在它调度的播放时间到达，接收方认为它丢失了。对于第二个调度时间，固定的初始播放时延设置为 $p'-r$。对于这个方案，所有分组都在它们调度的播放时间之前到达，因此没有丢失。

2. 适应性播放时延

上面的例子显示了当使用固定播放时延来设计播放策略时所引起的重要的时延与丢包折中。若初始播放时延设置得比较大，大多数分组能在它们的截止时间内到达，因此存在的丢失将可忽略不计；然而，对于如 VoIP 这样的会话式服务，长时延即使不是不能忍受的，至少也是令人厌恶的。在理想情况下，我们希望播放时延最小化，使丢包低于一定百分比的限制。

处理这种折中的自然方法是估计网络时延和网络时延变化，并且在每个话音突峰期的开始相应地调整播放时延。在话音突峰期开始时适应性地调整播放时延将导致发送方静默期的压缩和拉长；然而，静默期的少量压缩和拉长在谈话中是不易觉察的。

根据［Ramjee 1994］，我们现在描述一种接收方可以用于适应性地调整它的播放时延的通用算法。为此，令

t_i = 第 i 个分组的时间戳 = 该分组在发送方产生的时间

r_i = 分组 i 被接收方接收的时间

p_i = 分组 i 在接收方播放的时间

第 i 个分组的端到端网络时延是 $r_i - t_i$。由于网络时延抖动，这个时延在不同的分组之间会发生变化。令 d_i 表示接收到第 i 个分组时的平均网络时延的估计值。这个估计值根据如下的时间戳来构造：

$$d_i = (1-u)d_{i-1} + u(r_i - t_i)$$

式中 u 是一个固定的常数（例如，$u=0.01$）。这样 d_i 是观察到的网络时延 r_1-t_1，…，r_i-t_i的一个平滑均值。这个估计值为最近观察到的网络时延设置了比过去一段时间观察到的网络时延有更大的权重。这种估值的形式不应该是完全陌生的；相似的思想在第 3 章讨论估计 TCP 往返时间时就使用过。令 v_i 表示与估计平均时延的平均时延绝对偏差的估计值。这个估计值也可从这些时间戳构建：

$$v_i = (1-u)v_{i-1} + u\,|r_i - t_i - d_i|$$

为每个接收的分组计算估计值 d_i 和 v_i，尽管它们仅能用于为任何话音突峰期的第一个分组确定播放点。

一旦计算完了这些估计值，接收方为分组播放应用下列的算法。如果分组 i 是一个话音突峰期的第一个分组，它的播放时间 p_i 计算如下：

$$p_i = t_i + d_i + Kv_i$$

这里 K 是一个正的常数（例如 $K=4$）。Kv_i 项的目的是给将来设置足够大的播放时间，以便话音突峰期中只有一小部分到达的分组由于迟到而丢失。在一个话音突峰期中任何后续分组的播放点被计算为对于这个话音突峰期的第一个分组播放时间点的偏移。特别是，令

$$q_i = p_i - t_i$$

表示从话音突峰期的第一个分组产生到它播放的时间长度。如果分组 j 也属于这个话音突峰期，它播放的时刻是

$$p_j = t_j + q_i$$

刚才描述的算法在假设接收方能够辨明一个分组是否是话音突峰期中的第一个分组的情况下很有意义。这能够通过检查每个接收到的分组中的信号能量来做到。

9.3.3 从丢包中恢复

我们已经较为详细地讨论了一个 VoIP 应用能够怎样处理分组时延抖动。我们现在简要地描述在存在丢包的情况下几种试图保护可接受的音频质量的方案。这样的方案被称为**丢包恢复方案**（loss recovery scheme）。这里我们定义了广义的丢包：如果某分组不能到达接收方或者在它调度的播放时间之后才到达，该分组则丢失。我们再次用 VoIP 例子作为描述丢包恢复方案的环境。

如在本节开始提到的那样，在诸如 VoIP 等会话式实时应用中，重传丢失的分组通常是不可行的。的确，重传一个已经错过了播放截止时间的分组是绝对没有意义的。而且重传一个在路由器队列溢出的分组通常不能足够快地完成。由于这些考虑，VoIP 应用通常使用某种类型的丢包预期方案。两种类型的丢包预期方案是**前向纠错**（Forward Error Correction，FEC）与**交织**（interleaving）。

1. 前向纠错

FEC 的基本思想是给初始的分组流增加冗余信息。以稍微增加传输速率为代价，这些冗余信息可以用来重建一些丢失分组的近似或者准确版本。根据文献［Bolot 1996］和［Perkins 1998］，我们现在概括了两种简单的 FEC 机制。第一种机制是每发送 n 个块之后发送一个冗余编码的块。这个冗余块通过异或 n 个初始块来获得［Shacham 1990］。以这种方式，在这 $n+1$ 个分组的组中，如果任何一个分组丢失，接收方能够完全重建丢失的分组。但是如果这一组中有两个或更多分组丢失，接收方则无法重建丢失的分组。通过让

组的长度 $n+1$ 比较小，当丢失不是很多时，大部分丢失分组都可以恢复。然而组的长度越小，相对增加的传输速率就越大。特别是若传输速率以 $1/n$ 因子增加的话，例如 $n=3$，则传输速率增加 33%。此外，这个简单的方案增加了播放时延，因为接收方在能够开始播放之前，必须等待收到整个组的分组。有关 FEC 在多媒体传输上如何工作的更多实践细节参见［RFC 5109］。

第二个 FEC 机制是发送一个较低分辨率的音频流作为冗余信息。例如，发送方可能创建一个标称的音频流和一个相应的低分辨率、低比特率的音频流。（这个标称流可能是一个 64kbps 的 PCM 编码，而这个较低质量的流可能是一个 13kbps 的 GSM 编码。）这个低比特率流被认为是冗余信息。如图 9-5 所示，发送方通过从这个标称流中取出第 n 个块并附加上第（$n-1$）个块的冗余信息，以构建第 n 个分组。以这种方式，只要没有连续分组的丢失，接收方都可以通过播放和后续分组一起到达的低比特率编码块来隐藏丢失。当然，低比特率块比标称块的质量要低。然而，在一个流主要是由高质量块组成、偶尔出现低质量块并且没有丢失的块的情况下，其整体的音频质量良好。注意到在这种方案中，接收方在播放前只需接收两个分组，因此增加的时延小。此外，如果低比特率编码比标称编码少得多，那么传输速率的额外增加并不大。

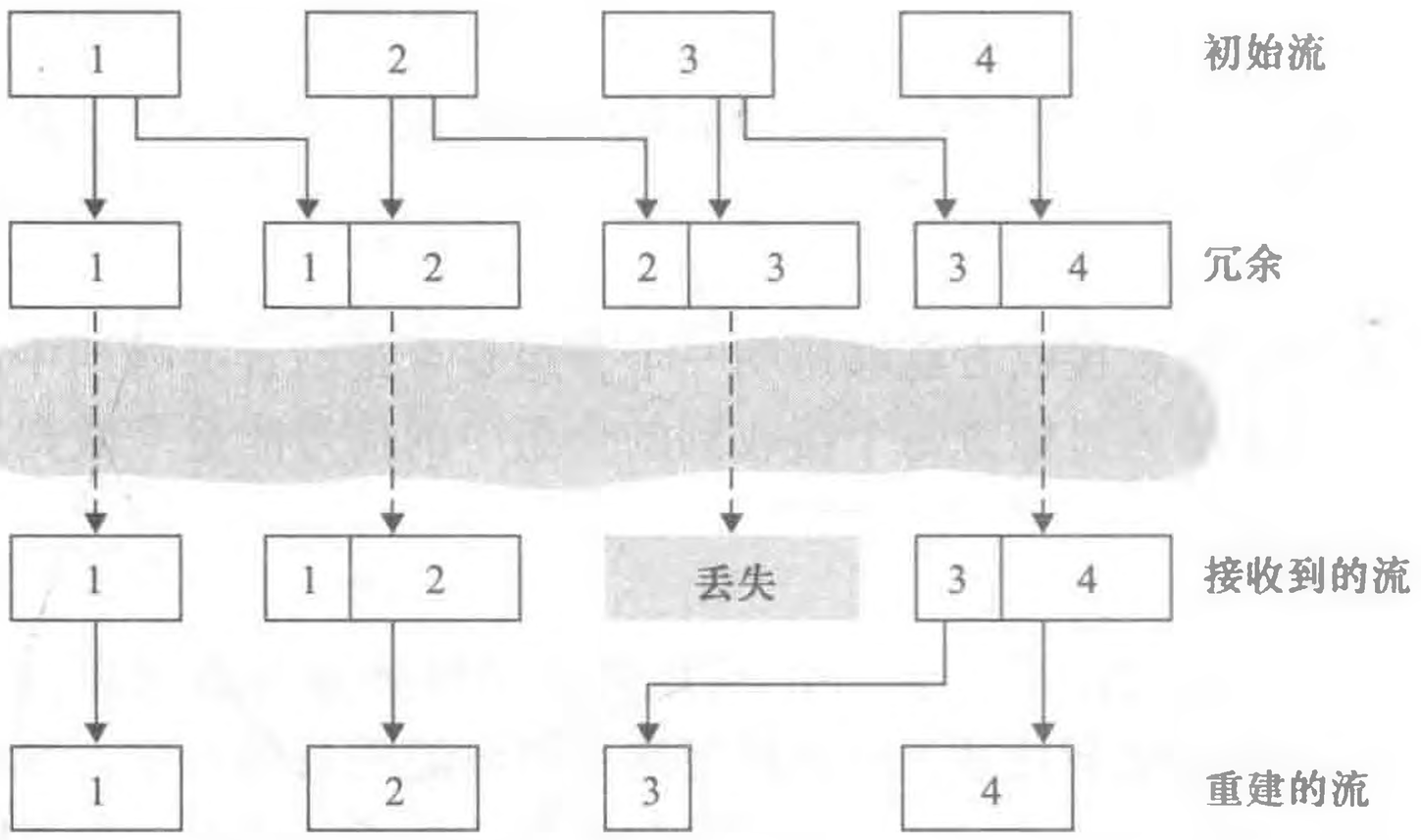

图 9-5　捎带低质量的冗余信息

为了处理连续丢失，我们能够使用率一个简单的修正方案。发送方不再仅为第 n 个标称块附加上第（$n-1$）个低比特率块，而是附加上第（$n-1$）个和第（$n-2$）个低比特率块，或者附加第（$n-1$）个和第（$n-3$）个低比特率块等等。通过给每个标称块附加上更多低比特率块，在各种恶劣的尽力而为服务环境下接收方的音频质量变得可接受。在另一方面，附加的块增加了传输带宽和播放时延。

2. 交织

作为冗余传输的另一种替代方案，VoIP 应用可以发送交织的音频。如图 9-6 所示，发送方在传输之前对音频数据单元重新排序，使得最初相邻的单元在传输流中以一定距离分离开来。交织可以减轻丢包的影响。例如，如果每个单元长为 5ms，块是 20ms（也就是每个块 4 个单元），那么第一个块可能包含 1、5、9 和 13 单元；第二个块可能包含 2、6、10 和 14 单元等等。图 9-6 显示了一个交织流的单个丢包导致重建流中的多个小间隙，这与在非交织流中将会导致单个大间隙形成对照。

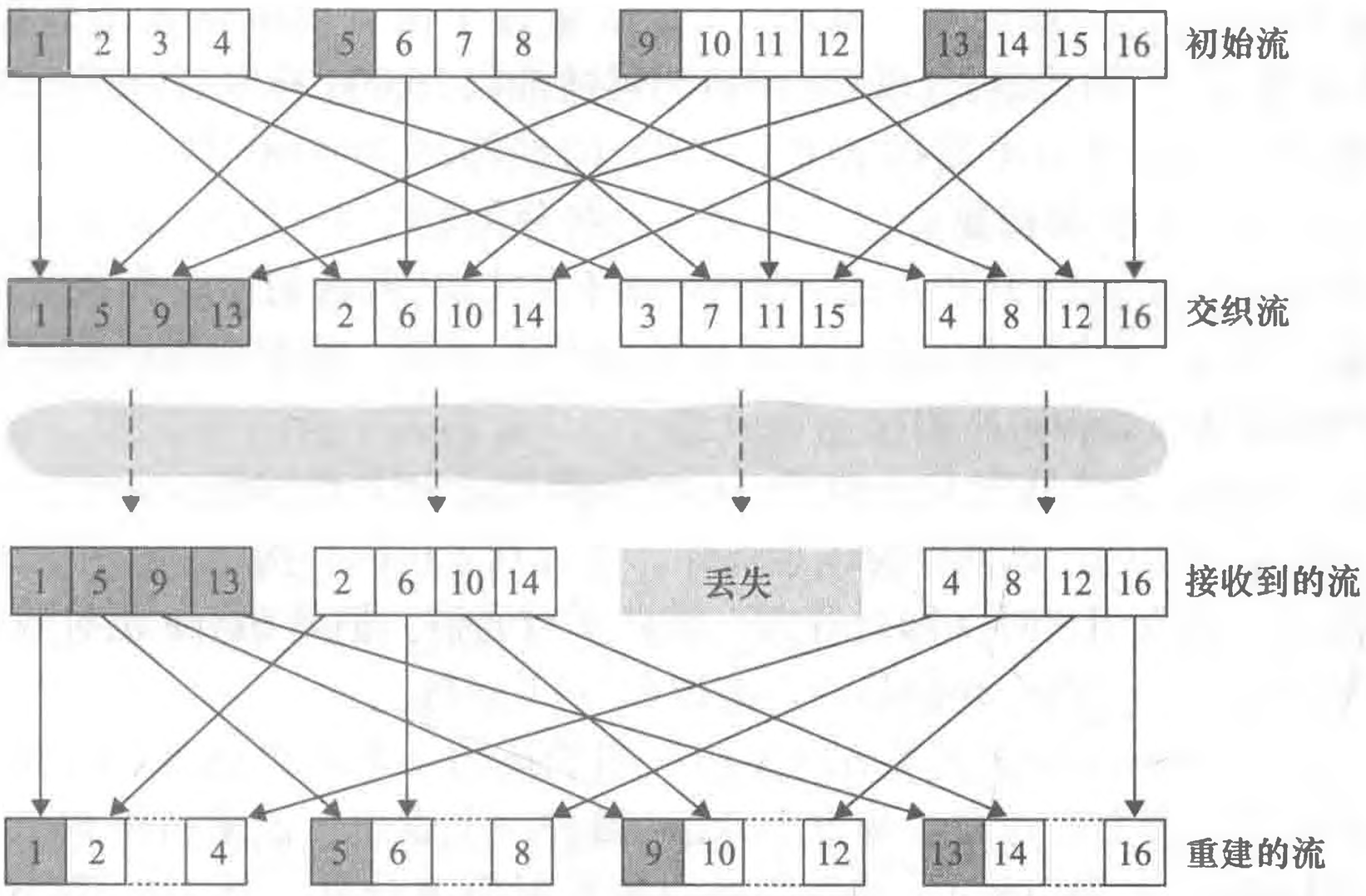

图 9-6 发送交织音频

交织能够明显地提高音频流可感觉到的质量［Perkins 1998］。它的开销也较低。交织明显的缺点是增加了时延。这限制了它在如 VoIP 这样的会话式应用中的使用，然而它能够很好地处理流式存储音频。交织的一个主要优点是它不增加流的带宽需求。

3. 差错掩盖

差错掩盖方案试图为丢失的分组产生一个与初始分组类似的替代物。正如在［Perkins 1998］中讨论的那样，因为音频信号（特别是语音）呈现出大量的短期自相似性，故该方案是可行的。正因为如此，这些技术适合于工作在相对小的丢包率（低于 15%）和小分组（4～40ms）的情况。当丢失长度接近音素的长度（5～100ms）时，这些技术就失效了，因为整个音素可能被听者错过。

也许基于接收方的恢复的最简单方式是分组重复。即用在丢失之前刚到达的分组的副本来代替丢失的分组。这种方法的计算复杂度低，并且工作得相当好。基于接收方恢复的另一种形式是内插法，它使用在丢失之前和之后的音频内插形成一个合适分组来隐藏丢失。内插法比分组重复稍微好一些，但是显然需要更高的计算强度［Perkins 1998］。

9.3.4 学习案例：使用 Skype 的 VoIP

Skype 是一个非常流行的 VoIP 应用，每天都有超过 5000 万个活跃的账户。Skype 除了提供主机到主机的 VoIP 服务，还提供主机到电话的服务，电话到主机的服务，以及多方主机到主机的视频会议服务。（这里，主机仍然是任一种因特网连接 IP 设备，包括 PC、平板电脑和智能手机。）Skype 于 2011 年被微软公司收购。

因为 Skype 协议是专用的，并且因为所有 Skype 的控制和媒体分组是加密的，所以精确地确定 Skype 的工作过程是非常困难的。无论如何，从 Skype 的 Web 网站和几项测量研究，研究人员已经知道了 Skype 总体上是怎样工作的［Baset 2006；Guha 2006；Chen 2006；Suh 2006；Ren 2006；Zhang X 2012］。对于语音和视频，Skype 客户都有许多自行支配的不同编解码器，这些编解码器能够以宽泛的速率和质量对媒体进行编码。例如，测量表明 Skype 的视频速率从用于低质量会话的低至 30kbps 到用于高质量会话的高至 1Mbps

左右［Zhang X 2012］。一般而言，Skype 语音质量好于由有线电话系统提供的“POTS（简单老式电话服务）”的质量。（Skype 编解码器通常以 16 000 样本/秒或更高速率对语音抽样，这提供比 POTS 更为丰富的音色，POTS 的抽样率为 8000/秒。）在默认状态下，Skype 通过 UDP 发送音频和视频分组。然而，控制分组经 TCP 发送，并且当防火墙阻挡 UDP 流时，媒体分组也通过 TCP 发送。Skype 对于经 UDP 发送的语音和视频流使用 FEC 处理丢包恢复。Skype 客户还通过改变视频质量和 FEC 开销，使它所发送的音频和视频流适应当前的网络情况［Zhang X 2012］。

Skype 以一些创新方式使用 P2P 技术，很好地阐述了 P2P 是如何应用于除内容分发和文件共享之外的应用中的。如同即时讯息那样，主机到主机因特网电话应用的核心内在地应用了 P2P 技术，因为用户对（即对等方）彼此实时通信。但是 Skype 也对两个其他重要功能应用了 P2P 技术，这两个功能是用户定位和 NAT 穿越。

如图 9-7 所示，Skype 中的对等方（主机）组织成为一个等级制覆盖网络，其中每个对等方分类为超级对等方和普通对等方。Skype 维护一个索引，该索引将 Skype 用户名映射为当前的 IP 地址（和端口号）。该索引经过超级对等方分发。当 Alice 要呼叫 Bob 时，她的客户搜索该分布式索引以决定 Bob 的当前 IP 地址。因为 Skype 协议是专用的，所以当前并不知道该索引映射是怎样跨越这些超级对等方进行组织的，尽管采用某种形式的 DHT 组织结构是非常可能的。

P2P 技术也被用于 Skype **中继**（relay）中，中继对于创建家庭网络中主机之间的呼叫是有用的。许多家庭网络配置提供通过 NAT 接入因特网，如第 4 章所讨论的那样。前面讲过 NAT 防止来自家庭网络外部的主机发起的对家庭网络内部主机的连接。如果两个 Skype 呼叫方都具有 NAT，则存在一个问题，即任一方都不能接受由其他一方发起的呼叫，使得呼叫看起来不可能实现。明智地使用超级对等方和中继很好地解决了这个问题。假设当 Alice 注册进入系统，她被指派了一个非 NAT 的超级对等方并对那个超级对等方发起一个会话。（因为是 Alice 发起了该会话，所以她的 NAT 允许该会话。）这个会话允许 Alice 和她的超级对等方交换控制报文。当 Bob 注册进入系统时发生了同样的事情。此时，当 Alice 要呼叫 Bob，她通知她的超级对等方，超级对等方依次通知 Bob 的超级对等方，Bob 的超级对等方依次通知 Bob 说“Alice 的入呼叫到了”。如果 Bob 接受了该呼叫，这两个超级对等方选择一个第三方非 NAT 超级对等方（即中继对等方），中继对等方的工作是中继 Alice 和 Bob 的数据。Alice 和 Bob 的超级对等方则分别指示 Alice 和 Bob 与该中继发起会话。如图 9-7 所示，Alice 则经过“Alice 到中继”连接向该中继发送语音分组（该连接由 Alice 发起），并且该中继经“中继到 Bob”连接转发这些分组（该连接由 Bob 发起）；从 Bob 到 Alice 的分组反方向地流经相同的两条中继连接。瞧！Bob 和 Alice 有了一条端到端连接，即使他们都不能

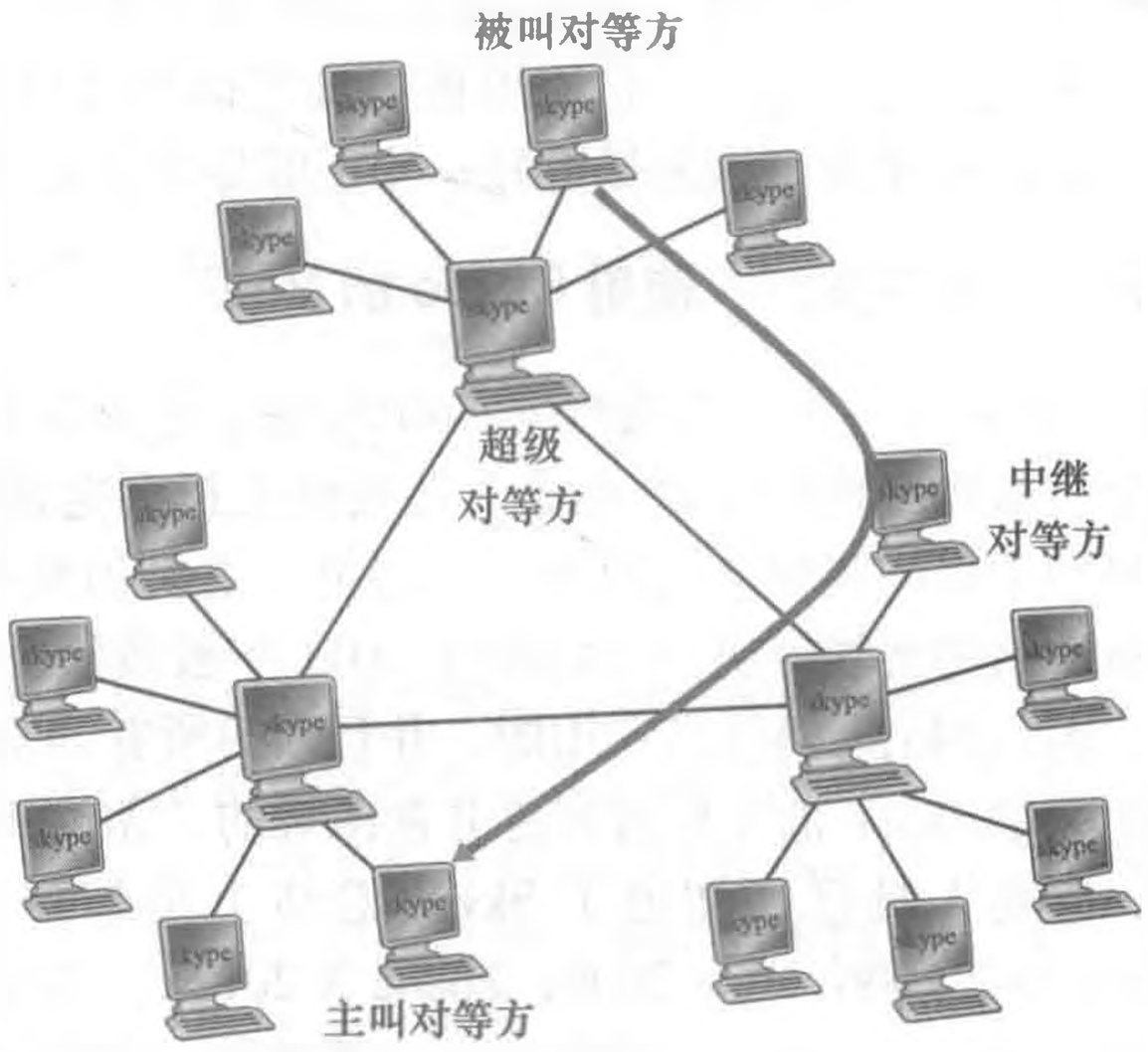

图 9-7　Skype 对等方

接受一条源于外部的会话。

到现在为止，我们有关 Skype 的讨论关注涉及两人的呼叫。现在我们观察多方音频会议呼叫。对于 $N>2$ 个参与者，如果每个用户希望向每个其他 $N-1$ 个用户发送它的音频流的一个副本，则为支持该音频会议总共 $N(N-1)$ 个音频流将需要发送到网络中去。为了减少这种带宽使用，Skype 应用了一种明智的发送技术。具体而言，每个用户向会议发起方发送它的音频流。会议发起方将这些音频流结合为一个流（基本上是将所有的音频信号加在一起），然后再向每个其他 $N-1$ 个参与者发送每个结合流的一个副本。以这种方式，流的数量被减少到 $2(N-1)$ 条。对普通的两人视频会议，Skype 路由对等方到对等方呼叫，除非需要 NAT 穿越，此时呼叫通过一个非 NAT 对等方中继，如前面所述。对于一个涉及 $N>2$ 个参与者的视频会议呼叫，由于视频媒体的性质，Skype 不像对语音呼叫那样在一个位置将呼叫结合进一条流中，然后将流向所有参与者重新分发。相反，每个参与者的视频流被路由到一个服务器集群（2011 年该服务器集群位于爱沙尼亚），该集群依次将 $N-1$ 个其他参与者的 $N-1$ 条流中继到每个参与者［Zhang X 2012］。你可能想知道为什么每个参与者向服务器而不是向每个其他 $N-1$ 其他参与者直接发送其视频流的副本呢？的确，对两种方法而言，$N(N-1)$ 个视频流正由会议中的 N 个参与者共同接收。其原因是，在大多数接入链路中上行链路带宽比下行链路带宽要低得多，上行链路可能不能支持使用 P2P 方法的 $N-1$ 条流。

诸如 Skype、QQ 和 Google Talk 等 VoIP 系统引发了对隐私性的新关注。具体而言，当 Alice和 Bob 经过 VoIP 通信，Alice 能够嗅探到 Bob 的 IP 地址，进而使用地理定位服务［MaxMind 2016；Quova 2016］来确定 Bob 的当前位置和 ISP（例如他的工作或家庭 ISP）。事实上，Alice 使用 Skype 能够在呼叫创建期间阻挡特定的分组传输，这样她获得 Bob 当前（比如说每个小时）的 IP 地址，而 Bob 并不知道他正被跟踪，并且 Alice 不在 Bob 的接触列表上。此外，从 Skype 发现的 IP 地址能被关联到在 BitTorrent 中发现的 IP 地址，因此 Alice 能够确定 Bob 正在下载的文件［LeBlond 2011］。此外，通过进行流中分组长度的流量分析，可能部分解密一个 Skype 呼叫［White 2011］。

9.4 实时会话式应用的协议

实时会话式应用（包括 VoIP 和视频会议）引人入胜并且非常流行。因此标准机构如 IETF 和 ITU 多年来一直忙于（而且要继续忙下去！）苦心推敲这类应用的标准是毫不奇怪的。借助于实时会话式应用的适当标准，各个独立公司正在创造新的能够互相操作的产品。在本节中，我们探讨用于实时会话式应用的 RTP 和 SIP。这两个标准正广泛地应用于工业产品中。

9.4.1 RTP

在前一节中，我们知道 VoIP 应用的发送端在将块传递给运输层之前为它们附加上首部字段。这些首部字段包括了序号和时间戳。因为大多数多媒体网络应用能够利用序号和时间戳，因此有一个包括音频/视频数据、序号、时间戳以及其他潜在有用字段的标准分组结构是方便的。定义在 RFC 3550 中的 RTP 就是这样一个标准。RTP 能够用于传输通用格式，如用于声音的 PCM、ACC 和 MP3，用于视频的 MPEG 和 H.263。它也可以用于传输专用的声音和视频格式。目前，RTP 在许多产品和研究原型中得到广泛实现。它也是其他

重要的实时交互协议（如 STP）的补充。

本节我们介绍 RTP。我们也鼓励读者去访问 Henning Schulzrinne 的 RTP 站点［Schulzrinne-RTP 2012］，该网站提供有关这个主题的很多信息。读者也可以访问 RAT 站点［RAT 2012］，它记载了使用 RTP 的 VoIP 应用。

1. RTP 基础

RTP 通常运行在 UDP 之上。发送端在 RTP 分组中封装媒体块，然后在 UDP 报文段中封装该分组，然后将该报文段递交给 IP。接收端从 UDP 报文段中提取出这个 RTP 分组，然后从 RTP 分组中提取出媒体块，并将这个块传递给媒体播放器来解码和呈现。

举例来说，考虑使用 RTP 来传输语音。假设语音源采用了 64kbps 的 PCM 编码（也就是采样、量化和数字化）。再假设应用程序在 20ms 块中收集这些编码数据，也就是一个块中有 160 字节。发送端在每个语音数据块的前面加上一个 **RTP 首部**（RTP header），这个首部包括音频编码的类型、序号和时间戳。RTP 首部通常是 12 字节。音频块和 RTP 首部一起形成 **RTP 分组**（RTP packet）。然后向 UDP 套接字接口发送该 RTP 分组。在接收端，应用程序从它的套接字接口收到该 RTP 分组，从 RTP 分组中提取出该音频块，并且使用 RTP 分组的首部字段来适当地解码和播放该音频块。

如果一个应用程序集成了 RTP，而非一种提供负载类型、序号或者时间戳的专用方案，则该应用程序将更容易和其他网络的多媒体应用程序互操作。例如，如果两个不同的公司都开发 VoIP 软件，并且都在它们的产品中集成了 RTP，则希望使用一种 VoIP 产品的用户能够和使用另一种 VoIP 产品的用户进行通信。在 9.4.2 节，我们将看到 RTP 经常和 SIP 一起使用，SIP 是一种因特网电话的重要标准。

应该强调的是，RTP 并不提供任何机制来确保数据的及时交付，或者提供其他服务质量（QoS）保证；它甚至不保证分组的交付，或防止分组的失序交付。RTP 封装的东西确仅为端系统所见。路由器不区分携带 RTP 分组的 IP 数据报和不携带 RTP 分组的 IP 数据报。

RTP 允许为每个源（例如一架照相机或者一个麦克风）分配一个它自己的独立 RTP 分组流。例如，对于在两个参与者之间的一个视频会议，可能打开 4 个 RTP 流，即两个流传输音频（一个方向一个），两个流传输视频（也是一个方向一个）。然而，在编码过程中很多流行的编码技术（包括 MPEG1 和 MPEG2）将音频和视频捆绑在单个流中。当音频和视频与编码器捆绑时，每个方向只产生一个 RTP 流。

RTP 分组并非限用于单播应用，它们也可以经过一对多和多对多的多播树发送。对于一个多对多的多播会话，所有的会话发送方和源通常使用同样的多播组来发送它们的 RTP 流。在一起使用的 RTP 多播流，例如在视频会议应用中从多个发送方发出的音频和视频流，同属于一个 **RTP 会话**（RTP session）。

2. RTP 分组首部字段

如图 9-8 所示，4 个主要的 RTP 分组首部字段是有效载荷类型、序号、时间戳和源标识符字段。

图 9-8　RTP 首部字段

RTP 分组中的有效载荷类型字段的长度是 7 比特。对于音频流，有效载荷类型字段用于指示所使用的音频编码类型（例如 PCM、适应性增量调制、线性预测编码）。如果发送方在会话过程中决定改变编码，发送方可以通过该有效载荷类型字段来通知接收方这种变化。发送方可能要通过改变该编码来提高语音质量或者减小 RTP 流比特率。表 9-2 列出了当前 RTP 支持的一些音频有效载荷类型。

表 9-2　RTP 支持的一些音频有效载荷类型

有效载荷类型编号	音频格式	采样速率	速率
0	PCMμ 律	8kHz	64kbps
1	1016	8kHz	4.8kbps
3	GSM	8kHz	13kbps
7	LPC	8kHz	2.4kbps
9	G.722	16kHz	48～64kbps
14	MPEG 音频	90kHz	—
15	G.728	8kHz	16kbps

对于一个视频流，有效载荷类型用于指示视频编码类型（例如运动 JPEG、MPEG1、MPEG2、H.261）。发送方也可以在会话期间动态改变视频编码。表 9-3 列出了当前 RTP 支持的一些视频有效载荷类型。

表 9-3　RTP 支持的一些视频有效载荷类型

有效载荷类型编号	视频格式	有效载荷类型编号	视频格式
26	运动 JPEG	32	MPEG1 视频
31	H.261	33	MPEG2 视频

其他重要的字段如下：

- 序号字段。序号字段长为 16 比特。每发送一个 RTP 分组则该序号增加 1，而且接收方可以用该序号来检测丢包和恢复分组序列。例如，如果应用的接收方收到的 RTP 分组流在序号 86 和 89 之间存在一个间隙，那么接收方则知道分组 87 和 88 丢失了。那么接收方能够设法来隐藏该丢失数据。
- 时间戳字段。时间戳字段长 32 比特。它反映了 RTP 数据分组中的第一个字节的采样时刻。如我们在上一节所见，接收方能够使用时间戳来去除网络中引入的分组时延抖动，提供接收方的同步播放。时间戳是从发送方的采样时钟中获得的。举例来说，对于音频的每个采样周期（例如对于 8kHz 的采样时钟每 125μs 为一个周期）时间戳时钟增加 1；如果该音频应用产生由 160 个编码采样组成的块的话，那么当源激活时，对每个 RTP 分组则时间戳增加 160。即使源未激活，该时间戳时钟也将继续以恒定速率增加。
- 同步源标识符（SSRC）。SSRC 字段长为 32 比特。它标识了 RTP 流的源。通常在 RTP 会话中的每个流都有一个不同的 SSRC。SSRC 不是发送方的 IP 地址，而是当新的流开始时源随机分配的一个数。两个流被分配相同 SSRC 的概率是很小的。如果发生了，这两个源应当选择一个新的 SSRC 值。

9.4.2　SIP

定义在［RFC 3261；RFC 5411］中的**会话发起协议**（Session Initiation Protocol，SIP）

是一个开放和轻型的协议，其功能如下：

- 提供了在主叫者和被叫者之间经 IP 网络创建呼叫的机制。它允许主叫者通知被叫者它要开始一个呼叫。它允许参与者约定媒体编码，也允许参与者结束呼叫。
- 提供了主叫者确定被叫者的当前 IP 地址的机制。因为用户可能动态地分配到地址（使用 DHCP），而且因为它们可能有多个 IP 设备，每个都有一个不同的 IP 地址，所以用户不具有单一的、固定的 IP 地址。
- 提供了用于呼叫管理的机制，这些机制包括在呼叫期间增加新媒体流、在呼叫期间改变编码、在呼叫期间邀请新的参与者、呼叫转移和呼叫保持等。

1. 向已知 IP 地址建立一个呼叫

为了理解 SIP 的要素，最好看一个具体的例子。在这个例子中，Alice 在使用 PC，并且她要呼叫 Bob，Bob 也在使用 PC 工作。Alice 和 Bob 的 PC 都配有基于 SIP 的软件来产生和接收电话呼叫。在这个初始的例子中，我们将假设 Alice 知道 Bob PC 的 IP 地址。图 9-9 描述了这个 SIP 呼叫建立的过程。

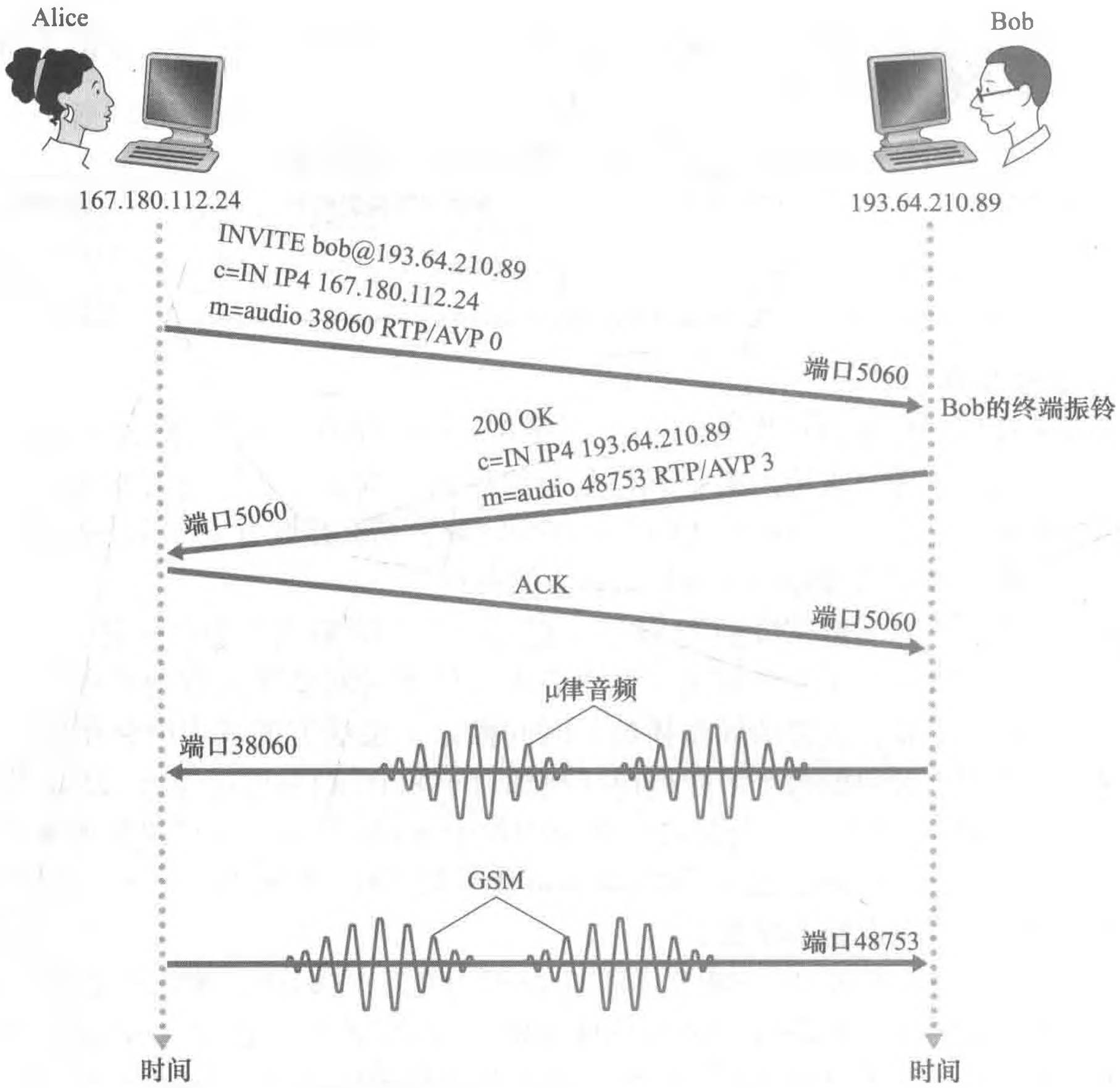

图 9-9　当 Alice 知道 Bob 的 IP 地址时的 SIP 呼叫建立过程

在图 9-9 中，我们看到当 Alice 给 Bob 发送一个 INVITE 报文（这类似于 HTTP 请求报文）时，一个 SIP 会话开始了。该 INVITE 报文通过 UDP 发送给 SIP 的周知端口 5060。（SIP

报文也可以经 TCP 发送。）该 INVITE 报文包括了对 Bob 的标识（bob@ 193.64.210.89）、Alice 当前 IP 地址的指示、Alice 希望接收的音频的指示（该音频以格式 AVP 0 编码，即 μ 律 PCM 编码，并在 RTP 中封装），以及她希望在端口 38060 接收 RTP 分组的指示。在收到了 Alice 的 INVITE 报文之后，Bob 发送一个 SIP 响应报文（该报文类似 HTTP 的响应报文）。该响应 SIP 报文也被发送到 SIP 端口 5060。Bob 的响应包括一个 200 OK 和他的 IP 地址的指示、他希望接收的编码和分组化，以及音频数据应该发送到达的端口号。注意到在这个例子中，Alice 和 Bob 准备使用不同的音频编码机制：要求 Alice 使用 GSM 来对她的音频编码，而要求 Bob 使用 μ 律 PCM 对他的音频编码。在接收到 Bob 的响应后，Alice 向 Bob 发送 SIP 确认报文。在这个 SIP 事务之后，Bob 和 Alice 可以交谈了。（为了看起来清晰，图 9-9 显示 Alice 在 Bob 之后说话，但是事实上他们通常可以同时进行交谈。）Bob 会根据要求对音频进行编码和分组化，并将音频分组发送给 IP 地址 167.180.112.24 的端口号 38060。Alice 也会根据要求对音频进行编码和分组化，并将音频分组发送给 IP 地址 193.64.210.89 的端口号 48753。

从这个简单的例子我们学到了一些 SIP 的关键特性。第一，SIP 是一个带外协议，即发送和接收 SIP 报文使用了一个不同于发送和接收媒体数据的套接字。第二，SIP 报文本身是可读的 ASCII，这与 HTTP 报文类似。第三，SIP 要求所有的报文都要确认，因此它能够在 UDP 或者 TCP 上运行。

在这个例子中，我们考虑一下如果 Bob 没有 μ 律 PCM 编解码器用于音频编码将会发生什么情况。在这种情况下，Bob 不用 200 OK 来响应，而可能用一个 606 Not Acceptable（不可接受）来响应，并在报文中列出他能够使用的所有编解码。然后 Alice 从中选择一个编解码，并发送另一个 INVITE 报文，以此通告了已选择的编解码器。Bob 也能够直接通过发送某个拒绝应答代码来直接拒绝该呼叫。（有很多这种代码，包括“busy（忙）”“gone（离开）”“payment（付费）”和“forbidden（禁止）”。）

2. SIP 地址

在前面的例子中，Bob 的 SIP 地址是 sip：bob@ 193.64.210.89。然而，我们希望许多（即使不是大多数）SIP 地址类似于电子邮件地址。例如，Bob 的地址可以是 sip：bob@ domain.com。当 Alice 的 SIP 设备发送 INVITE 报文，该报文包括这种类似于电子邮件地址的地址；然后 SIP 的基本设施将该报文转发给 Bob 正在使用的 IP 设备（如我们下面要讨论的那样）。其他可能的 SIP 地址形式可以是 Bob 过去的电话号码或者只是 Bob 的名字/中间名/姓氏（假设它是唯一的）。

SIP 地址的一个有趣特点是它们能够被包括在 Web 页面中，就像人们的电子邮件地址用 mailto URL 形式包含在 Web 页面中那样。例如，假设 Bob 有个人主页，并且他要为这个主页的访问者提供一个呼叫他的方法。于是他可能只是在主页中包括该 URL sip：bob@ domain.com。当访问者点击该 URL，访问者设备中的 SIP 应用将启动，并向 Bob 发送 INVITE 报文。

3. SIP 报文

在这个 SIP 简短的介绍中，我们无法包括所有 SIP 报文类型和首部，而只是简要地浏览一下 SIP INVITE 报文，以及少数通用的首部行。我们再假设 Alice 要对 Bob 发起 VoIP 呼叫，此时 Alice 只知道 Bob 的 SIP 地址 bob@ domain.com，并不知道 Bob 正在使用的设备的 IP 地址。那么她的报文可能看起来有些像下面这个：

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
Content-Type: application/sdp
Content-Length: 885

c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

这个 INVITE 行包括 SIP 的版本，这与 HTTP 请求报文一样。任何时候 SIP 报文通过一个 SIP 设备（包括产生该报文的设备）时，它附加上一个 Via 首部来指示该设备的 IP 地址。（我们不久将看到通常 INVITE 报文在到达被叫者的 SIP 应用之前会通过很多 SIP 设备。）与电子邮件报文类似，SIP 报文包括一个 From 首部行和一个 To 首部行。该报文包括一个 Call-ID（呼叫标识符），它唯一地标识该呼叫（类似电子邮件中的报文 ID）；包括一个 Content-Type（内容类型）首部行，定义用于描述包含在 SIP 报文中的内容的格式；还包括 Content-Length（内容长度）首部行，提供报文中内容的字节长度。最后，在一个回车和换行之后，该报文包含内容部分。在这个例子中，内容提供了有关 Alice 的 IP 地址和 Alice 要如何接收该音频的信息。

4. 名字翻译和用户定位

在图 9-9 的例子中，我们假设 Alice 的 SIP 设备知道能够联系到 Bob 的 IP 地址。但是这种假设是很不真实的，不仅因为 IP 地址通常是通过 DHCP 动态分配的，而且 Bob 可能有多个 IP 设备（例如，在家里、工作中和汽车里有不同的设备）。因此现在我们假设 Alice 只知道 Bob 的电子邮件地址 bob@ domain. com，而且对基于 SIP 的呼叫使用同样的地址。在这种情况下，Alice 需要获得用户 bob@ domain. com 正在使用的设备的 IP 地址。为了获得它，Alice 创建一个 INVITE 报文，它以 INVITE bob@ domain. com SIP/2.0 开始，并将该报文发送给一个 **SIP 代理**（SIP proxy）。该代理将以一个 SIP 回答来响应，该回答中也可能包含 bob@ domain. com 正在使用的设备的 IP 地址。该回答也可以选择包括 Bob 的语音信箱的 IP 地址，或者可能包括一个 Web 页面的 URL（上面写着“Bob 在睡觉，请不要打扰!”）。代理返回的结果也可能取决于呼叫者：如果呼叫是 Bob 的妻子发出的，它可能接受该呼叫，并提供 IP 地址；如果呼叫是 Bob 的岳母发出的，它可能用指向“我正在睡觉”的 Web 页面的 URL 来响应。

现在，你可能想知道这个代理服务器是怎样确定 bob@ domain. com 现在的 IP 地址的？为了回答该问题，我们需要讲一下另一个 SIP 设备，即 **SIP 注册器**（SIP registrar）。每个 SIP 用户都有一个相关联的注册器。任何时候用户在设备上发起 SIP 应用时，该应用给注册器发送一个 SIP 注册报文，通知注册器它现在的 IP 地址。例如，当 Bob 在他的 PDA 上发起 SIP 应用时，该应用将发送一个类似于下述内容的报文：

```
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP 193.64.210.89
From: sip:bob@domain.com
To: sip:bob@domain.com
Expires: 3600
```

Bob 的注册器跟踪 Bob 现在的 IP 地址。无论何时 Bob 切换到一个新的 SIP 设备时，该新设备将发送一个新的注册报文，指示该新的 IP 地址。如果 Bob 长时间使用同样的设备，该设备将发送刷新注册报文，指示最近发送的 IP 地址仍然有效。（在上面的例子中，需要

每隔3600秒发送刷新报文来维护在注册器中的地址。）值得注意的是注册器和DNS权威名字服务器类似：DNS服务器将固定的用户名翻译到固定的IP地址；SIP注册器把固定的人识别标志（例如bob@domain.com）翻译为一个动态的IP地址。SIP注册器和SIP代理通常运行在同一台主机上。

现在我们检查一下Alice的SIP代理服务器是如何获得Bob当前的IP地址的。从上面的讨论我们看到，代理服务器只需要转发Alice的INVITE报文给Bob的注册器/代理即可。然后注册器/代理将该报文转发给Bob现在的SIP设备。最后，在已收到了Alice的INVITE报文之后，Bob可以向Alice发送一个SIP应答。

举例来说，考虑图9-10，其中正在217.123.56.89上工作的jim@umass.edu要对正在197.87.54.21上工作的keith@upenn.edu发起IP语音（VoIP）会话。采用下面的步骤：①Jim向umass的SIP代理发送INVITE报文。②该代理在SIP注册器upenn.edu上进行DNS查询（在图中没有显示），并将该报文转发给注册服务器。③因为keith@upenn.edu没有在注册器upenn上注册，upenn注册器发送一个重定向应答，指示它应该试一试keith@nyu.edu。④umass的代理向NYU的SIP注册器发送INVITE报文。⑤NYU注册器知道keith@upenn.edu的IP地址，并将INVITE报文转发给正运行Keith的SIP客户的主机197.87.54.21。⑥~⑧通过注册器/代理把SIP响应返回给217.123.56.89上的SIP客户。⑨媒体直接在两个客户之间发送。（还有一个SIP确认报文，图中没有画出来。）

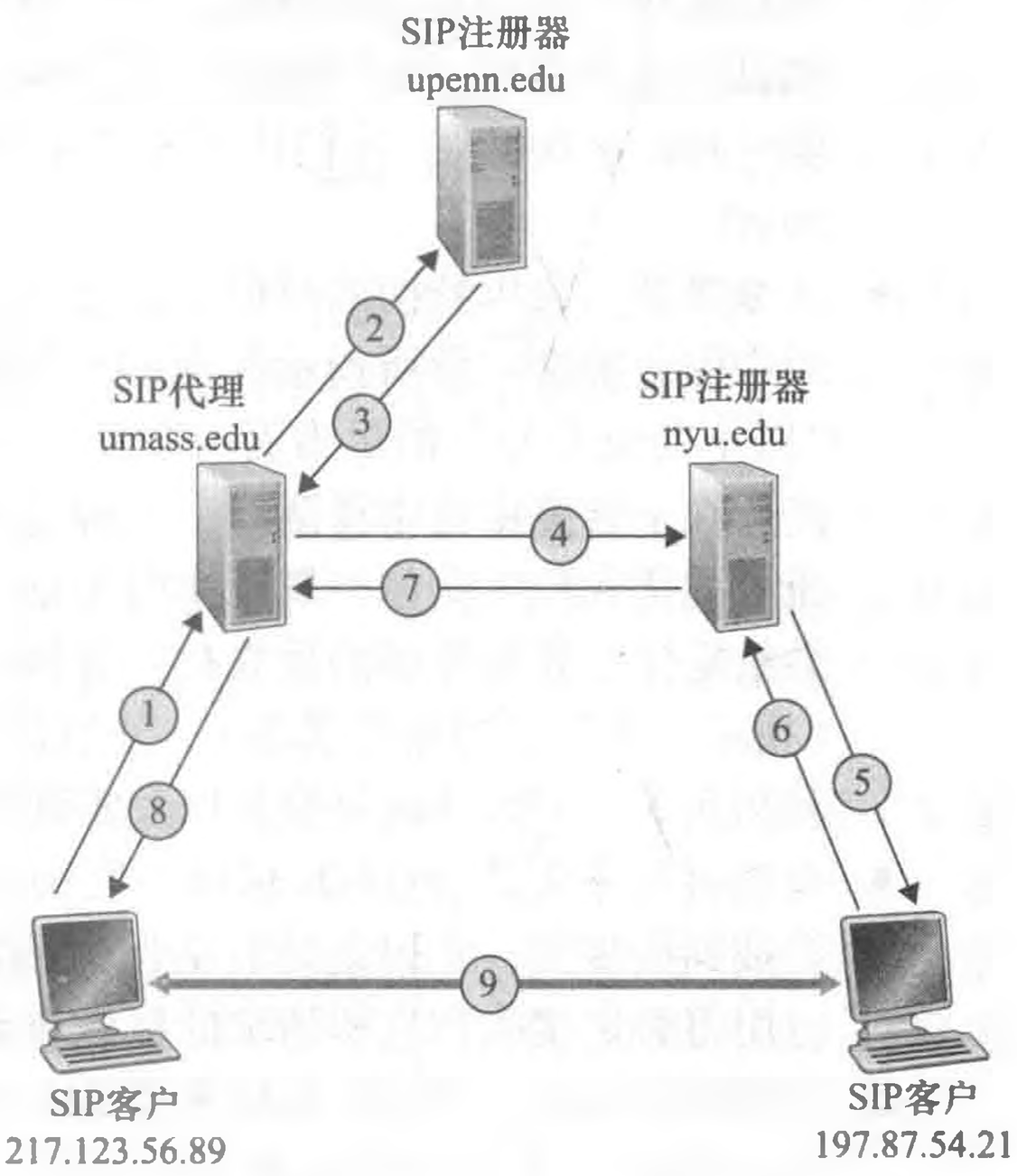

图9-10　涉及SIP代理与注册器的会话发起

我们有关SIP的讨论集中在语音呼叫的呼叫发起方面。SIP通常是一个发起和结束呼叫的信令协议，它能够用于视频会议呼叫和基于文本的会话。事实上，SIP已经成为许多即时讯息应用的基本组件。我们鼓励希望学习更多有关SIP知识的读者访问Henning Schulzrinne的SIP Web站点［Schulzrinne-SIP 2016］。特别是，你在这个站点上会发现SIP客户和服务器的开源软件［SIP Software 2016］。

9.5　支持多媒体的网络

在9.2~9.4节中，我们学习了诸如客户缓存、预取、对可用带宽的适应性媒体质量、适应性播放和丢包缓解技术等应用级机制如何用于多媒体应用，以改善多媒体应用的性能。我们也学习了内容分发网和P2P覆盖网络如何用于提供系统级的交付多媒体内容的方法。这些技术和方法都被设计用于今天的尽力而为因特网。的确，现在因特网得到应用就是因为它只提供单一的、尽力而为类型的服务。但是作为计算机网络的设计者，我们禁不住要问：网络（而不是应用程序或者仅应用级的基础设施）是否可以提供支持多媒体内容交付的机制。如我们很快将看到的那样，答案当然是肯定的！但是我们也将看到，许多这

样的新网络级机制还没有得到广泛部署。这可能是由于它们的复杂性和下列事实所致：应用级基础设施以及尽力而为服务和适当定制的网络资源（例如带宽）的确能够提供“足够好的”（即使不总是尽善尽美的）端到端多媒体交付服务。

表9-4总结了能够对多媒体应用提供网络层支持的三种宽泛的方法。

表9-4　支持多媒体应用的三种网络层方法

方法	粒度	保证	机制	复杂性	当前部署
尽可能利用尽力而为服务	公平处理所有流量	无或者软	应用级支持，CDN，覆盖网络，网络级资源供给	最小	无处不在
区分服务	不同类型的流量处理不同	无或者软	分组标识，监管，调度	中等	某些
每连接服务质量（QoS）保证	每个源到目的地流处理不同	一旦流被准入，软或者硬	分组标识，监管，调度，呼叫准入和信令	高	很少

- *尽可能利用尽力而为服务。*我们在9.2～9.4节学习的应用级机制和基础设施能够成功地用于定制良好的网络——该网络中偶然出现丢包和过大的端到端时延。当预见到需求增加，ISP部署额外的带宽和交换能力以持续确保满意的时延和丢包性能［Huang 2005］。我们将在9.5.1节中进一步讨论**网络定制**（network dimensioning）。
- *区分服务。*自因特网的早期，就已经设想不同类型的流量（例如，在IPv4分组首部中服务类型字段所指示的类型）能够由不同类型的服务所提供，而不是单一的“以不变应万变”的尽力而为服务。使用**区分服务**（differentiated service），当两类流量在一台路由器中排队时，一种类型的流量可以给定严格的优于另一种类型的流量的优先权。例如，属于实时会话式应用的分组由于其严格的时延限制，可能会给定优于其他分组的优先权。在网络中引入区分服务将要求一些用于分组标记（指示一个分组的服务类型）、分组调度和其他方面的新机制。我们将在9.5.2节和9.5.3节中涉及区分服务以及实现这种服务的新网络机制。
- *每连接服务质量（QoS）保证。*使用每连接QoS保证，每个应用的实例显式地预约端到端带宽，并因此具有确保的端到端性能。**硬保证**（hard guarantee）意味着应用将必定接收到它所请求的服务质量。**软保证**（soft guarantee）意味着应用将以高概率接收到它所请求的服务质量。例如，如果某用户要从主机A向主机B进行VoIP呼叫，该用户的VoIP应用在两台主机之间沿着路径在每条链路上显式地预留带宽。但是，允许应用做预约和请求网络同意该预约，这需要一些大的变化。首先，我们需要一个协议来代表应用程序，从发送方到其接收方沿路径预约链路带宽。第二，在路由器队列中将需要新的调度策略，使每连接带宽预约能够兑现。最后，为了进行预约，应用程序必须向网络给出描述来说明它们希望发送进网络的流量，并且网络将需要监管每个应用程序的流量以确保它遵守这个描述。当这些机制结合时，在主机和路由器中要求新的和复杂的软件。因为每连接QoS保证服务尚未见到大规模部署，我们将仅在9.5.4节中简要地涉及这些机制。

9.5.1　定制尽力而为网络

从根本上说，支持多媒体应用的困难是由其严格的性能要求引起的，即低的端到端分