CHAPTER

30 Voice over Internet Protocol

What You Will Learn

In this chapter, you will learn how VoIP is becoming more and more popular as an alternative to the traditional public switched telephone network (PSTN). We'll look at one form of "softphone" that lets users make "voice" calls (voice is really many things) over an Internet connection to their PC.

You will learn about the protocols used in VoIP, especially for the "data" (RTP and RTCP) and for signaling (H.323 and SIP). We'll put it all together and look at a complete architecture for carrying media other than data on the Internet.

In November 2006, when a person in Cardiff, Wales, made a local telephone call, no part of the British Telecom (BT) PSTN was involved. Only the "last mile" of the circuit was the same: No telephone central office, voice switches, or channelized trunks were used to carry the voice call. Instead, the calls were handled by multiservice access nodes (MSANs) and carried with IP protocols over the same type of network that handles BT's Internet traffic.

BT was so happy with the results that by 2011 they say their entire PSTN will be replaced with an IP network using MPLS to both secure and provide QoS for the calls. Many countries use IP voice on their backbones (such as Telecom Italia), but this is the first time a national system has decided to spend a huge amount of money (almost US$20 billion, BT says) to convert everything.

It's old news that many people, both around the world and in the United States, use the Internet to talk over the telephone. Not many of these customers know it, however, because various factors combine to make the use of voice over IP (VoIP) technology a sensitive subject. There are those who intentionally use the Internet for voice calls, and many software packages (such as those from Vonage and Avaya) are available. But not many people know that a percentage of calls (perhaps the majority) made over the PSTN are carried for part of their journey over the Internet using VoIP. The cellular telephone network is converging on IP protocols even faster than the landline network.

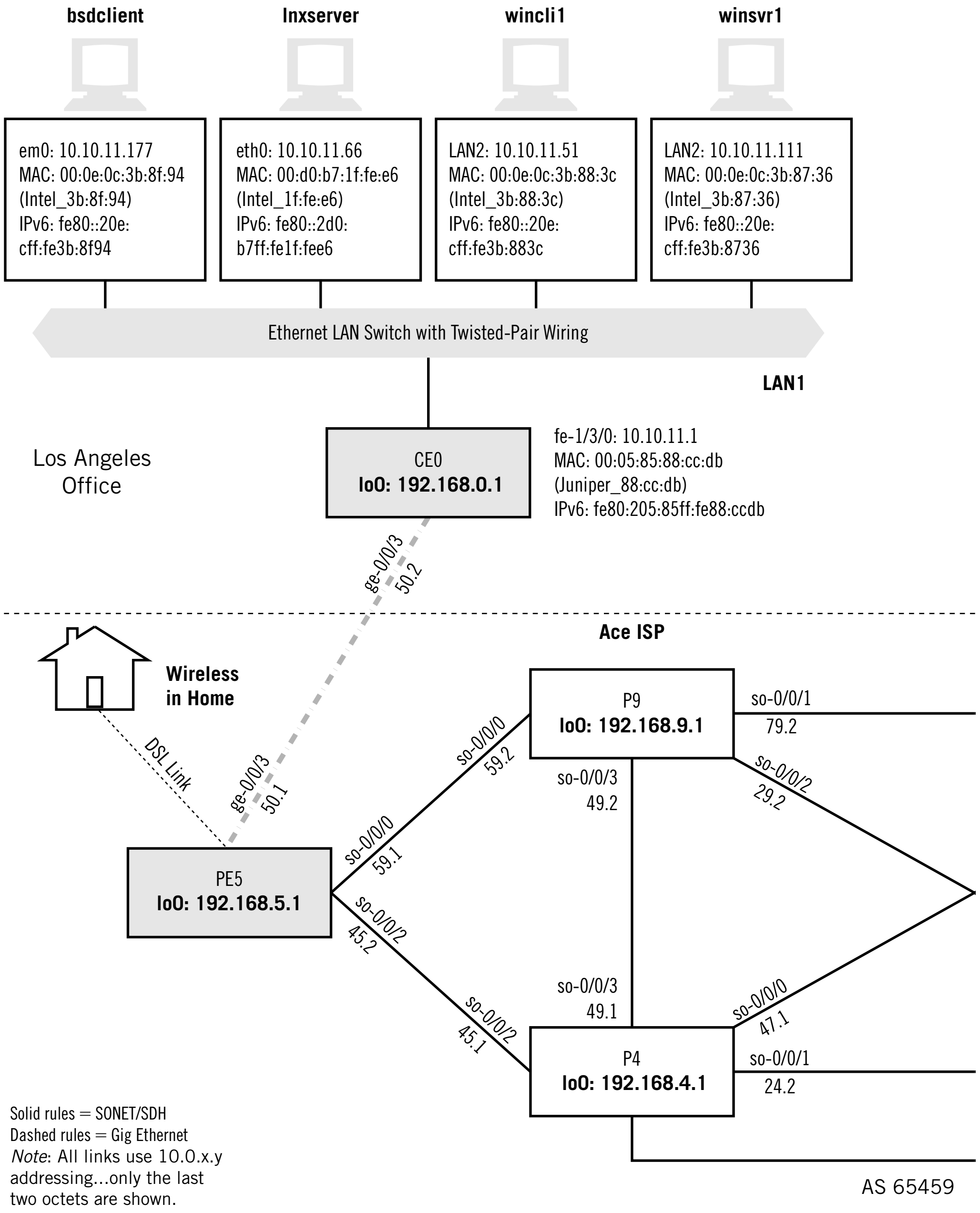
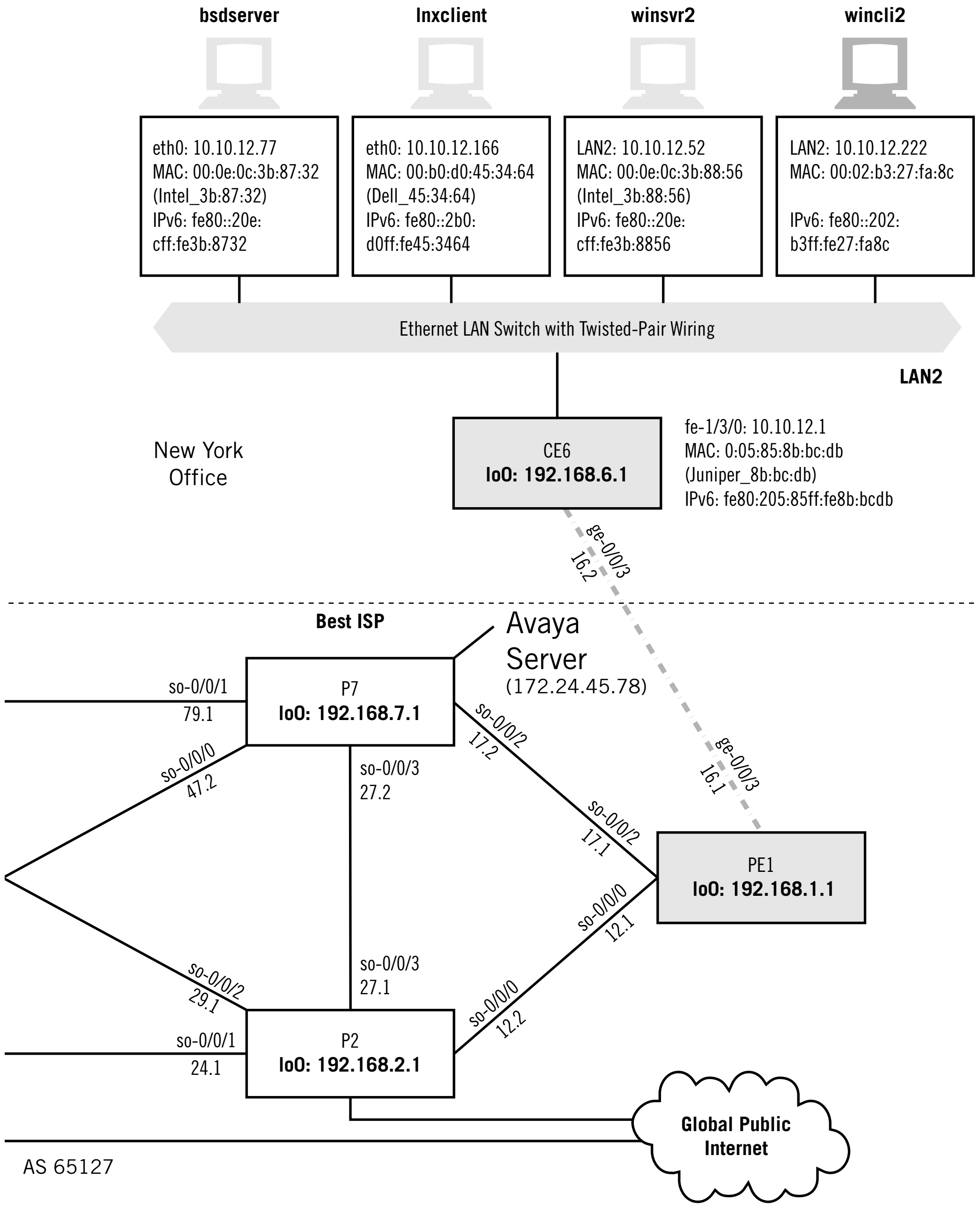


FIGURE 30.1

VoIP setup on the Illustrated Network, showing the host using an Internet telephony package.


bsdserver
eth0: 10.10.12.77
MAC: 00:0e:0c:3b:87:32
(Intel_3b:87:32)
IPv6: fe80::20e:cff:fe3b:8732
lnxclient
eth0: 10.10.12.166
MAC: 00:b0:d0:45:34:64
(Dell_45:34:64)
IPv6: fe80::2b0:d0ff:fe45:3464
winsvr2
LAN2: 10.10.12.52
MAC: 00:0e:0c:3b:88:56
(Intel_3b:88:56)
IPv6: fe80::20e:cff:fe3b:8856
wincli2
LAN2: 10.10.12.222
MAC: 00:02:b3:27:fa:8c
IPv6: fe80::202:b3ff:fe27:fa8c
Ethernet LAN Switch with Twisted-Pair Wiring
LAN2
New York Office
CE6
lo0: 192.168.6.1
fe-1/3/0: 10.10.12.1
MAC: 0:05:85:8b:bc:db
(Juniper_8b:bc:db)
IPv6: fe80:205:85ff:fe8b:bcdb
ge-0/0/3
16.2
Best ISP
Avaya Server
(172.24.45.78)
P7
lo0: 192.168.7.1
so-0/0/1
79.1
so-0/0/0
47.2
so-0/0/3
27.2
so-0/0/2
17.2
so-0/0/2
17.1
ge-0/0/3
16.1
PE1
lo0: 192.168.1.1
so-0/0/0
12.1
so-0/0/0
12.2
so-0/0/3
27.1
so-0/0/2
29.1
so-0/0/1
24.1
P2
lo0: 192.168.2.1
Global Public Internet
AS 65127

The exact percentage of PSTN traffic using VoIP is very difficult to pin down because some telephony carriers are relatively open about this fact and others are not, and all are as wary of their competitors as they ever were. The use of VoIP is also controversial because not too long ago the voice quality of such calls was (might as well admit it) horrible.

This chapter concerns voice, not audio, a distinction often glossed over by users but never by engineers. Voice is concerned primarily with comprehension of the spoken word, that is, of *what* is said rather than how it "sounds." Audio is generally a stereo representation of more than just speech. Think of audio as a motion picture soundtrack. The telephone system is "tuned" to the frequencies used in human speech, not music or special effects explosions. And that makes all the difference.

VoIP IN ACTION

It's a little too much to expect seeing a full-blown VoIP server and gateway on the Illustrated Network, although Juniper Networks does indeed make such software. Nevertheless, we can "borrow" an Avaya IP Softphone server for our network and install the client software on `wincli2` (`10.10.112.222`). Then we can use the VoIP software to place a call to a desk phone and capture the exchange of signaling and voice packets. This is shown in Figure 30.1.

Naturally, the server can place the call anywhere in the world, but having a conversation with a telephone in a local cubicle makes it easier to complete the call, talk, hang up, and so on. Figure 30.2 shows the main screen for the Avaya VoIP software. It doesn't look much like a phone, and some VoIP clients make an effort to make the user

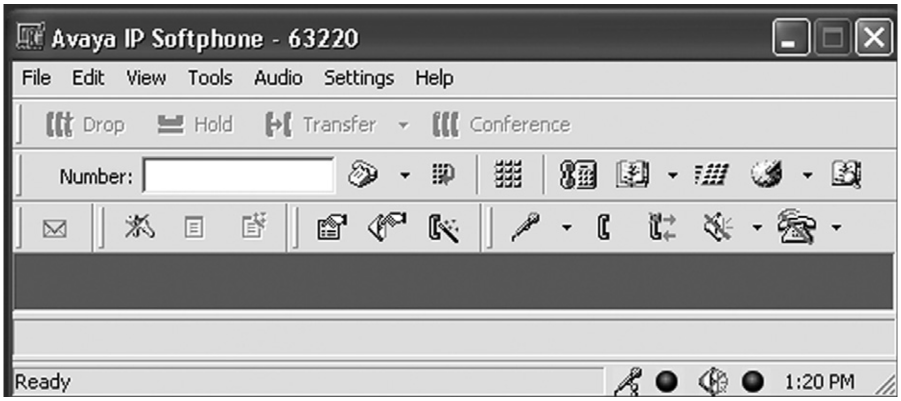


FIGURE 30.2

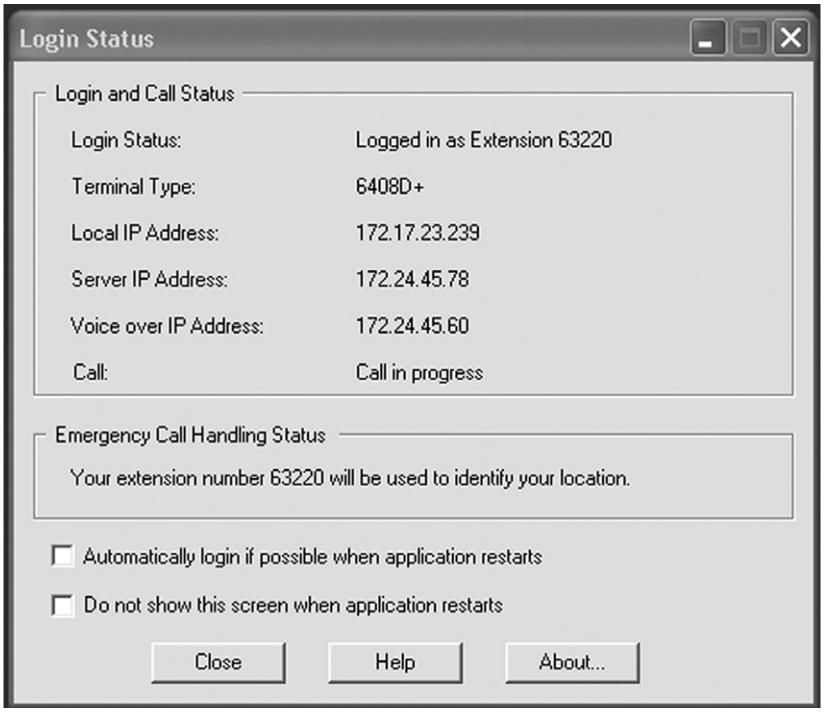
Avaya IP Softphone client interface. Note that this is not very "phone-like."

interface look like a "real" telephone. The best that Avaya does is place a small "keypad" on the screen so that you don't have to type the numbers in.

Before you can make a call, you have to log in to the server. A simple log-in ID and password is used, and then the screen shown in Figure 30.3 appears. It shows the extension the computer is acting as, its IP address (this capture is not from `wincli2`, so the addresses have been changed to the private range), the VoIP server's IP address, and the gateway "VoIP" address. The call status is shown also, and this screen was captured while the call was in progress.

The first thing that becomes obvious when capturing VoIP sessions is the blizzard of packets presented. The actual session, from "dialing" through conversation to "hang-up") lasted less than 30 seconds, and the log-in process, registration, and call setup took only a few seconds of that time. Yet in this 30-second window, some 756 packets passed back and forth from the VoIP client to server.

Most of them were small packets using the Real-Time Protocol (RTP), which carries 20 bytes of voice coded at 8 Kbps (the G.729 standard). A portion of the


Login Status

Login and Call Status

Login Status:	Logged in as Extension 63220
Terminal Type:	6408D+
Local IP Address:	172.17.23.239
Server IP Address:	172.24.45.78
Voice over IP Address:	172.24.45.60
Call:	Call in progress

Emergency Call Handling Status

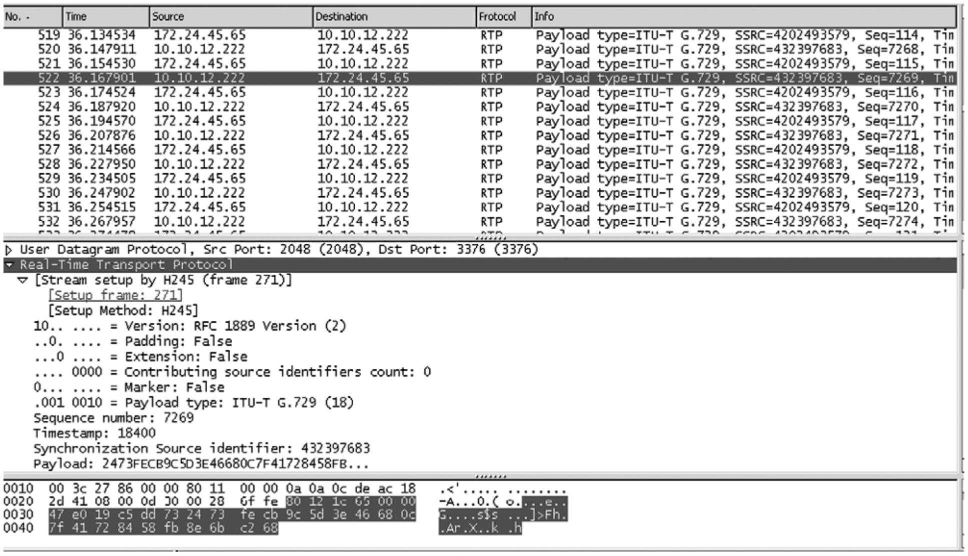
Your extension number 63220 will be used to identify your location.

☐ Automatically login if possible when application restarts

☐ Do not show this screen when application restarts

FIGURE 30.3

Avaya log-on screen with a call in progress.



```
No. .  Time       Source          Destination     Protocol  Info
  519 36.134534  172.24.45.65    10.10.12.222    RTP       Payload type=ITU-T G.729, SSRC=4202493579, Seq=114, Tim
  520 36.147911  10.10.12.222    172.24.45.65    RTP       Payload type=ITU-T G.729, SSRC=432397683, Seq=7268, Tim
  521 36.154530  172.24.45.65    10.10.12.222    RTP       Payload type=ITU-T G.729, SSRC=4202493579, Seq=115, Tim
  522 36.167901  10.10.12.222    172.24.45.65    RTP       Payload type=ITU-T G.729, SSRC=432397683, Seq=7269, Tim
  523 36.174524  172.24.45.65    10.10.12.222    RTP       Payload type=ITU-T G.729, SSRC=4202493579, Seq=116, Tim
  524 36.187920  10.10.12.222    172.24.45.65    RTP       Payload type=ITU-T G.729, SSRC=432397683, Seq=7270, Tim
  525 36.194570  172.24.45.65    10.10.12.222    RTP       Payload type=ITU-T G.729, SSRC=4202493579, Seq=117, Tim
  526 36.207876  10.10.12.222    172.24.45.65    RTP       Payload type=ITU-T G.729, SSRC=432397683, Seq=7271, Tim
  527 36.214566  172.24.45.65    10.10.12.222    RTP       Payload type=ITU-T G.729, SSRC=4202493579, Seq=118, Tim
  528 36.227950  10.10.12.222    172.24.45.65    RTP       Payload type=ITU-T G.729, SSRC=432397683, Seq=7272, Tim
  529 36.234505  172.24.45.65    10.10.12.222    RTP       Payload type=ITU-T G.729, SSRC=4202493579, Seq=119, Tim
  530 36.247902  10.10.12.222    172.24.45.65    RTP       Payload type=ITU-T G.729, SSRC=432397683, Seq=7273, Tim
  531 36.254515  172.24.45.65    10.10.12.222    RTP       Payload type=ITU-T G.729, SSRC=4202493579, Seq=120, Tim
  532 36.267957  10.10.12.222    172.24.45.65    RTP       Payload type=ITU-T G.729, SSRC=432397683, Seq=7274, Tim

▷ User Datagram Protocol, Src Port: 2048 (2048), Dst Port: 3376 (3376)
▾ Real-Time Transport Protocol
  ▽ [Stream setup by H245 (frame 271)]
      [Setup frame: 271]
      [Setup Method: H245]
    10.. .... = Version: RFC 1889 Version (2)
    ..0. .... = Padding: False
    ...0 .... = Extension: False
    .... 0000 = Contributing source identifiers count: 0
    0... .... = Marker: False
    .001 0010 = Payload type: ITU-T G.729 (18)
    Sequence number: 7269
    Timestamp: 18400
    Synchronization Source identifier: 432397683
    Payload: 2473FECB9C5D3E46680C7F41728458FB...

0010  00 3c 27 86 00 00 80 11  00 00 0a 0a 0c de ac 18   .<'..... ........
0020  2d 41 08 00 0d 30 00 28  6f fe 80 12 1c 65 00 00   -A...0.( o....e..
0030  47 e0 19 c5 dd 73 24 73  fe cb 9c 5d 3e 46 68 0c   G....s$s ...]>Fh.
0040  7f 41 72 84 58 fb 8e 6b  c2 68                     .Ar.X..k .h
```

FIGURE 30.4

RTP packets carrying 20 bytes of voice, shown highlighted in the bottom pane.

conversation between client and gateway is shown in Figure 30.4. (The gateway address `172.24.45.65` is now accessed from `wincli2`, and therefore different from that shown in Figure 30.3.)

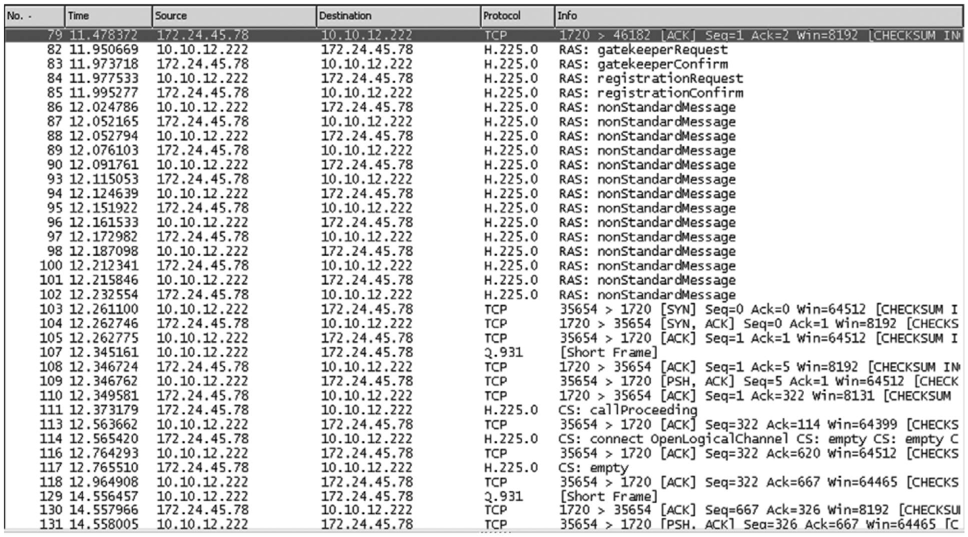
In addition to the TCP packets (which are used to set up the connection to the server), and the RTP packets carrying the voice bits (and the RTCP packets with status information), there are other control packets that serve to remind us that we are not in the data world anymore. The voice world uses a unique language, and an often obscure one at that. This VoIP implementation speaks H.323, a signaling protocol family for voice. The main signaling protocols seen during the call follow.

H.225.0 RAS packets—These are the registration, admission, and status packets used to register the VoIP host on the VoIP server and allow it to use the system to make calls.

H.225.0 CS packets—The call status packets trace the progress of the call. (Is the other phone ringing? Did someone answer?)

Q.931 signaling packets—These are not strictly H.323 signaling packets. Q.931 is the "normal" signaling method with packets used on the PSTN. These are passed from the VoIP client to the server by this VoIP implementation.

Some packets of each type are shown in Figure 30.5, which only shows the expanded upper pane of a full Ethereal capture window. Signaling protocols in VoIP, as opposed to the voice "data" itself, use TCP for its sequencing and resending features.



```
No. -  Time       Source          Destination     Protocol  Info
    79 11.478372  172.24.45.78    10.10.12.222    TCP       1720 > 46182 [ACK] Seq=1 Ack=2 Win=8192 [CHECKSUM IN
    82 11.950669  10.10.12.222    172.24.45.78    H.225.0   RAS: gatekeeperRequest
    83 11.973718  172.24.45.78    10.10.12.222    H.225.0   RAS: gatekeeperConfirm
    84 11.977533  10.10.12.222    172.24.45.78    H.225.0   RAS: registrationRequest
    85 11.995277  172.24.45.78    10.10.12.222    H.225.0   RAS: registrationConfirm
    86 12.024786  10.10.12.222    172.24.45.78    H.225.0   RAS: nonStandardMessage
    87 12.052165  172.24.45.78    10.10.12.222    H.225.0   RAS: nonStandardMessage
    88 12.052794  10.10.12.222    172.24.45.78    H.225.0   RAS: nonStandardMessage
    89 12.076103  172.24.45.78    10.10.12.222    H.225.0   RAS: nonStandardMessage
    90 12.091761  10.10.12.222    172.24.45.78    H.225.0   RAS: nonStandardMessage
    93 12.115053  172.24.45.78    10.10.12.222    H.225.0   RAS: nonStandardMessage
    94 12.124639  10.10.12.222    172.24.45.78    H.225.0   RAS: nonStandardMessage
    95 12.151922  172.24.45.78    10.10.12.222    H.225.0   RAS: nonStandardMessage
    96 12.161533  10.10.12.222    172.24.45.78    H.225.0   RAS: nonStandardMessage
    97 12.172982  172.24.45.78    10.10.12.222    H.225.0   RAS: nonStandardMessage
    98 12.187098  10.10.12.222    172.24.45.78    H.225.0   RAS: nonStandardMessage
   100 12.212341  172.24.45.78    10.10.12.222    H.225.0   RAS: nonStandardMessage
   101 12.215846  10.10.12.222    172.24.45.78    H.225.0   RAS: nonStandardMessage
   102 12.232554  172.24.45.78    10.10.12.222    H.225.0   RAS: nonStandardMessage
   103 12.261100  10.10.12.222    172.24.45.78    TCP       35654 > 1720 [SYN] Seq=0 Ack=0 Win=64512 [CHECKSUM I
   104 12.262746  172.24.45.78    10.10.12.222    TCP       1720 > 35654 [SYN, ACK] Seq=0 Ack=1 Win=8192 [CHECKS
   105 12.262775  10.10.12.222    172.24.45.78    TCP       35654 > 1720 [ACK] Seq=1 Ack=1 Win=64512 [CHECKSUM I
   107 12.345161  10.10.12.222    172.24.45.78    Q.931     [Short Frame]
   108 12.346724  172.24.45.78    10.10.12.222    TCP       1720 > 35654 [ACK] Seq=1 Ack=5 Win=8192 [CHECKSUM IN
   109 12.346762  10.10.12.222    172.24.45.78    TCP       35654 > 1720 [PSH, ACK] Seq=5 Ack=1 Win=64512 [CHECK
   110 12.349581  172.24.45.78    10.10.12.222    TCP       1720 > 35654 [ACK] Seq=1 Ack=322 Win=8131 [CHECKSUM
   111 12.373179  172.24.45.78    10.10.12.222    H.225.0   CS: callProceeding
   113 12.563662  10.10.12.222    172.24.45.78    TCP       35654 > 1720 [ACK] Seq=322 Ack=114 Win=64399 [CHECKS
   114 12.565420  172.24.45.78    10.10.12.222    H.225.0   CS: connect OpenLogicalChannel CS: empty CS: empty C
   116 12.764293  10.10.12.222    172.24.45.78    TCP       35654 > 1720 [ACK] Seq=322 Ack=620 Win=64512 [CHECKS
   117 12.765510  172.24.45.78    10.10.12.222    H.225.0   CS: empty
   118 12.964908  10.10.12.222    172.24.45.78    TCP       35654 > 1720 [ACK] Seq=322 Ack=667 Win=64465 [CHECKS
   129 14.556457  10.10.12.222    172.24.45.78    Q.931     [Short Frame]
   130 14.557966  172.24.45.78    10.10.12.222    TCP       1720 > 35654 [ACK] Seq=667 Ack=326 Win=8192 [CHECKSU
   131 14.558005  10.10.12.222    172.24.45.78    TCP       35654 > 1720 [PSH, ACK] Seq=326 Ack=667 Win=64465 [C
```

FIGURE 30.5

H.225 and Q.931 signaling packets. Note the presence of TCP packets for signaling.

We've done little more than scratch the surface of VoIP, but it is enough to show that VoIP is acceptable and commercially viable today. Let's see why, and explore some of the architectures and protocols in a little more detail.

The Attraction of VoIP

In a very short period of time, we've transitioned from a world where data rode on links optimized for voice by masquerading as sound (that's what a modem is for) to a world where voice rides on links optimized for data (unchannelized) by masquerading as data packets. VoIP is a grand scheme to make this process as easy as possible.

The trick is to have the voice packets preserve the quality-of-service parameters that regulated telephone companies always have to keep an eye on (or their next request for a rate increase might be rejected, and some companies have even been forced to send customers rebates due to poor voice service). In the discussion that follows in this chapter, it will be a good thing to remember that when engineers say "voice" they really mean four things (and no, one of them is *not* audio).

What Is "Voice"?

The PSTN can carry one of four types of "voice" traffic.

1. *Two people talking*—This is what most people think of when they say "voice."
2. *Fax*—Fax machines use low-speed modems to make digital representations of images look like sound. And fax traffic is growing like never before as a result of several social factors (faxes have higher legal standing than email, for one

thing) and the fact that many languages are still not particularly email and keyboard friendly.

3. *Modem data*—Not everyone is on DSL, and a good percentage of users around the world (and, sadly, in the United States) still use analog modems to push perhaps 30 to 50 Kbps back and forth to their ISP.
4. *Touch tone*—Officially, these are the dual-tone multifrequency (DTMF) sounds you hear when you press buttons on a telephone keypad. The familiar beeps are analog (sound) representations of the numbers (digits) pressed.

There are also some economic factors pertinent to VoIP, and VoIP is one reason that premium long-distance telephone calls (which used to cost many dollars per *minute*) are seldom an issue in anyone's budget. (You used to ask before making a long-distance call from someone else's phone, and people rushed out of the shower dripping wet to take a long-distance call because the rates were higher initially.) The use of VoIP as a PSTN bypass method has become less attractive, but the goal of convergence remains strong.

VoIP is also attractive to carriers if what is often called in the United States "toll-quality voice" can be delivered at a reduced bit rate as a stream of TCP/IP packets. Bandwidth savings directly translates into network savings, which is something anyone can understand.

The Problem of Delay

Voice quality is tied to more than just bit rate. Two key parameters in assessing voice quality are *latency* (delay) and *jitter* (delay variation). Voice is much more sensitive to the values of these two network parameters, much more so than the most rigid interactive data requirements. This is because data are usually not processed until the "whole" of something has arrived, and it makes no difference if the first packets that represent a file arrive faster than the last few packets (this is the jitter). And as long as the delay remains below a certain timeout threshold the application will work fine (this is the overall delay).

Delay and latency are often used interchangeably, and they will be here. End-to-end network delays consist of two components: *serial* delay and *nodal processing* delay.

Nodal processing delay is the amount of time it takes for the bits that enter a network node (end node or intermediate node alike) to emerge. End nodes can measure this between application and link, and intermediate nodes as link-to-link delays. Today's routers operate in many cases at "line speeds," but this is a relatively recent development. Early routers operated at much too leisurely a pace to route voice packets at anywhere near the pace required for telephony services (that's what circuit-switched voice switches were for), which basically had to span the globe in about one-quarter of a second. And this had to include the serial delay.

Nodal processing delay also occurs when the analog voice is first digitized. The algorithm used to digitize voice might be complex, adding delay to the entire process. And the more bits needed to be gathered into a packet (bigger packets mean fewer packets than can get lost), the higher the nodal processing delay. This initial delay is often called the *packetization delay*, but it is just another form of nodal delay.

Serial delay is simply an acknowledgment of the fact that bits are sent on a link one by one, so it takes a certain amount of time to send a given number of bits at a given bit rate. If the serial delay is too high for a given application, there are only two ways to lower it: Put fewer bits in a packet or raise the link bit rate. Of course, you can do both. You can put fewer bits in voice packet by lowering the bit rate of the voice inside (or sending more packets—it's a tradeoff).

Jitter is the variation of the end-to-end delay across the network. As the delay varies, bits arrive either early or late at the destination. If they arrive too quickly, bits might overflow a buffer. If they arrive too late, silence results. Gaps in the conversation occur either way. And even less extreme jitter can distort the analog voice that results from the bits. To smooth out arriving voice, a "jitter buffer" is used to add the delay necessary to make the voice sound like it all arrives with the same delay.

The delay issues in VoIP are shown in Figure 30.6. Naturally, the same process works in the other direction.

Just like overall delay, and apart from jitter buffers, jitter can be handled in a couple of ways. Delay variations usually result from nodal processing load variations and buffer queue depth. In other words, when the node is busy, things slow down. This effect can be minimized by splitting off the voice for special handling, getting faster network nodes, or by increasing link bandwidth. (Note that constant appearance of "increased

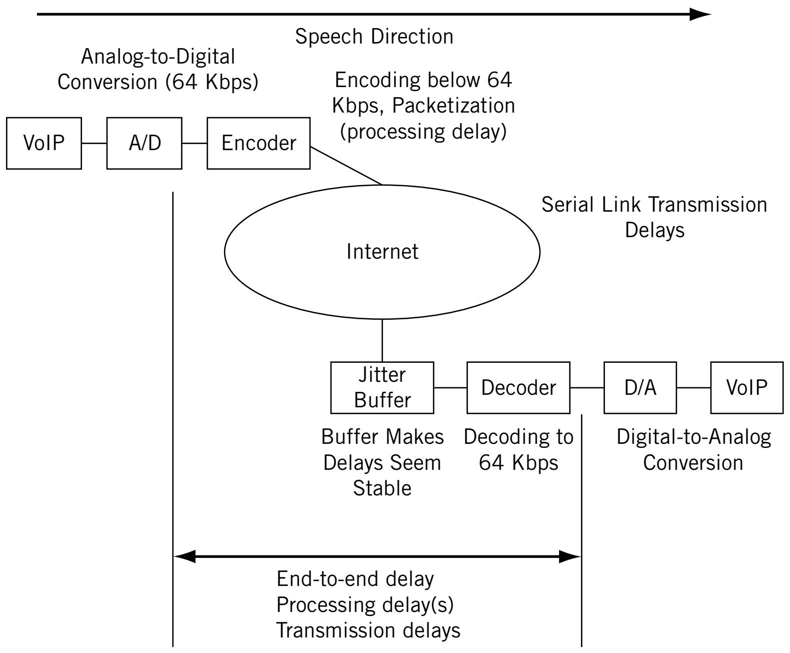

Speech Direction
Analog-to-Digital Conversion (64 Kbps)
Encoding below 64 Kbps, Packetization (processing delay)
VoIP
A/D
Encoder
Internet
Serial Link Transmission Delays
Jitter Buffer
Decoder
D/A
VoIP
Buffer Makes Delays Seem Stable
Decoding to 64 Kbps
Digital-to-Analog Conversion
End-to-end delay
Processing delay(s)
Transmission delays

FIGURE 30.6

VoIP processing and transmission delays. Note that the jitter buffer compensates for differences in delays during different parts of the call.

link bandwidth" as a solution to networking problems, a fact that has slowed development of alternative solutions to many issues.)

The key to VoIP is not so much digitizing voice at a low bit rate, but rather TCP/IP and the Internet carrying packetized voice with acceptable latency and jitter as perceived by the humans using it. (Related issues, such as replacing silence with "comfort noise" and detecting "voice activation," are beyond the scope of this chapter.)

Packetized Voice

Voice on the PSTN is usually a streaming bidirectional connection at a fixed 64 Kbps. Once digitized, there was little incentive to play around with voice too much because any reduction in bit rate was offset by a loss in voice quality. Regulated carriers had to maintain certain voice quality levels or risk customers not having to pay for the call. However, if the "slope" of the decline of voice could be leveled so that quality at 16 Kbps or even 8 Kbps was not that much different than at 64 kbps, more calls could be carried over the same facilities. Not only that, but any bandwidth not used for carrying voice calls could be used for data (packets).

However, low-bit-rate voice with acceptable quality—something achieved with modern digital signal processing (DSP) chips—is not the same as packetized voice. Using "spare" voice bandwidth for data was the idea behind ISDN and eventually DSL. But the voice stayed on the voice channel and the data stayed on the data channel. Only by truly packetizing voice can voice and data be combined in an efficient manner.

A "voice" service really consists of two major components: content—which can take on four different meanings (as we have seen)—and signaling. This signaling is not the same as touch tones, although the intent is similar. This signaling is already packetized, and is how the number you dial and other information (such as the number you dialed from) makes its way through the voice signaling network.

This signaling network is as packetized as TCP/IP, uses special network nodes (which still route), and is known as Signaling System 7 (SS7). The real issue in VoIP is not so much how to packetize the voice content (gather bits and stick a header on them and send them out) but how the SS7 signaling packets relate to the Internet and TCP/IP.

The main stumbling block to universal VoIP service today is not so much that there are many ways to packetize voice content (there are options in many other TCP/IP protocols) but that there are many ways (and many *architectures*) to carry voice signaling information in a TCP/IP environment. These VoIP protocol controversies are important enough for a detailed look.

PROTOCOLS FOR VoIP

Voice, like audio and video, is a "real-time" application. And, as in multicast TCP is a poor choice for voice connections over the Internet. This sounds odd because voice is as connection oriented as TCP and requires handshaking overhead to complete a "call." (Humans handshake with a ring and a vocalized shared "Hello.")

The problem is not just TCP overhead, it's the fact that TCP will *always* resend missing data units. That's what it's for. However, the meaningful resending of voice bits is impossible in VoIP given the real-time nature of voice. So, UDP (which blithely accepts lost data units with a shrug) is used in VoIP—just as in multicast.

But TCP headers contain a number of fields that are very helpful for end-to-end communications, which are fields lost in UDP, such as a sequence number to detect lost voice packets. So we'll have to take what fields we need from TCP and stick them inside (after) the UDP header. This new header will have to have a name and a place in the TCP/IP protocol stack. We'll call it the Real-Time Protocol (RTP) and use it for the transport of digitized voice inside our IP packets.

Signaling, however, is another matter. We might want to keep TCP for that because resending lost signaling packets is actually a good idea (calls that are not completed do not generate revenue for metered service or friends in the user community). In addition, the delays for signaling in regulated voice services are much less stringent than the delays for voice packets, which make TCP connection overhead tolerable. So, in some cases (especially over a WAN), TCP is acceptable for voice signaling.

But what form should TCP/IP voice signaling packets take? How should voice-capable TCP/IP devices find each other by IP address? How are VoIP calls handed off to (or received from) the PSTN network with SS7? Where are the voice gateways? Who runs the gateways—the customer or the service provider? In other words, what is the overall architecture of the TCP/IP voice-signaling network?

Unfortunately, we live in a world where there are competing answers to all of these signaling questions. Let's start by looking at RTP and then examining the major differences between the various systems of VoIP signaling.

RTP for VoIP Transport

RTP grew out of efforts to improve the Streams 2 (ST2) protocol defined in RFC 1819. ST2 was known as IPv5 and is why IPv4 evolved into IPv6. RTP was defined in RFC 1889 and deliberately left open-ended to allow room for the protocol to evolve.

RTP is really a framework using *application layer framing* and was initially aimed at audio (and video) multicast sessions. However, two-way phone calls are just special cases of audio multicast, so RTP is a good fit for VoIP.

RTP can replace TCP for many applications, but in VoIP it is used with UDP. The RTP architecture also includes another protocol, the Real-Time Control Protocol (RTCP), which uses IP directly to monitor the job RTP is doing in terms of delay and voice quality.

IP port numbers 5004 and 5005 are used for RTP and RTCP, respectively, and the ports are the same on both ends of the connection. The overall RTP architecture is shown in Figure 30.7.

There are many audio and video codecs supported by RTP, but not all of them are needed for VoIP (especially video codecs, naturally). In addition, the RTP architecture establishes devices called *mixers* (to mix multiple sources for conferences) and *translators* (to compensate for low and high bit-rate links and LANs). These functions can be implemented in some type of "voice and audio server" on a LAN, but are not used in VoIP.

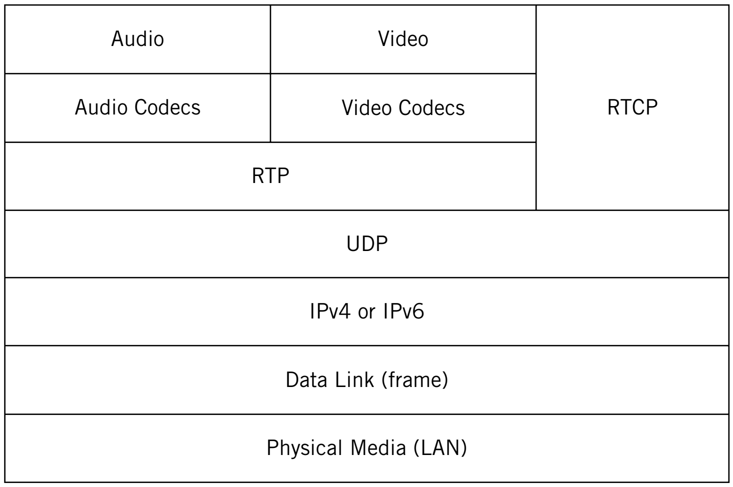


FIGURE 30.7

RTP and RTCP protocol stack, showing how these protocols use UDP instead of TCP.

The structure of the basic RTP header is shown in Figure 30.8. Only the fields that apply to two-party calls (point to point) are fully described.

V (version)—This 2-bit field gives the current version of RTP.

Pad (padding)—This 2-bit field aligns the packet to a specific boundary. The actual padding byte count is given in the last byte of the RTP data.

E (extension)—This 1-bit field extends the length of the RTP header, mostly for experimental purposes, and is almost always set to zero.

M (marker)—This 1-bit field is used in the first packet sent after a period of silence.

Payload type—This 7-bit field is used to define 128 types of RTP payloads. Some are static, and can only be used for the defined type, but newer ones are dynamic and are assigned by the control protocol (such as SIP).

Sequence number—This 16-bit field increases by one for each RTP packet sent. Receivers can use this field to detect missing or out-of-sequence packets.

Timestamp—This 32-bit field is most useful for video (all bits from the same frame have the same timestamp), but it is used for the voice sampling rate as well.

The count field gives the number of "contributors" to a conference. For multiparty calls, the synchronization source identifier (SSRC) and a series of contributing source identifiers (CSRC) matching the count are not used. The VoIP RTP header adds 8 bytes to the voice stream. The format of the payload in the RTP data field is determined by the values in the categories listed in Table 30.1.

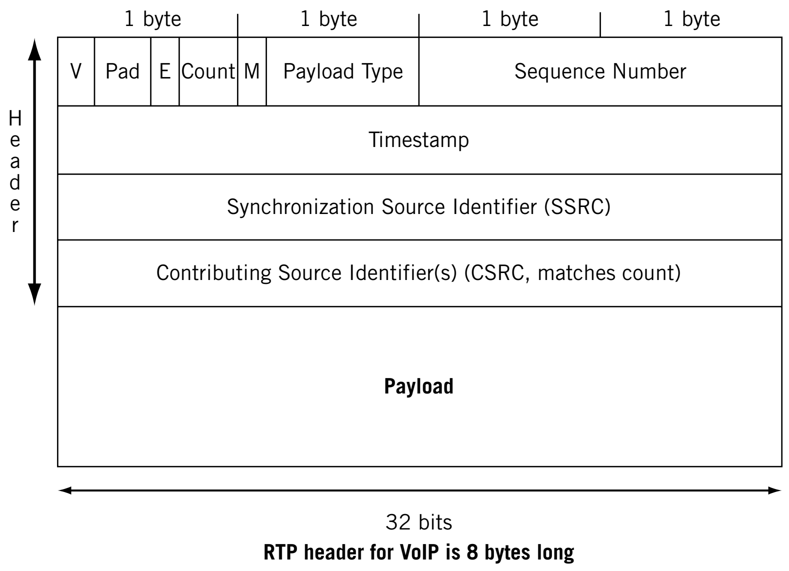

1 byte
1 byte
1 byte
1 byte
V
Pad
E
Count
M
Payload Type
Sequence Number
Timestamp
Synchronization Source Identifier (SSRC)
Contributing Source Identifier(s) (CSRC, matches count)
Payload
Header
32 bits
RTP header for VoIP is 8 bytes long

FIGURE 30.8

RTP header fields, which preserve some aspects of TCP fields.

Table 30.1 RTP Payload Formats and Their Meanings

Type	Meaning
0–34	Static assignment (most popular bit rates and formats here)
35–71	Unassigned
72–76	Reserved
77–95	Unassigned
96–127	Dynamic assignment (under the control of a call control protocol)

RTP is a pure transport mechanism. Feedback on quality and immediate network conditions is provided by the receiver to the sender with RTCP. RTCP doesn't say what senders should *do* with this information, such as the revelation that a router is becoming overloaded and dropping more packets than it is sending, but at least the ability to detect problems is there.

RTP generates periodic "reports" about the RTP session. There are five RTCP message types.

1. *Sender report*—Contains transmission and reception statistics from conference participants that are active senders.

2. *Receiver report*—Reception statistics from conference participants that are *not* active senders.
3. *Source description*—Items relating to the source, including the canonical DNS name.
4. *Bye*—Used to end a session.
5. *Application specific*—Contains any information that the applications agree to share.

The possible payload formats that can be used to carry voice bits following the RTP header are complex, seemingly fiendishly so. These are defined in RFC 2833. Fortunately, they are usually of interest only to telephony engineers.

Signaling

I first encountered voice over IP around the same time I encountered the Web, in the early 1990s. It was in a university setting, where the absolute utility and cost effectiveness of things are not as rigid as in the business world. In the fluid environment of an educational institution, many things happen because they are instructive, groundbreaking, and just, well, cool.

A graduate student of mine was in the lab one day, busily chattering into a microphone hooked up to a PC and intently listening to the garbled voice coming out of the PC's speakers. Much of the conversation consisted of "What?" and "Huh?"

When I asked, he informed me that he was talking over the Internet to an old friend in a similar lab at RPI in Troy, New York, about 150 miles north of us—and in those days usually an expensive long-distance call away (especially for graduate students). I asked him how the friend in Troy knew to be in the lab at the right time to answer his PC. "Oh," my student said, "I called his dorm room from your office and told him to go there."

Things have come a long way since the early 1990s. The trouble back then was that the world of Internet telephony was a closed world, limited to Internet-attached devices. There were no signaling gateways to translate phone numbers to IP addresses and back, and so no way to enable calls with one end on the Internet and the other end in the PSTN to complete calls.

This is not to say that there were not VoIP gateways. There were. But these used proprietary protocols for the most part, and only connected to their cousin devices from the same vendor. So, there was a need to create standard signaling protocols for VoIP.

Today, the issue seems to be *not* a lack of proposed standard protocols for VoIP but their proliferation. There are three general protocol stacks that can be used for VoIP. These are shown in Figure 30.9.

Note that the third stack combines two methods known as the Multimedia Gateway Control Protocol (MGCP) and Megaco/H.248 into a single stack. The two are similar enough to allow this.

However, things are not as bad as they might seem at first. All three of the signaling protocols could have a role in the "converged" VoIP architecture of Internet and PSTN. Before we see how this is possible, let's take a look at each of the protocols in turn.

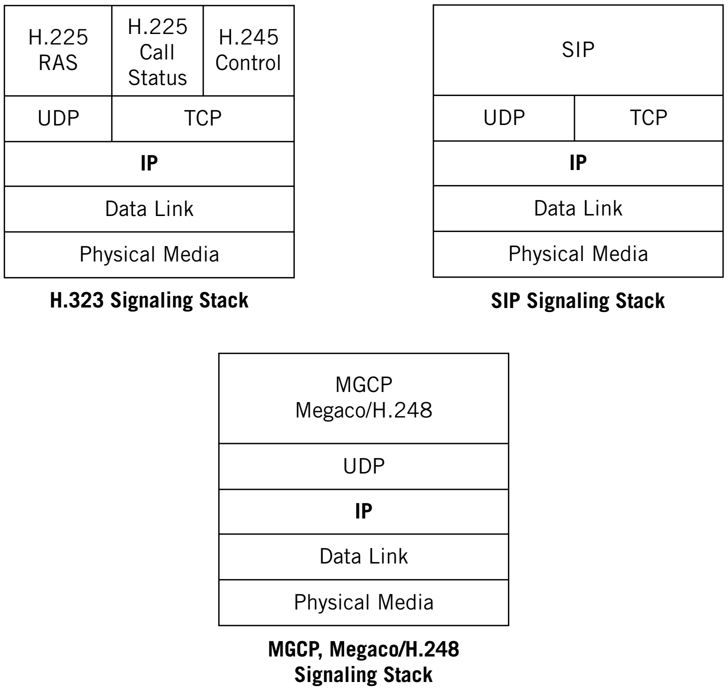


FIGURE 30.9

Three VoIP signaling architectures.

H.323, the International Standard

The H.323 signaling protocol framework is the international telephony standard for all telephony signaling over the packet network (not just the Internet). When work on H.323 began, the packet network most commonly mentioned for H.323 was X.25, then ATM, and not the Internet. In a sense, H.323 doesn't care—it's just an umbrella term for what needs to be done.

Like RTP, H.323 was designed for audio and video conferencing, not just point-to-point voice conversations. A LAN with devices that support H.323 capabilities (H.323 terminals, which have many different subtypes) also has an H.323 multipoint control unit (MCU) for conference coordination. The LAN includes an H.323 *gateway* to send bits to other H.323 *zones* and an H.323 *gatekeeper*. The gatekeeper is optional, and is needed only if the terminals are so underpowered they cannot generate or understand H.323 messages on their own. (Most can, although H.323 is not trivial.) The H.323 gateway is essentially a router, but with the ability to support packetized voice to PSTN connections (and the terminals are computers, of course).

The main H.323 signaling protocols used with VoIP are H.225 RAS (Registration, Admission, and Status), which is used to register the VoIP device with the gatekeeper, and H.255 CS (call status), which is used to track the progress of the call. The structure

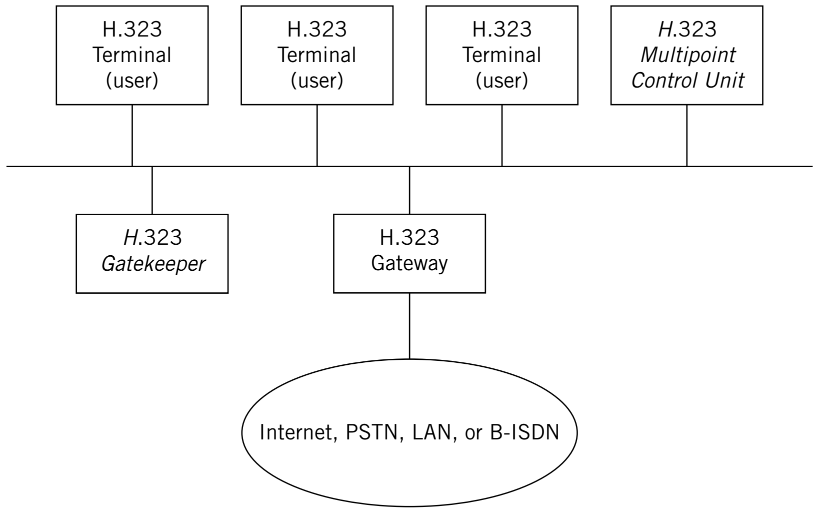


FIGURE 30.10

H.323 zone components. (Optional components are shown in italic.)

of a typical H.323 zone is shown in Figure 30.10. H.323 signaling uses both UDP and TCP when run on an IP network, and uses RTP and RTCP for transport. Components that are not strictly needed for VoIP are shown in italics.

H.323 supports not only audio and video conferencing but also *data* conferencing, where users can all see the same information on their PCs and changed data are updated across the network. Cursors are usually distinguished by distinctive colors.

The trouble with H.323 was that it is complete overkill for VoIP. Data and video support are not needed for VoIP, and some wondered why H.323 was needed in VoIP at all given its telephony roots and the hefty amount of power needed to run it. Maybe the Internet people could come up with something better.

SIP, the Internet Standard

The Session Initiation Protocol (SIP), defined in RFC 3261, is the official Internet signaling protocol for IP networks. Each session can also include audio and video conferencing, but right now SIP is mainly used for simple voice over the Internet. SIP is a text-based protocol similar to HTTP and SMTP, uses multicast Session Description Protocol (SDP) for the characteristics of the media, and is technically independent of any particular packet protocol.

Both H.323 and SIP define mechanisms for the formal processes of call signaling, call routing (the path the voice bits will follow), capabilities exchange (the bit rate that should be used), and supplementary services (such as collect calling). However, SIP attempts to perform these functions in a more streamlined fashion than H.323.

VoIP combines the worlds of the telephony carriers (H.323) and the Internet (SIP). Not surprisingly, both telephony carriers and Internet people see their way as the best way for a unified signaling protocol suitable for both environments.

The SIP architecture is client–server in nature, as expected, but with adaptation for the peer-to-peer nature of telephony. The main SIP components are the user agent (the "endpoint" device), the "intermediate servers" (which can be proxy servers or redirect servers), and the registrar.

Proxy servers forward SIP requests from the user agent to the next SIP server or user agent and retain accounting and billing information. User agents can be clients (UACs) when they send SIP requests, and servers (UASs) when they receive them. SIP redirect servers respond to client requests and tell the UACs the requested server's address.

The SIP registrar stores information about user agents, such as their location. This information is not maintained or accessed by SIP, but by a separate "location service" that is still part of the SIP framework. SIP is flexible enough to support stateless requests or to remember them, and is not tied to any one directory method to locate SIP users and components.

The general SIP architecture is shown in Figure 30.11. The only piece that is missing is the registrar, which takes the SIP register request information and uses it to update the information stored in the location server. The figure shows the sequence of SIP requests and responses to establish a session (call). The details of each step are beyond the scope of this chapter, but the point is that a lot of messages are required to complete the call. Once the called party is found and alerted in Step 8, however, the call is quickly completed from proxy to proxy and back to the calling party.

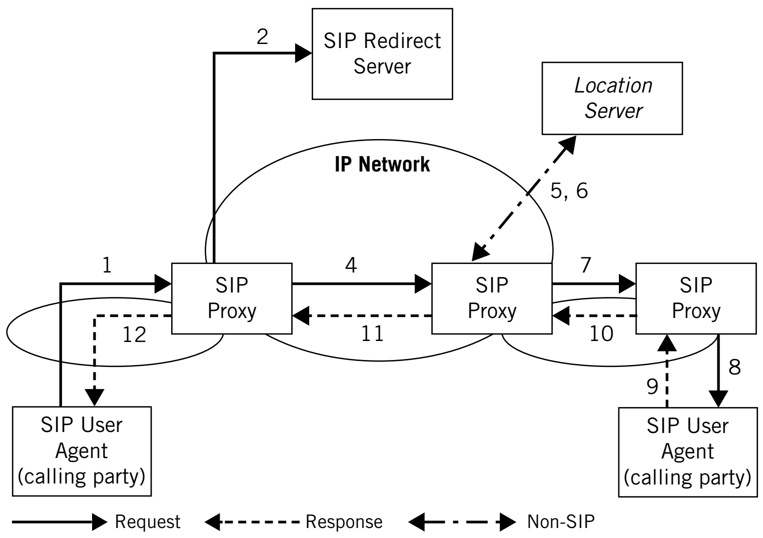


FIGURE 30.11

SIP session initiation steps.

There are six basic types of SIP requests.

1. *Invite*—Start a session.
2. *ACK*—Confirms that the client has received a final response to an invitation.
3. *Options*—Provides capabilities information, such as voice bit rates supported.
4. *BYE*—Release a call.
5. *Cancel*—Cancel a pending request.
6. *Register*—Sends information about a user's location to the SIP registrar server.

SIP responses follow the familiar three-digit codes used in many other TCP/IP protocols. The major response categories in SIP follow:

- 1xx Provisional, used for searching, ringing, queuing, and so on
- 2xx Success
- 3xx Redirection, forwarding
- 4xx Server failure
- 5xx Global failure

SIP even allows PSTN signaling messages (packets) to use the Internet to set up calls that use the PSTN on both ends, so telephony carriers can send calls directly over the Internet. This version of SIP is called SIP-T (SIP for Telephony).

MGCP and Megaco/H.248

It's one thing to describe a network of media gateways leading to the PSTN (as in H.323), or a series of servers that relay call setup packets across the Internet, as in SIP. But these elements do not function independently, despite the fact that H323 Media gateways and SIP proxy servers are on the customer premises and on LANs. If VoIP must handle the most general situations with endpoints anywhere on the Internet or PSTN, some type of overall control protocol must be developed.

That's what the Media Gateway Control Protocol (MGCP) is for. Despite the H.323 terminology, MGCP was defined in RFC 2705 as a way to control VoIP gateways from "external call control elements." In other words, MGCP allows the service providers (telephony carriers or ISPs) to control the VoIP aspects of the customer's network, whether it uses H.323 or SIP. These control points are known as call agents, and MGCP only defines how a call agent talks to the media gateway—not how the call agents talk to each other. Call agent communication uses H.323 or SIP, so this is not a limitation.

The terminology for all of these signaling protocols is starting to get confusing. Let's back up and see what we've got so far.

Media gateways—The H.323 component that handles all voice bits sent to and from the "zone" (usually a LAN).

Proxy servers—The SIP components that handle requests for SIP-capable user agents on the LAN.

Call agents—The MGCP components that control the media gateways and can do so over the Internet link itself.

But wait, didn't SIP have a media gateway? No, SIP defines a signaling framework that can tell you where the gateway *is*, but doesn't include that device in its framework. If you think about it, it all makes sense and all of the pieces are needed to make VoIP as useful as possible.

The biggest clash is between parts of H.323 and SIP. You don't need to have *both* running on the "terminals" or "user agents," no matter which terminology you use. However, many vendors are hedging their bets and supporting both H.323 and SIP right now. The funny thing is that they usually don't support MGCP.

How's that? Well, MGCP was modified into something called *Megaco* to make it more palatable to the telephone carriers. Megaco was standardized as H.248, so the result often appears as Magaco/H.248. The architecture of Megaco/H.248 is very similar to that of MGCP.

PUTTING IT ALL TOGETHER

How do H.323, SIP, and Megaco/H.248 relate to one another today? Well, they all have a place in a VoIP network that can place or take calls to and from the PSTN and handle IP transport of what appear to customers to be PSTN calls. Figure 30.12 shows the overall architecture of such a converged VoIP network.

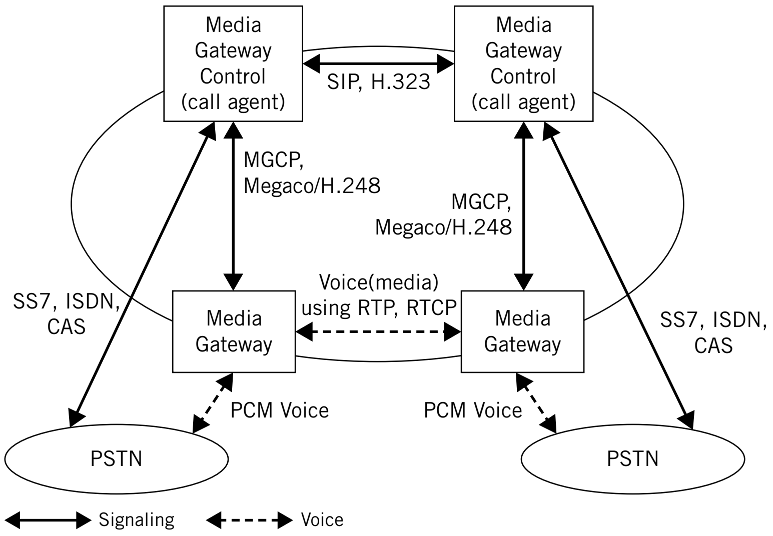


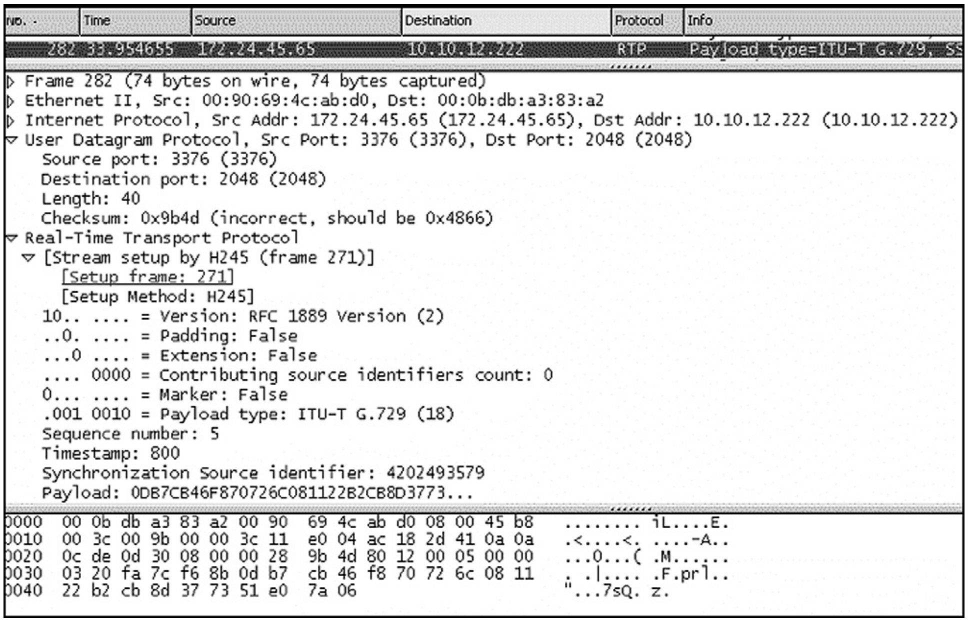
FIGURE 30.12

VoIP converged network architecture, showing how VoIP protocols can work together.

We’ve seen ISDN and SS7 signaling before, and channel-associated signaling (CAS) is used on aggregate circuits with many voice channels. Pulse code modulation (PCM) is a common way to carry the voice bits on the PSTN. Therefore, the “upper” path through the figure describes the signaling, and the “lower” path shows the “media” channel using RTP and RTCP over the Internet (or private IP network).

QUESTIONS FOR READERS

Figure 30.13 shows some of the concepts discussed in this chapter and can be used to answer the following questions.



```
No. .    Time        Source            Destination       Protocol  Info
   282 33.954655   172.24.45.65      10.10.12.222      RTP       Payload type=ITU-T G.729, SS
▷ Frame 282 (74 bytes on wire, 74 bytes captured)
▷ Ethernet II, Src: 00:90:69:4c:ab:d0, Dst: 00:0b:db:a3:83:a2
▷ Internet Protocol, Src Addr: 172.24.45.65 (172.24.45.65), Dst Addr: 10.10.12.222 (10.10.12.222)
▽ User Datagram Protocol, Src Port: 3376 (3376), Dst Port: 2048 (2048)
    Source port: 3376 (3376)
    Destination port: 2048 (2048)
    Length: 40
    Checksum: 0x9b4d (incorrect, should be 0x4866)
▽ Real-Time Transport Protocol
  ▽ [Stream setup by H245 (frame 271)]
      [Setup frame: 271]
      [Setup Method: H245]
    10.. .... = Version: RFC 1889 Version (2)
    ..0. .... = Padding: False
    ...0 .... = Extension: False
    .... 0000 = Contributing source identifiers count: 0
    0... .... = Marker: False
    .001 0010 = Payload type: ITU-T G.729 (18)
    Sequence number: 5
    Timestamp: 800
    Synchronization Source identifier: 4202493579
    Payload: 0DB7CB46F870726C081122B2CB8D3773...
0000  00 0b db a3 83 a2 00 90  69 4c ab d0 08 00 45 b8   ........ iL....E.
0010  00 3c 00 9b 00 00 3c 11  e0 04 ac 18 2d 41 0a 0a   .<....<. ....-A..
0020  0c de 0d 30 08 00 00 28  9b 4d 80 12 00 05 00 00   ...0...( .M......
0030  03 20 fa 7c f6 8b 0d b7  cb 46 f8 70 72 6c 08 11   . .|.... .F.prl..
0040  22 b2 cb 8d 37 73 51 e0  7a 06                     "...7sQ. z.
```

FIGURE 30.13

Frame 282 using RTP captured from a VoIP call.

1. What are the four types of "voice" carried by VoIP?
2. In the figure, is `wincli2` sending (talking) or receiving (listening)?
3. Which UDP port is the client using for the call?
4. Which international standard protocol is used to set up the stream?
5. Which voice coding standard is used for the "data" in the voice packet?