# The IETF Internet Telephony Architecture and Protocols

Henning Schulzrinne
Dept. of Computer Science
Columbia University
hgs@cs.columbia.edu

Jonathan Rosenberg
Bell Laboratories
Lucent Technologies
jdrosen@bell-labs.com

March 18, 1999

**Abstract**

Internet telephony was first used as a simple way to provide point-to-point voice transport between two IP hosts. However, the growing interest in providing integrated voice, data, and video services has caused its scope to be expanded. Internet telephony now encompasses a range of services. These services include not only traditional conferencing, call control, multimedia, and mobility services, but also new ones that integrate web, email, presence, and instant messaging applications with telephony. Internet telephony and traditional circuit switched telephony will co-exist for quite some time, requiring interworking between the two. In this paper, we present a suite of protocols, developed in the Internet Engineering Task Force (IETF), which provide a partial solution to this complex problem.

## 1 Introduction

Internet telephony was first used as a simple way to provide point-to-point voice transport between two IP hosts, primarily to replace expensive international phone calls. However, the growing interest in providing integrated voice, data, and video services has caused its scope to be expanded. Internet telephony now encompasses a range of services. These services include not only traditional conferencing, call control supplementary services, multimedia transport, and mobility, but also new services that integrate web, email, presence, and instant messaging applications with telephony. Furthermore, it is generally accepted that Internet telephony and traditional circuit-switched telephony will co-exist for quite some time, requiring gateways between the two worlds.

Consider the following example of what an integrated IP telephony call might be like. John is sitting at his computer, and all of a sudden his machine sounds a “boing”, followed by speaking “Audio and video call from Joe”. He accepts the call, and talks for a while. He then decides that Alice needs to be in on the call, too. He says “Add Alice”, and the speech recognition software on his PC interprets the command. His client application consults a local Internet “white pages” directory and adds ’‘alice@ieee.org” to the call. The call setup request reaches Alice’s personal agent software. The agent has been instructed to “ring” her cell phone, home PC and work PC, all in parallel. To complete the call to the cell phone, Alice has instructed her agent to use the cheapest gateways that support credit card billing. The agent finds an appropriate gateway, and rings Alice at her cell phone, home PC, and work PC, all at once. Alice picks up in her car, joining the call voice-only. During the conversation, Joe remembers that a video segment from a recent IEEE tutorial presentation would be helpful. He finds the media server with the content, and plays the media stream into the conversation. Later, John decides to leave the call. He transfers Joe and Alice to a web page containing the additional information on the IEEE tutorial. Joe’s web browser jumps to the page, and Alice’s phone displays a text-only version, which they continue to discuss. Joe then decides to add Bob to the call. Bob

is not available, but his agent returns a web page containing his appointments, along with a hyperlink to a voicemail service.

The services contained in the call scenario above require many protocol components in order to work. In this paper, we examine the various protocols and discuss how they fit into the broader picture. First, a signaling protocol is needed to allow Joe to call John, and to establish a multimedia session so they can exchange audio and video. We discuss signaling protocols in section 2. The actual audio and video are exchanged between session participants using a transport protocol called RTP, which we discuss in section 3. Directory access protocols, used to access white pages services, for example, are also important, but we do not discuss them here further, as they are the same as for email service, for example [1]. During the call, Joe brings in a media server and instructs it to play a video segment. This is accomplished using a streaming media control protocol, called the Real Time Streaming Protocol (RTSP), which we describe in section 4. The intelligent agent concept described above interacts with the signaling protocol to provide advanced services. To realize these agents, we briefly describe a call processing language in section 5. Section 6 presents the Gateway Location Protocol (GLP), which helps the agent in selecting a gateway for terminating the call from the Internet on the telephone network.

This tutorial does not cover the protocols that allow controllers and signaling gateways (gateways connecting to the PSTN at the signaling layer) to communicate. Proposals for such protocols are being discussed within the IETF at this time, including MGCP [2] and MDCP [3]. Other protocols, such as those for billing and authentication, are also beyond the scope of this survey.

## 2 Signaling Protocols

Signaling protocols are at the heart of Internet telephony and distinguish it from other services. They play several roles [4], discussed below.

**User location:** If user $A$ wishes to communicate with user $B$, $A$ first needs to find out where $B$ is currently located on the network, so that the session establishment request (below) can reach him. This function is known as *user location*. Users can be in different places at different times, and even reachable by several means at the same time (by work PC or by traditional phone). This function is particularly important for users whose PCs do not have a permanent IP address. (Almost all modem connections, including ADSL and cable modems, assign addresses to PCs dynamically, using the Dynamic Host Configuration Protocol (DHCP) [5].)

**Session establishment:** The signaling protocol allows the called party to accept the call, reject it, or redirect it to another person, voicemail or a web page. (Generally, the terms "call" and "session" are used interchangeably in this paper, although "session" has a somehat wider meaning, including for example, a group of hosts listening to an "Internet radio" multicast.)

**Session negotiation:** The multimedia session being set up can comprise different media streams, including audio, video and shared applications. Each of these media streams may use a variety of different speech and video compression algorithms, and take place on different multicast or unicast addresses and ports. The process of session negotiation allows the parties involved to settle on a set of session parameters. This process is also sometimes known as *capabilities exchange*.

**Call participant management:** New members can be added to a session, and existing members can leave a session.

**Feature invocation:** Call features, such as hold, transfer, and mute, require communication between parties.

Several protocols exist to fill this need. One is the International Telecommunications Union (ITU) Recommendation H.323 [6], which describes a set of protocols. The Internet Engineering Task Force (IETF) has defined two protocols to perform many of the above tasks: the Session Initiation Protocol (SIP) [7], and the Session Description Protocol (SDP) [8]. A detailed comparison of SIP/SDP and H.323 can be found in [9]. In this article, we focus on the IETF protocols. Excellent articles on H.323 can be found in [**?**].

### 2.1 Session Initiation Protocol (SIP)

As its name implies, SIP is used to initiate a session between users. It provides for user location services (this is its greatest strength, in fact), call establishment, call participant management (using a SIP extension [10]), and limited feature invocation. Interestingly, SIP does not define the type of session that is established. SIP can just as easily establish an interactive gaming session as an audio/video conference.

Each SIP request consists of a set of header fields that describe the call as a whole followed by a message body which describes the individual media sessions that make up the call. Currently, the Session Description Protocol (Section 2.2) is used, but consenting parties may agree on another capability exchange protocol.

SIP is a client-server protocol, similar in both syntax and semantics to HTTP [11]. Requests are generated by one entity (the client), and sent to a receiving entity (the server). The server processes the requests, and the sends a response to the client. A request and the responses which follow it are called a *transaction*. The software on an end system that interacts with a human user is known as a *user agent*. A user agent contains two components, a user agent client (UAC), and user agent server (UAS). The UAC is responsible for initiating calls (sending requests), and the UAS is responsible for answering calls (sending responses). A typical Internet telephony appliance or application contains both a UAS and a UAC. (Note that this differs from a web browser, which acts only as a client.)

Within the network, there are three types of servers. A *registration server* receives updates on the current locations of users. A *proxy server* receives requests and forwards them to another server (called a *next-hop server*), which has more precise location information about the callee. The next-hop server might be another proxy server, a user agent server, or a redirect server. A *redirect server* also receives requests, and determines a next-hop server. However, instead of forwarding the request there, it returns the address of the next hop server to the client. The primary function of proxy and redirect servers is *call routing* - the determination of the set of servers to traverse in order to complete the call. A proxy or redirect server can use any means at its disposal to determine the next-hop server, including executing programs and consulting databases. A SIP proxy server can also *fork* a request, sending copies to multiple next-hop servers at once. This allows a call setup request to try many different locations at once. The first location to answer is connected with the calling party.

As in HTTP, the client requests invoke *methods* (commands) on the server. SIP defines several methods. INVITE invites a user to a call. BYE terminates a connection between two users in a call. OPTIONS solicits information about capabilities, but does not set up a call. ACK is used for reliable message exchanges for invitations. CANCEL terminates a search for a user. Finally, REGISTER conveys information about a user's location to a SIP registration server.

A client sets up a call by issuing an INVITE request. This request contains header fields used to convey information about the call. The most important header fields are To and From, which contain the callee's and caller's address, respectively. The Subject header field identifies the subject of the call. The Call-ID header field which contains a unique call identifier, and the CSeq header field contains a sequence number. The Contact header field lists addresses where a user can be contacted. It is placed in responses from a redirect server, for example. The Require header field is used for negotiation of protocol features, providing extensibility. The Content-Length and Content-Type header fields are used to convey information about the body of the message. The body contains a description of the session which is to be established.

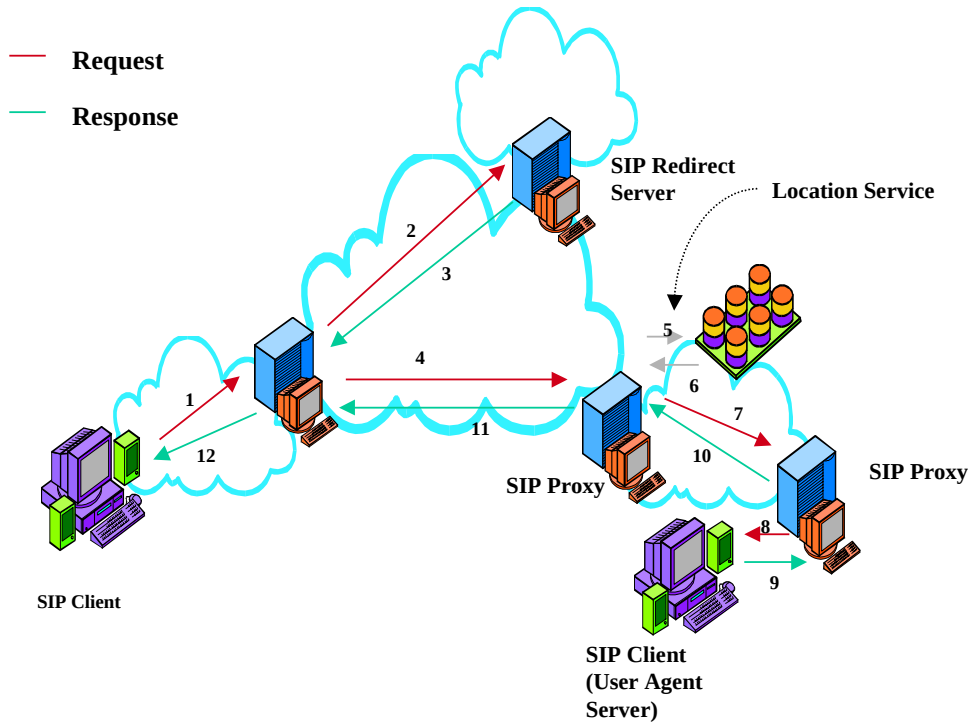


Figure 1: SIP Operation

Extensions can be defined with new header fields. One such extension, used for call control [10] define several new headers used for feature invocation (such as call transfer), and multi-party conferencing [12, 13].

A typical SIP transaction is depicted in Figure 1. A SIP user agent creates an INVITE request for `sip:joe@company.com`. This request is forwarded to a local proxy (1). This proxy looks up `company.com` in DNS, and obtains the IP address of a server handling SIP requests for this domain. It then proxies the request to this server (2). The server for `company.com` knows about the user `joe`, but this user is currently logged in as `j.user@university.edu`. So, the server redirects the proxy (3) to try this address. The local proxy looks up `university.edu` in DNS, and obtains the IP address of its SIP server. The request is then proxied there (4). The university server consults a local database (5), which indicates (6) that `j.user@university.edu` is known locally as `j.smith@cs.university.edu`. So, the main university server proxies the request to the computer science server (7). This server knows the IP address where the user is currently logged in, so it proxies the request there (8). The user accepts the call, and the response is returned through the proxy chain (9),(10),(11),(12) to the caller.

## 2.2 Session Description Protocol (SDP)

The Session Description Protocol (SDP) is used to describe multimedia sessions, both for telephony and for distribution applications like “Internet radio”. The protocol includes information about:

**Media streams:** A multimedia session can contain many media streams, for example two audio streams, a video stream and a whiteboard session. SDP conveys the number and type of each media stream. It currently defines audio, video, data, control, and application as stream types, similar to MIME types used for Internet mail.

**Addresses:** For each stream, the destination address (unicast or multicast) is indicated. Note that the addresses for different media streams may differ, so that a user may, for example, receive audio on a

low-delay Internet telephone appliance and video on a workstation.

**Ports:** For each stream, the UDP port numbers for sending and/or receiving are indicated.

**Payload types:** The media formats which can be used during the session are also conveyed. For unicast sessions ("traditional" IP telephony), this list is called a *capability set*.

**Start and stop times:** For broadcast-style sessions like a television program) the start, stop, and repeat times of the session are conveyed. Thus, one can announce or invite to a weekly TV show or a Tuesday/Thursday lecture[1].

**Originator:** For broadcast-style sessions, the session description names the originator of the session and how that person can be contacted, e.g., in case of technical difficulties.

SDP conveys this information in a simple textual format. In fact, the acronym for SDP is a misnomer, since SDP is more of a description format. When a call is setup using SIP, the INVITE message contains an SDP body describing the session parameters acceptable to the caller. The response from the called party contains a modified version of this description, incorporating the capabilities of the called party. Figure 2 shows an example.

```
v=0
o=g.bell 87728 8772 IN IP4 132.151.1.19
s=Come here, Watson!
u=http://www.ietf.org
e=g.bell@bell-telephone.com
c=IN IP4 132.151.1.19
b=CT:64
t=3086272736 0
k=clear:manhole cover
m=audio 3456 RTP/AVP 96
a=rtpmap:96 VDVI/8000/1
m=video 3458 RTP/AVP 31
m=application 32416 udp wb
a=orient:portrait
```

Figure 2: Example SDP Description

The v line is a version identifier for the session. The o line is a set of values which form a unique identifier for the session. The u and e lines given the URL and email addresses for further information about the session. The c line indicates the address for the session, the b line indicates the bandwidth (64 kb/s in this case), and the t line the start and stop times (where 0 means that the session continues indefinitely). The k line conveys the encryption key for the session. There are three m lines, each of which identifies a media stream type (audio, video, and whiteboard application), the port number for that stream, the protocol, and a list of payload types. The a line specifies an attribute. For example, the line below the audio stream definition defines the codec parameters for RTP payload type 96.

[1] SIP can be used not just to make a traditional phone call, but a caller can also invite others to, say, a TV program, without the caller and callee talking to each other.

# 3 Real Time Transport Protocol (RTP)

The Real Time Transport Protocol (RTP) [14], as the name implies, supports the transport of real-time media (such as audio and video) over packet networks. (It is also used by H.323.)

The process of transport involves taking the bitstream generated by the media encoder, breaking it into packets, sending the packets across the network, and then recovering the bitream at the receiver. The process is complex because packets can be lost, delayed by variable amounts, and reordered in the network. The transport protocol must allow the receiver to detect these losses. It must also convey timing information so that the receiver can correctly compensate for jitter (variability in delay). To assist in this function, RTP defines the formatting of the packets sent across the network. The packets contain the media information (called the RTP payload), along with an RTP header. This header provides information to the receiver that allows it to reconstruct the media. RTP also specifies how the codec bitstreams are broken up into packets [15]. RTP was also "engineered" for multicast. This means it works in conferencing applications and broadcast environments where multicast is used to distribute the media. Its important to note that RTP does not try to reserve resources in the network to avoid packet loss and jitter; rather, it allows the receiver to recover in the presence of loss and jitter.

RTP plays a key component in any Internet telephony system. It is effectively at the "heart" of the application - moving the actual voice among participants. The relationship between the signaling protocol and RTP is that the signaling protocols are used to establish the parameters for RTP transport.

RTP provides a number of specific functions:

**Sequencing:** Each RTP packet contains a sequence number. This can be used for loss detection and compensation for re-ordering.

**Intra-media synchronization:** Packets within the same stream may suffer different delays (jitter). Applications use playout buffers [16, 17, 18] to compensate for delay jitter. They need the timestamps provided by RTP to measure it.

**Payload identification:** In the Internet, network conditions such as packet loss and delay vary, even during the duration of a single call. Speech and video codecs differ in their ability to work properly under various loss conditions. Therefore, it is desirable to be able to change the encoding for the media (the "payload" of RTP) dynamically as network conditions vary. To support this, RTP contains a payload type identifier in each packet to describe the encoding of the media.

**Frame indication:** Video and audio are sent in logical units called frames. It is necessary to indicate to a receiver where the beginning or end of a frame is, in order to aid in synchronized delivery to higher layers. A frame marker bit is provided for this purpose.

**Source identification:** In a multicast session, many users are participating. There must be a way for a packet to contain an indicator of which participant sent it. An identifier, called the *Synchronization Source* (SSRC) is provided for this purpose.

RTP also has a companion control protocol, called the Real Time Control Protocol (RTCP). RTCP provides additional information to session participants. In particular, it provides:

**QoS feedback:** Receivers in a session use RTCP to report back the quality of their reception from each sender. This information includes the number of lost packets, jitter, and round trip delays. This information can be used by senders for adaptive applications [19, 20, 21] which adjust encoding rates and other parameters based on feedback.

**Inter-media synchronization:** For flexibility, audio and video are often carried in separate packet streams, but they need to be synchronized at the receiver to provide "lip-sync". The necessary information for the synchronization of sources, even if originating from different servers, is provided by RTCP.

**Identification:** RTCP packets contain information such as the email address, phone number, and full name of the participant. This allows session participants to learn the identities of the other participants in the session.

**Session control:** RTCP allows participants to indicate that they are leaving a session (through a BYE RTCP packet). Participants can also send small notes to each other, such as "stepping out of the office".

RTCP mandates that all session participants (including those who send media, and those who just listen) send a packet periodically which contains the information described above. These packets are sent to the same address (multicast or unicast) as the RTP media, but on a different port. The information is sent periodically since it contains time-sensitive information - such as reception quality - which becomes stale after some amount of time. However, a participant cannot just send an RTCP packet with a fixed period. Since RTP is used in multicast groups, there could be sessions (like a large lecture) with hundreds or thousands of participants. If each one were to send a packet with a fixed period, the network would become swamped with RTCP packets. To fix this, RTCP specifies an algorithm that allows the period to increase in larger groups [22].

# 4 Real Time Streaming Protocol (RTSP)

The Real Time Streaming Protocol (RTSP) [23] is used to control a *stored media server*. A stored media server is a device capable of playing pre-recorded media from disk to the network, and recording multimedia content to disk. RTSP offers controls similar to those in a VCR remote control. A client can instruct the server to play, record, fast-forward, rewind, and pause. It can also configure the server with the IP addresses, UDP ports, and speech codecs to use to deliver (or record) the media. Typically, media is sent from the media server using RTP.

Stored media has a number of applications in Internet telephony:

- A media server can record the content of a conference for future reference.
- Media servers can play media into an existing conference. For example, if participants in a multiparty conference are discussing a movie, it would be useful to be able to bring a media server into the conference, and have it play portions of the movie into the conference (This is done in the introductory example, where Joe has the media server play the IEEE tutorial into the conference).
- Media servers can record voicemail. RTSP clients can use the protocol to control playback of the message. This would allow users to listen to their voicemail, and rewind to a critical part (such as a phone number in the message). RTSP can also be used to record the incoming or outgoing voice mail message.

A client executes the following steps to cause a media server to play back content:

**Obtain the presentation description.** The first step is to obtain the presentation description. This description enumerates the various components of the session. As an example, a classroom presentation might contain three components - a document camera, a video view of the professor and the audio. For each component, the description defines the media parameters needed to decode the component,

such as the codec type and frame rate. The presentation description can be obtained in several ways. The client can issue a DESCRIBE request to the server, which causes the server to return a description. It is also possible to obtain a description through other means, such as a web page.

**Setup the server.** Once it has obtained the description, the client can issue a SETUP request to the server. This request establishes the destination where the media should be delivered. The destination includes the IP address (unicast or multicast), the port numbers, protocols (usually, but not necessarily, RTP over UDP), TTL, and number of multicast layers. In the response to the SETUP message, the server returns a session id. This id is used in further requests.

**Issue media requests.** After the stream has been set up, the client can issue media requests to the server. These include operations such as PLAY, RECORD, and PAUSE. The PLAY method encompasses seek, fast forward and reverse, in addition to straight play. This is accomplished by including a Scale header which indicates the time speedup (or speeddown) at which the media should be played. The Range header specifies where in the stream playback should start from.

**Teardown.** Once interaction with the server is complete, the client issues a TEARDOWN request. This request destroys any state associated with the session. Further requests for the given session id will no longer be valid.

# 5 Call Processing Language (CPL)

The Call Processing Language (CPL) [24, 25] is a scripting language that allows end users to specify the behavior of call agents that execute on their behalf. The call agents are invoked when a call arrives at a SIP server. These agents execute the instructions contained in the CPL. This allows an end user to specify their own call services. For example, a user can instruct the agent to connect a call by trying a cell phone, home PC, and work PC, all at once.

# 6 Gateway Location Protocol (GLP)

SIP allows a user on the Internet to call another user, also on the Internet. What if an Internet user wishes to call someone not on the Internet, but rather, on the telephone network? In this case, an Internet telephony gateway is needed. This device is capable of converting signaling and media between a packet network and the telephone network (PSTN, "public switched telephone network"). We anticipate that many gateways will be deployed in the future by ISPs and telephone companies.

To complete a call to a PSTN endpoint, an IP host must send a SIP invitation to the gateway. However, given a phone number to call, how does the caller find and select one of the many gateways to complete the call? In theory, each gateway could dial (almost) any phone number, but the caller may want to minimize the distance of the PSTN leg of the call, for example. This function is supported by the Gateway Location Protocol (GLP).

An overall architecture for GLP is shown in figure 3. In that architecture, there are a number of Internet telephony domains in the Internet, each of which is under the control of a single authority. Each domain has some number of IP telephony gateways that provide connectivity between Internet and PSTN. Each domain also has some number of IP users and some number of *location servers* (LS). The LSes in a domain know about the gateways in their own domains, by means of an *intra-domain protocol*, such as the Service Location Protocol (SLP) [26]. The intra-domain protocol propagates information from the gateways to the location servers within a domain.

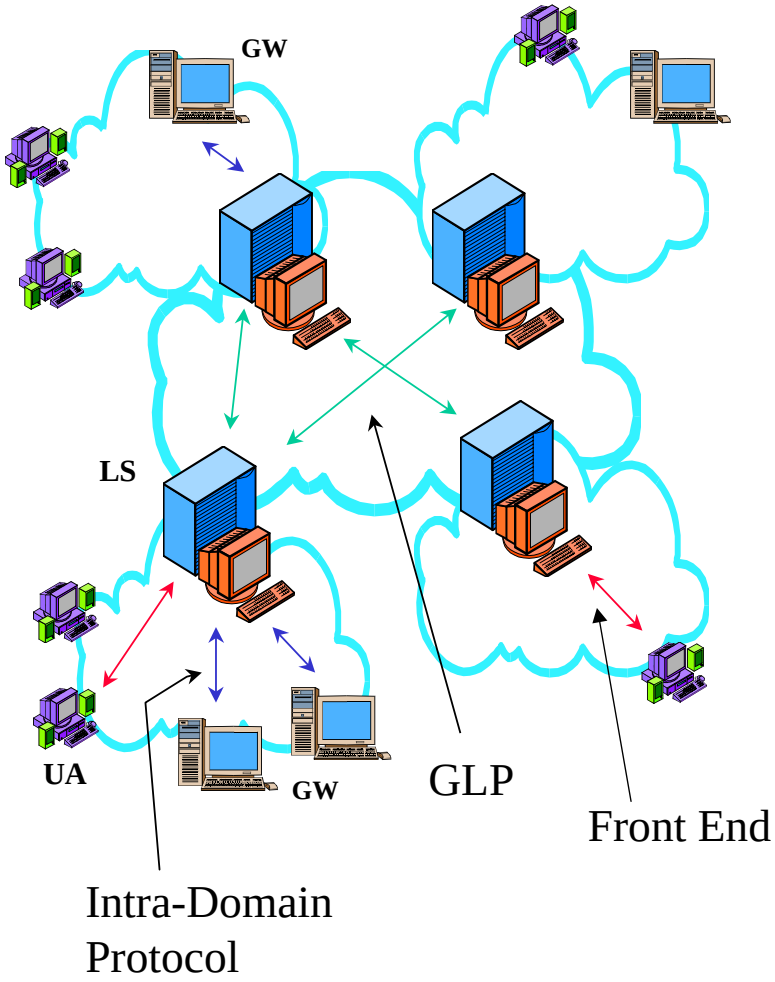


Figure 3: GLP Operation

Unfortunately, it is unlikely that a single administrative domain will have access to enough gateways to complete calls to all possible telephone numbers. As a result, users in one administrative domain can make use of gateways in another administrative domain. This usually requires pre-established business relationships between domains. Once the agreements are in place, it is necessary for the LS in one domain to exchange information about its gateways with the LS in another domain. An LS can then take this information and exchange it with other LS's that it has an established relationship with. The protocol used for these exchanges the Gateway Location Protocol (GLP) [27]. GLP allows an LS to build up a database of gateways in other domains. The database contains entries with the IP address of the gateway, a range of numbers the gateway is willing to terminate, and attributes that describe the gateway. These attributes include signaling protocols, cost information, and provider identifiers, among others. An LS can use the attributes to decide which gateways to use to terminate a call to a particular number. The information can also be used to determine which gateways to further advertise to other LS's. Both of these decisions are embodied in the *policy* that directs the behavior of the LS.

When a client within the domain wishes to make a call to a number in the phone network, it can proceed in several ways:

**Lightweight Directory Access Protocol (LDAP):** The database of the LS is made available through LDAP [1]. The calling client queries this database with the destination phone number, and the LS returns the

IP address of the gateway. The client can then send a SIP invitation to that address.

**SIP:** Rather then sending a SIP invitation directly to the gateway, the caller sends it to the LS. The invitation contains the desired destination phone number. The LS consults its database, finds the right gateway, and proxies the call to it. In this case, the LS also acts as a SIP proxy. By acting as a proxy, the LS hides the gateway selection process from the caller. The caller application doesn't need to know whether the address being called is a phone number or a SIP URL. In either case, the invitation is sent to the local proxy.

**Web pages:** The LS can make its database available through web pages. A user that wishes to make a call browses the web page, finds the gateway it likes (perhaps this can be done through a web form), and the LS returns the address of the gateway on the web page. The user copies this address to their SIP software, and completes a call to the gateway.

The GLP is just beginning the process of specification. As it is similar to existing inter-domain IP routing protocols, such as BGP4 [28], it is likely to borrow heavily from them [29, 30].

# 7 Conclusion

Providing an integrated Internet telephony service is no small task. It requires signaling protocols, transport protocols, directory protocols, service specification languages, gateway discovery protocols, and a host of other mechanisms. In this paper, we have provided an overview of some of the protocols being developed to solve these problems, and have demonstrated how they work together as building blocks to provide the infrastructure for telephone services in the Internet. It should be noted that almost all of these protocols have uses outside Internet telephony. For example, SDP and RTP are used for broadcasting streaming media, RTSP for controlling media-on-demand servers and GLP may turn out to be useful as a wide-area service location protocol for locating media translators or web server replicas. Due to space constraints, we have not covered protocols and architectures that assure "telephone quality", such as RSVP [31, 32] or differentiated services [33], even though Internet telephony may well be a primary "customer" of these resource reservation mechanisms.

# References


[1] W. Yeong, T. Howes, and S. Kille, "Lightweight directory access protocol," Request for Comments (Draft Standard) 1777, Internet Engineering Task Force, Mar. 1995.

[2] C. Huitema, J. Cameron, P. Mouchtaris, and D. Smyk., "An architecture for internet telephony service for residential customers," *IEEE Network*, vol. 13, May/June 1999.

[3] P. Sijben, M. Buckley, J. Wachter, J. Segers, C. Li, and R. Coldren, "Toward the PSTN/Internet internetworking MEDIA DEVICE CONTROL PROTOCOL," Internet Draft, Internet Engineering Task Force, Feb. 1999. Work in progress.

[4] H. Schulzrinne and J. Rosenberg, "Internet telephony: Architecture and protocols – an IETF perspective," *Computer Networks and ISDN Systems*, vol. 31, pp. 237–255, Feb. 1999.

[5] R. Droms, "Dynamic host configuration protocol," Request for Comments (Proposed Standard) 1541, Internet Engineering Task Force, Oct. 1993.

[6] International Telecommunication Union, "Visual telephone systems and equipment for local area networks which provide a non-guaranteed quality of service," Recommendation H.323, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, May 1996.

[7] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: session initiation protocol," Request for Comments (Proposed Standard) 2543, Internet Engineering Task Force, Mar. 1999.

[8] M. Handley and V. Jacobson, "SDP: session description protocol," Request for Comments (Proposed Standard) 2327, Internet Engineering Task Force, Apr. 1998.

[9] H. Schulzrinne and J. Rosenberg, "A comparison of SIP and H.323 for internet telephony," in *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, (Cambridge, England), July 1998.

[10] H. Schulzrinne and J. Rosenberg, "SIP call control services," Internet Draft, Internet Engineering Task Force, Feb. 1998. Work in progress.

[11] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee, "Hypertext transfer protocol – HTTP/1.1," Request for Comments (Proposed Standard) 2068, Internet Engineering Task Force, Jan. 1997.

[12] H. Schulzrinne and J. Rosenberg, "Signaling for internet telephony," Technical Report CUCS-005-98, Columbia University, New York, New York, Feb. 1998.

[13] H. Schulzrinne and J. Rosenberg, "Signaling for internet telephony," in *International Conference on Network Protocols (ICNP)*, (Austin, Texas), Oct. 1998.

[14] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: a transport protocol for real-time applications," Request for Comments (Proposed Standard) 1889, Internet Engineering Task Force, Jan. 1996.

[15] H. Schulzrinne, "RTP profile for audio and video conferences with minimal control," Request for Comments (Proposed Standard) 1890, Internet Engineering Task Force, Jan. 1996.

[16] R. Ramjee, J. Kurose, D. Towsley, and H. Schulzrinne, "Adaptive playout mechanisms for packetized audio applications in wide-area networks," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, (Toronto, Canada), pp. 680–688, IEEE Computer Society Press, Los Alamitos, California, June 1994.

[17] W. A. Montgomery, "Techniques for packet voice synchronization," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, pp. 1022–1028, Dec. 1983.

[18] S. B. Moon, J. Kurose, and D. Towsley, "Packet audio playout delay adjustment: performance bounds and algorithms," *ACM/Springer Multimedia Systems*, vol. 5, pp. 17–28, Jan. 1998.

[19] J.-C. Bolot and A. V. Garcia, "Control mechanisms for packet audio in the internet," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, (San Fransisco, California), Mar. 1996.

[20] I. Busse, B. Deffner, and H. Schulzrinne, "Dynamic QoS control of multimedia applications based on RTP," *Computer Communications*, vol. 19, pp. 49–58, Jan. 1996.

[21] C. Perkins and O. Hodson, "Options for repair of streaming media," Request for Comments (Informational) 2354, Internet Engineering Task Force, June 1998.

[22] J. Rosenberg and H. Schulzrinne, "Timer reconsideration for enhanced RTP scalability," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, (San Francisco, California), March/April 1998.

[23] H. Schulzrinne, A. Rao, and R. Lanphier, "Real time streaming protocol (RTSP)," Request for Comments (Proposed Standard) 2326, Internet Engineering Task Force, Apr. 1998.

[24] J. Lennox and H. Schulzrinne, "CPL: a language for user control of internet telephony services," Internet Draft, Internet Engineering Task Force, Mar. 1999. Work in progress.

[25] J. Rosenberg, J. Lennox, and H. Schulzrinne, "Programming internet telephony services," *IEEE Network*, vol. 13, May/June 1999.

[26] J. Veizades, E. Guttman, C. Perkins, and S. Kaplan, "Service location protocol," Request for Comments (Proposed Standard) 2165, Internet Engineering Task Force, June 1997.

[27] J. Rosenberg and H. Schulzrinne, "A framework for a gateway location protocol," Internet Draft, Internet Engineering Task Force, Feb. 1999. Work in progress.

[28] Y. Rekhter and T. Li, "A border gateway protocol 4 (BGP-4)," Request for Comments (Draft Standard) 1771, Internet Engineering Task Force, Mar. 1995.

[29] M. Squire, "A gateway location protocol," Internet Draft, Internet Engineering Task Force, Feb. 1999. Work in progress.

[30] D. Hampton, D. Oran, H. Salama, and D. Shah, "The IP telephony border gateway protocol architecture," Internet Draft, Internet Engineering Task Force, Feb. 1999. Work in progress.

[31] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: a new resource ReSerVation protocol," *IEEE Network*, vol. 7, pp. 8–18, Sept. 1993.

[32] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation protocol (RSVP) – version 1 functional specification," Request for Comments (Proposed Standard) 2205, Internet Engineering Task Force, Sept. 1997.

[33] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated service," Request for Comments (Proposed Standard) 2475, Internet Engineering Task Force, Dec. 1998.