SIP Basics and Beyond

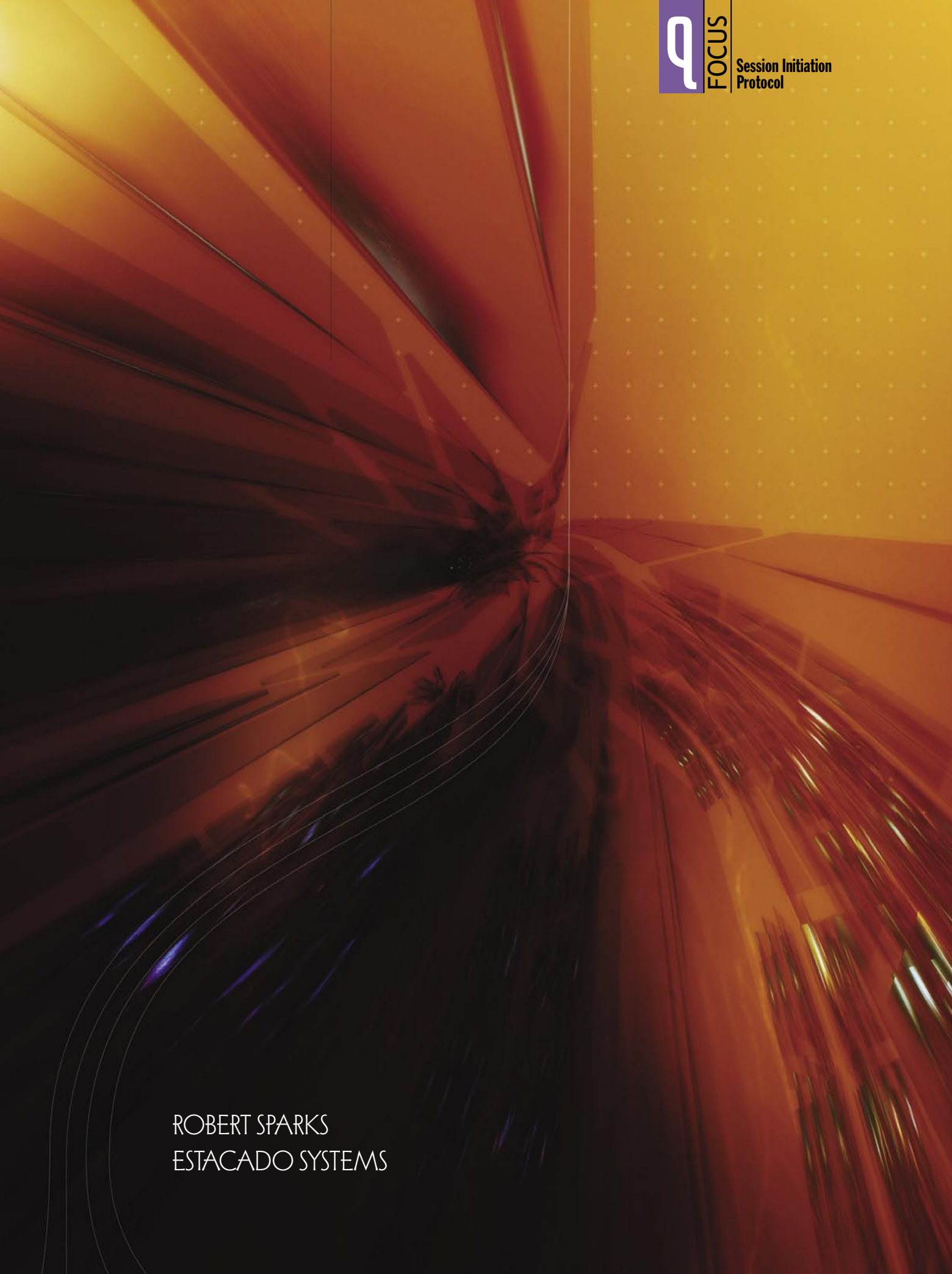
More than just a simple telephony application protocol, SIP is a framework for developing communications systems.

Chances are you're already using SIP (Session Initiation Protocol). It is one of the key innovations driving the current evolution of communications systems. Its first major use has been signaling in Internet telephony. Large carriers have been using SIP inside their networks for interconnect and trunking across long distances for several years. If you've made a long-distance call, part of that call probably used SIP.

More recently, SIP has made it into the hands of the end user through a variety of devices that look and act a lot like telephones (but have only an Ethernet jack) and service providers such as Vonage that offer telephony over any existing Internet connection. The next generation of mobile phones will use SIP as the primary signaling technology. SIP's utility does not end with telephony: it is already employed as a basic technology for IM (instant messaging) and presence.

WHAT SIP BRINGS TO THE TABLE

In the traditional circuit-switched system, telephones were required to have essentially the same set of capabilities. The mechanics of reaching them were based on their being at the end of a particular fixed section of copper wire. In the new Internet-based systems, a user can attach a phone anywhere on the Internet at any time and be immediately reachable. That phone can have a wide



ROBERT SPARKS
ESTACADO SYSTEMS

SIP Basics and Beyond

range of capabilities including many different codecs (voice encoding algorithms), support for bidirectional video, IM, and possibly even file sharing. The options available for people wishing to communicate interactively in realtime are much richer in this environment.

SIP solves two fundamental problems in establishing these realtime communication sessions. First, it helps two parties wanting to communicate find each other on the Internet (*rendezvous*). Then it allows those parties to negotiate *how* they are going to communicate (*session negotiation*). The majority of these negotiations in systems running today result in voice sessions, but nothing prevents negotiating an IM session or even an online game.

SIP is an open standard developed by the IETF (Inter-

A SIP GLOSSARY

3GPP (Third-Generation Partnership Project): A collaboration among several telephony standards organizations to produce standards for the evolution of the GSM wireless telephony network.

AoR (address of record): A well-known address for a user, suitable for printing on a business card. In SIP, this is a URI (see SIP URI) such as sip:RjS@example.net.

B2BUA (back-to-back user-agent): An intermediary that, unlike a SIP proxy, terminates and reoriginates the SIP signaling and media. For a given call, it behaves like two endpoints that are internally coupled. When these endpoints sit across a policy enforcement point, such as a firewall, the B2BUA is often called a session border controller (SBC).

Branch: A "hop" for a SIP message between devices, such as between proxies or between a proxy and an endpoint. When a request forks, each copy leaving the proxy forms a separate branch.

Codec (media coder/decoder): A codec is the algorithm used to create the representation of audio, video, or other media inside the media transport stream.

Dialog: An association between SIP endpoints created to group SIP messages related to a call or a subscription.

Dialog usage: An association between endpoints within a dialog formed to keep information specific to a particular application. Currently, there are two dialog usage types: the session usage for calls, and the subscribe usage for subscriptions.

DTMF (dual-tone multifrequency) digits: What you get when you press 1. DTMF is an encoding of standard 16-keypad key presses (most phones expose only 12 of the keys, leaving out A, B, C, and D) created by giving each row and column a unique frequency. VoIP systems can carry the actual tones in-band in the media if the codec can reproduce those frequencies. Alternatively, a special telephone-events media type defined in RFC 2833 can be used to convey pressed digits without using the tones.

Early media: Media exchanged before a call is completed. Early media can range from ring tones to the entire interaction with an interactive voice response system.

ECRIT (Emergency Context Resolution with Internet Technologies): An IETF working group chartered to work on some of the technical issues involved in providing emergency-services phone calls using technologies such as SIP.

Endpoint: An Internet device that terminates SIP signaling and, often, media. An endpoint may look like a phone, or it may be a high-density voice-mail server.

Forking: The process of forwarding a SIP request to more than one destination. This may be done sequentially (serial forking) or simultaneously (parallel forking). Forking allows more than one phone to ring when a given user is called.

Gateway: A device that translates SIP into some other realtime communication protocol. Usually, this is a SIP-to-ISUP gateway used to connect SIP networks to the PSTN.

GEOPRIV (geographic location/privacy): An IETF working group chartered to develop a way to express geographic location and permissions policy that allows users to state where they are and who can see that.

IETF (Internet Engineering Task Force): The open-standards body tasked with developing and maintaining Internet protocols.

IMS (Internet-protocol multimedia subsystem): A component of the 3GPP's architecture for providing the next version of GSM wireless phone services. The IMS brings traditional and emerging Internet services to the cellular world.

Media: The content of realtime communication. This may be voice, video, instant messaging, or any other stream of information exchanged as part of a realtime session.

Offer-answer: A negotiation for how to communicate. An offer carries a session description using SDP. The answer carries another session description that chooses ways to

net Engineering Task Force).[1] It has a large and active implementation community. Nearly 100 unique implementations were present at each of the recent SIP Forum SIPit (SIP Interoperability Test) events. These events not only improve the quality of individual implementations, but also allow us to refine the specifications and discover trouble scenarios before they affect live deployments.

SIP provides a framework for developing communications systems. It is not just a simple telephony application protocol. It is being used to construct peer-to-peer systems, residential telephony services, PBX replacement systems, and large-scale carrier next-generation networks, such as the IMS (IP Multimedia Subsystem) of the 3GPP (Third Generation Partnership Project) (see http://www.3gpp.org/).

HOW SIP WORKS

SIP is a transaction-oriented, text-based protocol. Its messages are similar in syntax to HTTP, but there is little similarity in protocol behavior. SIP endpoints, also known as UAs (user-agents), can both generate and answer requests. Most SIP servers (registrars, gateways, voicemail servers)

communicate besides those in the offer. Only one offer may be outstanding at any time in a given dialog.

Peer-to-peer SIP (P2PSIP): SIP systems with little or no "in the network" infrastructure. The rendezvous function is provided by the endpoints through technologies such as distributed hash tables rather than using a centralized proxy/registrar. The IETF is considering forming a working group to standardize peer-to-peer SIP.

Provisional responses: SIP responses to an invitation to enter a session (an INVITE request) before the final accepting or rejecting response. Provisional responses typically indicate the remote end is "ringing" but can be used to set up early media sessions.

Proxy: An intermediary that forwards SIP requests. Frequently, SIP proxies are used in conjunction with registrars to provide the rendezvous function.

Registrar: A SIP server that manages bindings between long-lived well-known AoRs and the ephemeral contact addresses endpoints get when they connect to the network.

Rendezvous: The process of two endpoints finding each other to establish realtime communication.

Retargeting: Changing the destination (the Request URI) for a SIP request while forwarding it to a proxy. Typically, retargeting is performed as part of the rendezvous process.

RTP (Realtime Transport Protocol): The protocol used to carry media between endpoints. RTP packets contain a time stamp, some sequence data, and a chunk of encoded media. RTP receivers use the information to reconstruct the media stream, accurately accounting for packet delay, jitter, and loss.

SBC (session border controller): See B2BUA.

SDP (Session Description Protocol): A format for describing the types of media to use in a session, including codecs, codec parameters, and destination IP addresses and ports for media streams. SDP is used in the offer-answer exchange that establishes or modifies a realtime communications session.

SIGCOMP (signaling compression): A framework used to compress signaling messages (such as SIP) using arbitrary compression algorithms. SIGCOMP is particularly useful in environments (such as wireless telephony) with relatively infrequent events that are sensitive to latency.

Signaling: The control channel for establishing realtime communication. SIP is a signaling protocol. Signaling establishes separate media channels.

SIMPLE (SIP for Instant Messaging and Presence Leveraging Extensions): An IETF working group creating extensions to SIP and companion protocols to facilitate building instant messaging and presence systems. SIMPLE defines the SIP Events "presence" event package, the Rich Presence Information Data (RPID) format, the XML Configuration Access Protocol (XCAP) for establishing user lists and permissions, and the Message Session Relay Protocol (MSRP).

SIP (Session Initiation Protocol): An IETF protocol used to establish realtime communications sessions. SIP allows endpoints wishing to communicate to find each other (rendezvous) and negotiate how they want to exchange media.

SIP URI: A uniform resource identifier with the scheme "sip:". An example is sip:RjS@example.net. SIP URIs identify a particular resource (RjS) at a domain (example.net). SIP systems use the domain component along with DNS to determine where to send SIP messages.

Target: The destination of a SIP request. The target is carried in the request URI, a SIP URI in the first line of each request.

Transaction: The combination of a SIP request and associated responses. SIP INVITE transactions can last arbitrarily long. All other SIP transactions have a fixed and finite lifetime, measured in seconds.

Transport protocol: The underlying Internet protocol used to carry SIP or RTP messages. Defined transport protocols for SIP include UDP, TCP, TLS, and SCTP. RTP is carried over UDP. Work is under way for both protocols to define transporting them over DTLS.

UA (user agent): See endpoint.

Usage: See dialog usage.

SIP Basics and Beyond

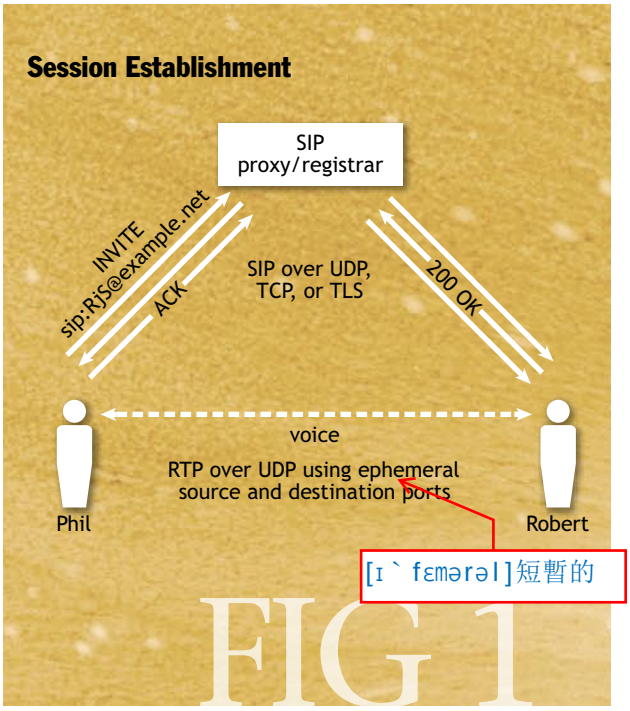
are endpoints. The only well-defined SIP element that is not an endpoint is a proxy. An element can play more than one role at a time. Typically, proxies and registrars are colocated. A proxy/registrar is an endpoint when handling registration and an intermediary when forwarding requests (proxy/registrars are often referred to as proxies). SIP messages can be carried using a variety of transports, such as UDP or TCP, and a given message can shift between transport protocols as it is forwarded through proxies. SIP itself defines transaction-level state machines and timers that invoke retransmission for providing reliability over transports such as UDP that may lose packets.

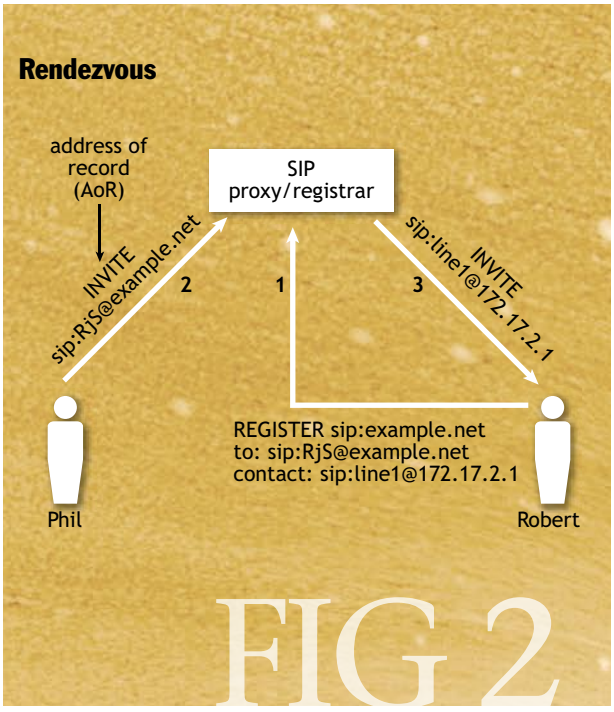
In the simplest case, two endpoints can exchange SIP messages directly. In currently deployed systems, however, most signaling between endpoints involves one or more SIP proxies. Figure 1 shows the beginning of a session being established in a basic architecture. Note that the SIP signaling and the media traverse the network independently. Voice and video media are carried using RTP (Realtime Transport Protocol).[2]

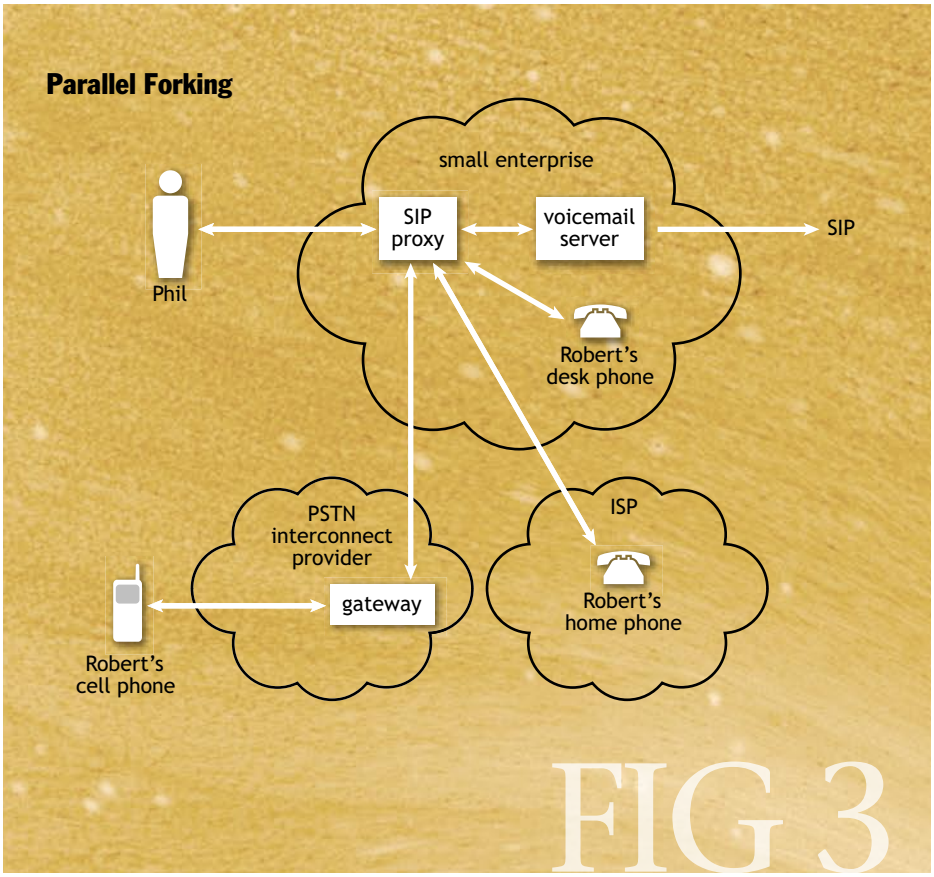
Rendezvous is established using these proxy/registrars and is based on users having a well-known AoR (address-of-record). SIP AoRs look much like e-mail addresses (for example, sip:RjS@example.net). An AoR can be placed on a business card or Web page. In Figure 2, Robert registers with the SIP services at example.net (using DNS to find them), establishing a binding between his AoR and the ephemeral IP address his phone acquired when he plugged it into some hotel network. When Phil wants to talk to Robert, all he has to know is Robert's AoR. Phil's invitation makes it to the service responsible for that AoR (again using DNS), which retargets and forwards the request to the registered contact, making that phone ring.

More than one endpoint can be registered with an AoR. When a request arrives for that AoR, it can be forwarded to all of them. The proxy providing service for the AoR has a great deal of flexibility in the order in which it uses each bound contact. With many deployed services, it will simply forward the request to each contact simultaneously (*parallel forking*). In other services, it will try the contacts one at a time, letting the first time out before trying the second (*serial forking*). Forking is a primary source of power in SIP's rendezvous function. It also is the source of the greatest complexity in the protocol.

In figure 3, Robert has four contacts bound to his AoR: two through the registration process we've already described, and two (a PSTN gateway and a voicemail service) through configuration. Phil's call to Robert simul-







taneously rings all three of Robert's phones and his voicemail server (which is configured to wait several seconds before sending the response that will answer the call). When Robert answers his cellphone, the proxy cancels the invitation to the other two phones and the voicemail server, causing them to stop ringing.

In general, when a proxy forks a request, it is responsible for gathering the responses from each branch and choosing the best one to return to the original request. A response conveying information that will help the requester succeed later (such as a challenge for credentials) is a better choice than a response that doesn't (such as "busy"). A success response is always the best choice. In fact, if more than one branch returns a success to an invitation to a session, the proxy has to return *all* of those responses to the requester. This special case exists by design—the goal was to avoid having someone pick up a phone to hear only dead air because of an arbitrary choice some proxy made. In this situation, the endpoint has to handle multiple calls simultaneously. It might choose one of the responses with which to continue the call and explicitly terminate the session with the other responder. Or it might mix all of the respondents into an ad-hoc local conference.

Once the rendezvous process gets a request to the right places, the endpoints enter into a negotiation for the type of media they will exchange in the session. This is done by carrying a separate protocol known as SDP (Session Description Protocol)[3] within the SIP messages and executing an offer-answer exchange.

RFC 3264

The offer indicates how the offerer wants to receive media. It describes protocols, media types, codecs, addresses, and ports. The answer says similar things about the media the answerer wants to receive, listing its own addresses and ports. The answerer selects from items in the offer, accepting some media streams and declining others. The first successful offer-answer exchange establishes a session. Additional offer-answer exchanges can modify that session (changing the codec or placing the streams on hold). The body of the message in figure 4 contains a simple offer to receive G.711-encoded media at port 49172 on pc33.atlanta.example.com.

SIP URIs

The AoR used in figure 2 is an example of a SIP URI: sip:RjS@example.net. The username portion identifies a resource at the domain that appears after the @ sign. DNS is used to convert that domain name into a transport to use (such as TCP), a port, and an IP address. RFC 3263 defines coordinating the DNS queries involved.[4] In short, the name after the @ sign is used with a NAPTR (Naming Authority Pointer) query to determine which transport protocol to use. The results are then used in an SRV (service) query to determine a port and the name of the particular server to use. That name is used in an A (or AAAA for IPv6) query to find the IP address of that server. Each of those queries can return multiple results, and there are

SIP Basics and Beyond

specified algorithms for processing them in order. The SRV results, in particular, will contain parameters that influence load balancing and failover. It is also possible to encode the transport, port, and address directly into the URI, and it is necessary to do so when an endpoint obtains an ephemeral address that has no associated DNS resource records.

SIP URIs appear in several places in SIP messages. The RURI (Request-URI) in the first line of a request (see figure 4) determines where the request is going to go. In essence, this is to whom the request is targeted. Proxies can retarget a request while forwarding it by changing the RURI.

The To and From URIs are sources of confusion. If the RURI determines whom the request is for, what does *To* mean? Originally, the To header field was intended to carry whom the request was originally targeted toward (in case some proxy retargeted the RURI before it got to its final destination). The From header field was intended to identify who sent the request. Because of a chain of decisions, however, including the way proxies were specified to behave, it became unsafe to ascribe the original meaning to those fields. Endpoints can put whatever they want there and intermediaries can cheat and change the value. Any processing based on the perceived identities in those fields can be easily subverted. In the absence of extensions to the protocol, they are best treated as opaque bits. There are extensions, notably the SIP Identity header field,[5] that make cryptographic assertions that return some meaning to those fields, particularly the From header field.

The Contact URI in this message tells the recipient where to target future requests in the dialog this message might establish. It is possible that once rendezvous has taken place, the proxy can step out of the path and the endpoints can exchange the rest of the SIP signaling directly. (See RFC 3261's discussion of Route and Record-Route for more detail on how this occurs.)

Some URIs may be GRUUs (Globally Routable User-agent URIs).[6] These URIs are similar to AoRs in that they are long-lived—they could be placed on a business card or in an e-mail signature. They are different from AoRs in that they route to exactly one endpoint. Endpoints can obtain GRUUs during registration. One important motivation for GRUUs was solving a problem with call transfer. Figure 5 shows Jean in the process of transferring Phil to Robert. Previously, Phil had called Jean and she decided he needed to talk to Robert. She put Phil on hold and called Robert, reaching him at his home phone. Now she needs to tell Phil's endpoint (UA) to call Robert. If she tells Phil's UA to use Robert's AoR, Phil's call will fork to Robert's desk, home, and voicemail (and Robert's home phone will probably return "Busy" so Phil will end up in voicemail). If she tells Phil's UA to use the contact URI from the dialog she and Robert are sharing, Phil's invitation may fail to route (that URI may indicate an address behind a NAT (network address translator) or might work only in the context of the route established by Jean's call to Robert). A GRUU for Robert's home phone is needed.

URIs in a SIP Message

request-URI (RURI)
to URI
from URI
contact URI

```
INVITE sip:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP pc33.atlanta.example.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.example.com>
From: Alice <sip:alice@atlanta.example.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.example.com
CSeq: 293482 INVITE
Contact: <sip:alice@pc33.atlanta.example.com>
Content-Type: application/sdp
Content-Length: 138

v=0
o=UserA 2890844526 2890844526 IN IP4 atlanta.example.com
s=Session SDP
c=IN IP4 pc33.atlanta.example.com
t=0 0
m=audio 49172 RTp/AVP 0
a=rtpmap:0 PCMU/8000
```

FIG 4

Figure 6 shows Robert's home UA obtaining a GRUU and providing it to Jean as part of their call. Jean refers Phil to that GRUU. Phil's call routes exactly to Robert's home UA with enough information to let the UA know this call is replacing the call it currently has with Jean.

HOW SIP IMPROVES REALTIME COMMUNICATION

By providing rendezvous and the ability to negotiate arbitrary realtime sessions, SIP enables a wide range of new communications applications. Reproducing the traditional concept of a phone call is a first step, not the end goal. SIP also provides a publish-subscribe mechanism called SIP Events that lets elements learn about state changes elsewhere in the network. For example, a service can subscribe to the dialog state of a UA to learn when it becomes available, making it easy to build a camp-on or call-back service. The most powerful event currently available, however, is presence.[7] This event, defined by the IETF's SIMPLE working group (SIP for Instant Messaging and Presence Leveraging Extensions), conveys information about the end user's availability and willingness to communicate. This RPID (rich presence information) ranges from where the user is (geographic location) to what he or she is doing (driving, meeting, or eating, for example) and even what kind of environment he or she is in (such as a movie theater). Someone wanting to communicate can use this information to choose the most appropriate time and mechanism.

Suppose Jean wants to let Robert know about a change in a conference schedule. She subscribes to Robert's presence. Robert's SIP presence server has access to his calendar and reports that Robert is currently on an airplane. She decides to call and leave a voice message, which Robert's SIP voicemail server collects, converts to an audio file, and e-mails to him.

Later Phil needs to get both Robert and Jean in a call to discuss the conference's agenda. Phil invokes a service that subscribes to the presence for all three of them. The first time they are all available at a sane hour, the service sends each of them a SIP instant message letting them know a call could happen now. If they all reply that they're willing, the service calls each of them and puts them into a conference. Jean takes the call using SIP user-agent software on her laptop. During the call, she needs to walk the group through a slide deck. Robert and Phil get their workstations to join and using SIP re-INVITEs, Jean adds an MSRP (Message Session Relay Protocol[8]) media stream to the conference.

A youth soccer team whose parents don't all have SIP phones can use the same autoconferencing application. The coach needs to get all the parents together to discuss travel to an upcoming out-of-town game. Some parents have SIP phones, ranging from software running on their home computers to simple analog terminal adapters that came with their ISP services allowing them to use their old PSTN (public switched telephone network) phones with SIP. A few simply subscribe to a PSTN gateway service that allows SIP callers to reach them on their

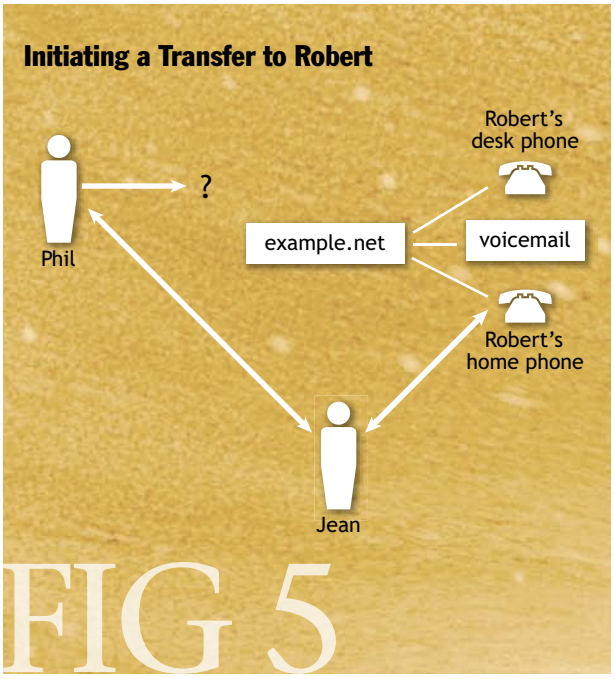


FIG 5

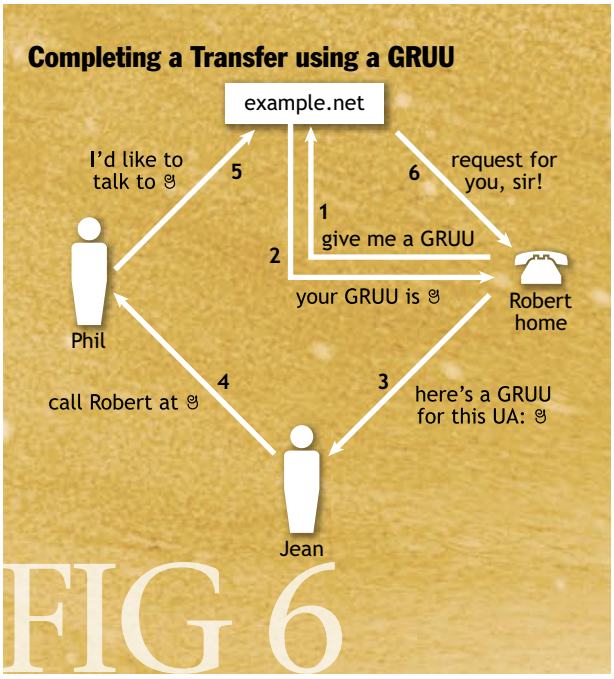


FIG 6

SIP Basics and Beyond

cellphones. Some of the parents' UAs publish presence directly and the service providers infer availability of the others based on a variety of heuristics. When a sufficient number of the parents show available, the autoconference system will call them.

These examples leverage several important architectural constructs. First, SIP user-agents and servers are full-fledged Internet devices that can easily integrate any other Internet technology or application to get the job done. This allows leveraging of lower-level infrastructure in addition to high-level applications such as calendaring and e-mail. For example, SIP already uses advanced features in DNS that help with high availability, allowing endpoints automatically to reach different elements in a cluster of servers. Other routing technologies such as anycast can be used underneath SIP to provide redundancy.

Second, SIP is a framework for the development of communications applications. It allows applications to be distributed to arbitrary elements, including the endpoints. An enterprise can integrate a presence server from one vendor and a voicemail server from another. They could even build their own voicemail service using an application server from one vendor and a media server from a second. Although a lot of early effort has gone into defining and deploying SIP using a service-provider model, the framework allows many others. A community within the IETF is defining the use of SIP in a peer-to-peer fashion (this effort is known as P2PSIP and is likely to become a working group in the near future; see the article by David Bryan on page 34 for more information).

Finally, SIP is designed to be self-healing. Almost all state changes in a running system are "soft"—they are created with a finite, negotiated lifetime and must be refreshed or they go away on their own. Within the period of a refresh cycle, systems using SIP will adapt to unexpected partial network outages or the unexpected side effects of planned large-scale administrative changes.

CURRENT ISSUES

Any evolving protocol will have a few rough edges. SIP is in a continual state of improvement through extension and clarification. At the moment, there is significant activity around making realtime communication work through NATs, securing the signaling and media channels, dealing with multiple early media streams, and making sure the right information is in place to provide emergency service calls.

Communicating through NATs. SIP carries routing information for both signaling and media in several fields scattered throughout the messages. For media in particular, these fields frequently carry explicit IP addresses and ports. A SIP endpoint behind a NAT will send messages with its private address and unmapped port, each of which will be useless to other endpoints not behind the same NAT. Furthermore, most NATs (and firewalls) will prevent incoming TCP connections and UDP traffic that doesn't line up with a temporary pinhole that outgoing UDP traffic establishes. As a result, SIP endpoints that implement only the base protocols will not work without external help when behind a NAT.

That help can come in the form of an ALG (application layer gateway) inside the NAT. A SIP-aware ALG can dive into the message, looking for places internal addresses or ports are mentioned, and "fix" them to match what's on the outside. This can be (and has been) made to work for the simplest of scenarios, but SIP is a framework protocol, not a single application. New uses arise frequently, and if the ALG doesn't know the nuances of the new use, it often prevents successful signaling.

A more powerful version of the same idea is to put a pair of user-agents back to back across that NAT or firewall point, terminating and reoriginating signaling and media on both sides. This gives the intermediate element, often called a B2BUA or session border controller, flexibility in how it moves both the SIP and media across the boundary point. These devices have been popular in recent deployments to address both policy control and interoperability concerns. Again, however, the B2BUA has to learn any new protocol features and behaviors before allowing them to pass.

Work on extensions to make NAT traversal easier for the endpoint is well under way. The STUN (Simple Traversal Underneath NATs) protocol[9] has been available for a few years and is widely used by SIP endpoints to discover what their IP address and port look like on the other side of a NAT (the endpoint can then put its external addresses in all the right places, sacrificing talking to other endpoints behind the same NAT). The TURN protocol (recently called STUN relay)[10] allows a SIP endpoint to establish a connection with a device with a publicly routable address, request an address and port on that device, and have all of its traffic relayed in and out of that

bound public address and port. The ICE (Interactive Connectivity Establishment) framework[11] allows endpoints to advertise all of their possible addresses to each other and defines an algorithm for finding the ones that work.

An endpoint using STUN, TURN, and ICE in conjunction with each other can communicate with other endpoints behind the same NAT, behind other NATs, or in the open Internet. TURN and ICE are in their final stages of standards development in the IETF's BEHAVE and MMUSIC working groups, respectively.

In the worst case, an endpoint might find itself behind a firewall or NAT that prevents all incoming traffic. Only packets belonging to a TCP stream that the endpoint opened will make it through. The SIP outbound extension provides a mechanism that allows the endpoint to establish a long-lived set of connections with a first-hop proxy on the other side of that NAT or firewall and signal that all SIP traffic destined for the endpoint must come down those connections. The GRUU extension works in conjunction with SIP outbound by providing a URI at the service provider that acts as an AoR but routes to only this endpoint (thus using the outbound connections).

SIP using STUN is widely deployed today. Endpoints supporting TURN and ICE (using both TURN and STUN) are just emerging. Very few SIP outbound implementations are available as of this writing, but more are expected at the next SIPit event in spring 2007.

Early media. SIP's initial design carried an offer in the invitation to a session (the INVITE request) and an answer in the message accepting that offer (a 200 OK). With the offer and answer in those locations, media are exchanged only when a session is accepted. The traditional PSTN allows transporting media before the call completes. This is used to play ringing sounds to the caller, and even for interacting with IVR (interactive voice response) and announcement systems.

To facilitate integration of SIP and PSTN systems, several extensions have been added to SIP to support this notion of early media. The INVITE transaction allows for any number of provisional, informational responses before the final response to the transaction, but the base protocol does not ensure their reliable delivery if they cross any UDP hops. The 100rel extension adds this reliability using a new method, PRACK, to acknowledge the receipt of the provisional response. A second extension, UPDATE, allows the requester to make changes to the session while the INVITE is waiting for the final response.

These architectural modifications bring power, and though they may make it more obvious how to build a gateway between SIP and the traditional PSTN, they introduce quite a bit of complexity for the basic SIP endpoint. Once an INVITE containing an offer has forked to multiple locations, each of them may start streaming early media. The requesting phone now has to decide which stream to render to the user (or find some sensible way to render all of them). This is similar to having multiple endpoints answer as described earlier, but more pervasive.

When forwarding a 200 OK final response to an INVITE, a proxy will try to cancel any other ringing branches. The only way to have multiple branches answer is to lose a race condition where the second phone

SIP Basics and Beyond

answers before the proxy's cancellation arrives. The potential for losing this race exists for a period of time measured in milliseconds. With early media, however, it makes no sense for the proxy to cancel branches. If the request forks, multiple early media streams will likely be the norm. Furthermore, it is difficult to tell which early media stream belongs to which provisional response. Work on securing the media streams should improve that situation.

Finally, even without forking, early media brings special conditions into consideration. Several large institutions have arranged with telephone carriers to leave a call in the ringing state the whole time the caller is interacting with their voice menu systems, collecting account numbers, selecting departments, or doing whatever work makes sense for that institution. The call will not complete until the caller begins talking to a human (in many cases, such as with large credit card companies and airlines, this is expected to happen in only a small percentage of the calls). SIP gateways, SIP phones, and the proxies between them must anticipate leaving a call (more specifically an INVITE transaction) in this ringing, uncompleted state for long periods of time, measured in tens of minutes. During this time, the endpoint needs to allow the user to send media, especially DTMF digits.

Securing the signaling and media. SIP signaling can be protected hop-by-hop by using a secure transport such as TLS (Transport Layer Security) or DTLS (Datagram TLS). These transports ensure that a third party can neither read nor change the messages crossing that hop. If a message has to cross several hops, each hop can be protected individually. In fact, a companion URI scheme called "sips" is intended to signal to each proxy along the path of a request that only TLS hops may be used. Unlike the https URI scheme, however, the proxies between each hop have full access to and can change the contents of the messages. The efficacy of sips is under question within the IETF. The current standard allows transport types other than TLS under certain conditions and provides no way to verify that a node hasn't cheated (or unintentionally violated the protocol in an insecure way).

Those parts of the SIP message not essential to routing can be encrypted end to end using S/MIME. This would allow, for example, offers and answers to be exchanged between endpoints without letting the proxies see the details of the session negotiation. Unfortunately, very few SIP implementations that use S/MIME are available.

Securing the signaling also entails protecting it from denial-of-service attacks. The IETF's SIP working group is completing a repair to the protocol to remove an attack discovered at a SIPit test event that allows an attacker to leverage forking to stimulate vast amounts of traffic—potentially on the order of 2^{70} messages—by sending as few as two requests.[12]

Much effort is going into securing the media channel, with many alternatives being evaluated. The oldest is SRTP (Secure Realtime Transport Protocol). This is a framework that allows a variety of cryptosystems to be used to encrypt and integrity-protect the payloads of RTP packets. These systems rely on the secure exchange of keying material. Many of the discussions around securing the media are focused on how to perform this key exchange. Other contenders include transmitting RTP over DTLS instead of UDP and an approach that sets up media security entirely using the media channel championed by Phil Zimmerman of PGP fame. With so many different ways to address this problem, it is highly unlikely that two implementations chosen at random will have a media-securing mechanism in common.

An important factor driving work on these deployability problems is the looming threat of spam-like activity over IM or even VoIP (sometimes called SPIM and SPIT). The best ideas so far for protection against these unwanted intrusions are based on the following: provid-

ing a strong sender identity, such as that provided by SIP Identity; securing the signaling so that identity is harder to tamper with; and securing the media so that accurate decisions can be made about which media streams to render to the end user.

Emergency services. Providers of emergency services in the traditional PSTN network leveraged several features and assumptions that do not carry into Internet telephony. Most notably, they used the property that the calling phone was at the end of a fixed, provisioned copper wire (which evolved into being at a relatively statically provisioned PBX port or near a particular wireless cell). On the Internet, there is no such luxury for identifying the calling location. Several years of work have already gone into providing the necessary extensions to distinguish emergency service calls from others, present the location of the caller in a way that won't violate privacy, and build transitional elements so that the current systems can continue to be used.

In the IETF, this work is taking place in the ECRIT and GEOPRIV working groups. For North America, the National Emergency Number Association (http://www.nena.org) offers a summary of the current issues involved in providing emergency services and drives much of the work of building the new system. The situation isn't as dire as some popular press articles like to present it, but users of SIP-based Internet telephony systems need to keep track of the limitations of their current systems.

緊迫的

SUMMARY

SIP has become a fundamental building block for realtime communication systems. It is already deployed, and the base of users is growing rapidly. Simple telephony pushed it into the network first, but that's the tip of an iceberg of applications that are being built on SIP. The flexible framework SIP provides is stimulating new ideas and enabling better communication experiences, but it comes with a different set of problems to solve than those faced in the switched telephony network.

SIP is an open standard, and the development community that has formed around it is dedicated to high levels of interoperability. The IETF and a few other organizations are working on further enhancements and solutions to the known problems in an open forum. If this article captured your interest, please join these discussions. Q

REFERENCES

1. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E. 2002. SIP: Session Initiation Protocol, RFC 3261.
2. Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V. 2003. RTP: A transport protocol for real-time applications. STD 64, RFC 3550 (July).
3. Handley, M., Jacobson, V., Perkins, C. 2006. SDP: Session Description Protocol, RFC 4566 (July).
4. Rosenberg, J., Schulzrinne, H. 2002. Session Initiation Protocol (SIP): Locating SIP servers, RFC 3263 (June).
5. Peterson, J., Jennings, C. 2006. Enhancements for authenticated identity management in the Session Initiation Protocol (SIP), RFC 4474 (August).
6. Rosenberg, J. 2006. Obtaining and using Globally Routable User Agent (UA) URIs (GRUU) in the Session Initiation Protocol (SIP), draft-ietf-sip-gruu-11 (work in progress, October).
7. Rosenberg, J. 2004. A presence event package for the Session Initiation Protocol (SIP), RFC 3856 (August).
8. Campbell, B. 2006. The Message Session Relay Protocol, draft-ietf-simple-message-sessions-16 (work in progress, October).
9. Rosenberg, J. 2006. Simple traversal underneath network address translators (NAT) (STUN), draft-ietf-behave-rfc3489bis-05 (work in progress, October).
10. Rosenberg, J. 2006. Obtaining relay addresses from simple traversal underneath NAT (STUN), draft-ietf-behave-turn-02 (work in progress, October).
11. Rosenberg, J. 2006. Interactive Connectivity Establishment (ICE): A methodology for network address translator (NAT) traversal for offer/answer protocols, draft-ietf-mmusic-ice-12 (work in progress, October).
12. Sparks, R. 2006. Addressing an amplification vulnerability in Session Initiation Protocol (SIP) forking proxies, draft-ietf-sip-fork-loop-fix-04 (work in progress, October).

LOVE IT, HATE IT? LET US KNOW

feedback@acmqueue.com or www.acmqueue.com/forums

ROBERT SPARKS is vice president, research and development, for Estacado Systems, where he manages the development of products and components for new markets. He was previously CTO at Xten Networks and has held management and research positions at Dynamicsoft, Lucent, and MCI. For the past seven years, he has focused on designing and developing SIP-based IP communications systems. He is active in standards and industry development and chairs the IETF's SIMPLE working group. He was a contributing editor to RFC 3261, which defines SIP. Sparks holds a master's degree in mathematics and a bachelor's degree in computer science from Texas A&M University.