STANDARDS

SIP's Future: Complicated And Competitive

John G. Waclawsky Ph.D.

How important is it for applications and devices to share a common signaling protocol?

A decade of development effort has produced hundreds of engineering specifications but has failed to make the IETF's Session Initiation Protocol (SIP) the signaling protocol of choice for popular voice and multimedia applications. Instead, most long distance voice sessions rely on old signaling standbys like SS7 or H.323, while the latest applications like instant messaging, peer-to-peer (P2P) file sharing and VOIP telephony are being built on newer signaling protocols, including IAX, Skype, XMPP and others.

SIP began auspiciously enough, by providing total endpoint control of the session initiation process, but the once-simple protocol has become complicated and overextended along the way. Since the 3GPP became interested in SIP, it has evolved towards providing more call control in the network core, via extensions to SIP and the use of techniques such as B2BUAs (back-to-back user agents) and private headers.

Now Internet advocates, who still want control at the edge, are frustrated by what they see as SIP becoming a tool for the core network to control user activities, and many of them have moved on to other protocols. Meanwhile, SIP's more recent proponents, who in fact want extensive call control in the core, are frustrated as well, but by the controls and complexity that SIP still leaves in edge devices. No one seems to be happy.

I believe that success for a protocol is proven by widespread usage, not by standards activities, marketing endorsements or even the number of lines of shipping code. Notwithstanding practically every major vendor's marketing claims that they "support SIP," or the recent co-opting of SIP by the ITU into the IP Multimedia Subsystem (IMS), the facts speak for themselves: SIP is not in widespread use today, and it has been avoided by virtually every successful new Internet-based application. Now even the ITU is having second thoughts about SIP, having initiated exploratory work on an alternative to SIP called H.325.

There are two basic questions here, which we will explore further. The first is how SIP could have gone so wrong, and the second is whether the concept of a universal signaling protocol is even appropriate anymore—if it ever was. But first, let's clear up any doubts about SIP's inability to achieve the kind of popularity some of its proponents still want to insist it has.

Severe Initiation Problems

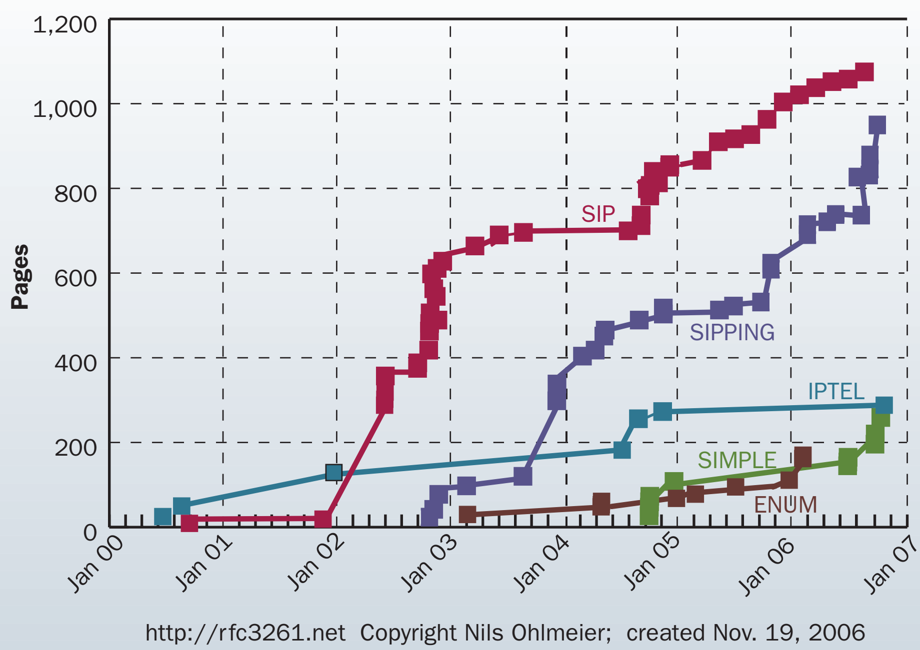
You could argue that Asterisk and Skype developers had the same business reasons anyone else would have to avoid SIP and create their own proprietary protocols—they want to develop an installed base and leverage that against their competitors. But Asterisk's IAX is open, and both Asterisk and Skype are on record as having gone their own way because SIP didn't technically satisfy their design needs. More recently, Google chose XMPP, not SIP, as the basis of Google Talk, and Skype is now expanding by targeting business environments with its own (proprietary) session initiation protocol (see the article at www.networkworld.com/news/2006/110706-skype-sets-eyes-on-enterprise.html).

Nor has SIP lived up to expectations of mix-and-match interoperability in the enterprise. Sure, every major IP-telephony vendor has implemented "basic" SIP, but they also have implemented many of the hundreds of SIP extensions. Since these additional bits of function are rarely handled the same way by different vendors, it's nearly impossible for enterprise customers to truly mix and match SIP-compliant components.

If you look closely, the major carriers have not really embraced SIP. Many still regard the ISDN-based legacy protocol H.323 as the real VOIP and videoconferencing workhorse. H.323 is also the foundation for cheap calls made using discount cards, and it integrates easily into the carrier billing systems. Verizon has made a point of emphasizing the need to specifically include "non-SIP" services in its version of next-genera-

John G. Waclawsky is the chief software architect with Motorola. The views expressed are his own. He can be reached at jgw@motorola.com.

FIGURE 1 VOIP Signaling RFC Pages (excluding obsoleted RFCs)



None of the developers of popular new Internet applications used SIP

tion convergence, Advances to IP Multimedia Subsystem (A-IMS). Finally, billable SIP services have not matured.

Last year (2006), the ITU began talking about replacing SIP. The H.325 work was started by ITU-T Study Group 16, which is the lead study group on multimedia systems and audio/video coding. The intent is to explore alternatives to SIP's complexity, inadequate error handling and diagnostics as well as poor capabilities negotiation. An additional goal is to better meet the requirements (modular, faster call establishment, better service control) for the carriers' anticipated next-generation network environments.

Currently H.325 requirements are just beginning to be discussed. In May 2006, several presentations at the Joint ITU-T Workshop and IMTC Forum entitled "H.323, SIP: Is H.325 next?" (available online, see www.itu.int/ITU-T/worksem/h325/200605/) indicated that SIP is a mistake-ridden practice attempt, and H.325 expects to get it right the next time round. If H.325 proceeds—and this has not yet been decided—then the protocol work is expected to start sometime in 2007 or 2008.

Meanwhile, Figure 1 and Table 1 illustrate the difference between supporting SIP and using SIP. Figure 1 shows the growing number of SIP-related RFC pages (the IETF's name for a specification) that have been produced through mid-November 2006. The vendors who send their representatives to the IETF all "support" SIP in this way, by participating in the ongoing (some would say never-ending) standards-setting process.

But when it comes to actually using SIP, Table 1, p. 24 tells the tale. Admittedly, it's mostly about instant messaging and presence protocols, but it makes the point. Moreover, it will have to do, since there are no comparisons of actual usage data for SIP, IAX, MGCP, H.323, etc. In Table 1, we see how competitors to the SIP for Instant Messaging and Presence (SIMPLE) protocol have dominated IM in the marketplace.

Like IPv6, SIP stacks are embedded in a lot of software, but SIP simply hasn't achieved widespread usage. Why is that?

SIP Is Complicated

In reality, no single signaling protocol has ever been crowned the universal way to do everything. Instead, several popular protocols have been developed over the years to solve specific problems. For example, SS7 is used for the PSTN user and core network signaling in 99 percent of PSTN networks, while ISDN has its own user-network signaling. H.323 is used mainly for videoconferencing and for VOIP carriage on core networks, while MGCP mainly offloads calls from the PSTN backbones onto IP networks.

From a consumer standpoint, SS7 has been a terrific success. Years of public telephony services have made SS7's simple, dependable primitives the gold standard for basic telephony: on-hook, off-hook, ring, busy, etc. It's all very basic stuff and it just works. Today, a consumer has no problem going to a store, buying a basic phone, plugging it in and achieving instant interoperability.

SIP is completely different from SS7, but in the last several years, it has come to be accepted that one of SIP's roles will be to replace SS7. SIP's

TABLE 1 Instant Messaging User Base

Service Provider	Number of Active/ Total Users	Services Offered	Source of Data
AIM	53M Active/195M Total	IM, VOIP, Presence, Video Chat	Nielsen//Net Ratings
Windows Live Messenger	29M Active/155M Total	IM, VOIP, Presence, Video Chat	Nielsen//Net Ratings
Yahoo! Messenger	21M Active	IM, VOIP, Presence, Video Chat	thestreet.com
ICQ	20M Active/400M Total	IM, VOIP, Video Chat	wikipedia
QQ	20M Peak/500M Active	IM, VOIP, Presence	nmscommunications.com (spring 2006), wikipedia
Skype	7M Peak/100M Total	IM, VOIP, Presence, Video Chat	Share.skype.com
Jabber	13.5M/21M	IM, VOIP, Presence,	wikipedia
Sametime	15M enterprise users	IM, VOIP, Presence, Video Chat	marketwire.com (1/2006)
Gadu-Gadu	5.6M	IM, VOIP, Presence	wikipedia
MXit	2M/5M logons per day	IM on mobiles	Bus Rpt (10/2006) wikipedia
Vonage	1.5M	VOIP – doesn't do IM	Zdnet.com 3/2006
Meebo	>.5M	IM, VOIP– Web based IM interoperability solution	http://radar.oreilly.com (10/2006)

developers have been trying very hard to achieve that and more, by making SIP the Swiss Army knife of IP signaling protocols. Although SIP started out simply, it has been extended again and again. These extensions bring more potential value to developers and to customers—but they also bring more code, more choices, and more state to manage in more network devices—in short, more complexity.

For example, more than 80 IETF documents (and climbing) now describe SIP and its extensions: 20 of them cover the core of SIP signaling, but there are at least seven which define extensions for PSTN issues, seven more for security, several more for conferencing, QOS, IM, presence, emergency services, call control and a large number of others that could be labeled miscellaneous or infrastructure-based. And this total doesn't include the numerous SIP-related technical specification and implementation agreement documents developed by the ITU, ETSI, 3GPP and the Multiservice Switching Forum.

Product developers who use SIP also have to face the fact that there is no "SIP certification" logo which would prove to the customer this is a SIP-compliant device or software product. SIP certification is problematic because SIP, besides being complicated, has limited error handling and fault management, and poor separation between service logic and call processing, along with no clear separation between the user-to-network and the network-to-network interfaces (UNIs and NNIs). In addition, multiple variants of SIP have been described to meet different market requirements. Adding all this up leads to innumerable interoperability issues.

And SIP Is Getting More Complicated

In November 2000, 3GPP chose SIP as the signaling protocol for the IMS architecture. Since then, IETF activity driven by 3GPP participants has shaped SIP into an IMS control plane protocol designed to mimic circuit-switched environments and to facilitate session control in the core of the network. That may have seemed like a good idea earlier in the decade, but since then, the complexity of the IMS platform and the net neutrality controversy surrounding IMS have not helped to make SIP any more popular.

Meanwhile, SIP activity in the IETF is producing bigger SIP stacks and larger message sizes. This is the price being paid to "optimize" SIP for the major carriers to use so that they can:

- service large populations of users *and*
- bill them for all aspects of their Internet activity *and*
- do so within a hierarchically controlled client-server structure.

As SIP itself becomes more confusing, the IETF and the 3GPP have begun to disagree about how to save or extend SIP further, about which extensions will accomplish that, and/or which are most useful and/or most important, and/or which will give them a competitive edge, etc. In short, the IETF and 3GPP have forked on SIP and are extending each fork of SIP into what could easily become incompatible telephony control systems.

Product designers would rather not put multiple, complex SIP stacks in a product because of the design complexity, but also because of the potential impacts on processor, memory and battery life. Nevertheless, the Nokia dual-mode E60/61/70 WiFi/GSM phone comes equipped with two SIP stacks, one for the 3GPP standard and the other for the IETF standard. Will the end user be pleased or confused by having to deal with SIP configuration choices? Is this end-user value?

Some mobile system operators have already had problems with the heavy bandwidth usage and other aspects of both SIP and H.323, and are

Product designers and engineers grumble about SIP's bandwidth consumption and its complexity

instead using H.324M (a variation of H.324 for mobile phones). For example, SIP's unnecessary instructions and numerous messages can cause mobile devices to wake up and use precious radio and battery resources.

Meanwhile, engineers grumble about SIP's bandwidth requirements and the number of messages required by its complex, per-call capability exchange. Most of these people know from experience that session initiation and VOIP can be handled more efficiently by other protocols. Some of them have downloaded Mark Spencer's Asterisk protocol and are also familiar with his published comparison of Asterisk IAX protocol to SIP, which concludes, among other things, that SIP uses more overhead, more bandwidth, more processing and more/larger messages.

Everyone knows that complexity results in more issues and errors (see *BCR* January 2007, p. 59–62). Even if no one is saying it publicly about SIP yet, they will. Eventually, more developers will accept that SIP just doesn't do what needs to be done.

Which brings us to the underlying question: Is it important that developers use some universal session initiation protocol? So far, developers of the most successful Internet-based applications—peer to peer file sharing and instant messaging—have found it easier to build their own signaling protocols or use something besides SIP.

Who Really Cares About SIP?

End users don't care whether SIP or some other signaling protocol is underneath any of their multimedia activities, or that the versions of SIP they use interoperate—they just want the applications to work. And they usually can work around a lack of application interoperability pretty easily.

For example, end users who have colleagues on several different IM systems might like to have an integrated display of all their buddy lists, but it probably isn't a high priority. By now they are used to running AIM, MSN and Yahoo, and all three at once when needed.

Another option comes from the Web portal mash-up Meebo, a start-up that lets users chat over AIM, Google Talk, MSN and Yahoo all at once. Meebo further undermines the need for IM protocol interoperability.

Application developers don't really care about interoperability with other applications either. Since the application developer makes the decisions on functionality and delivers the application, s/he is the best person to decide how to initiate, control and optimize application activity. If the application can handle its own session initiation, optimized to do exactly what it needs/wants and no more, or if the application can get the capabilities and controls easily elsewhere, why sign up for SIP's complexity, overhead and other burdens?

Of course the SIP extensions are optional and can be negotiated at session initiation, but this adds complexity, too. If it's a choice between struggling with SIP's complexity or more quickly getting to market with a proprietary product, guess which path most developers will take? The innovative developer's attitude seems to be: Why muck with all this complex stuff? I'll just initiate *my* session *my*self, *my* way.

Some application developers also may bypass SIP in order to avoid potential snooping and controls from within the network. Following an easy-to-monitor SIP control/activity "template" would open the door to such snooping and controls, especially if and when deep-packet inspection and/or IMS are deployed. Why would Internet developers want to make it easy for their potential competitors to control their traffic and/or inspect their packet flows? Any information gleaned from deep packet inspection could help competitors improve their own services by exploiting defects in others.

Arguably, a common protocol (SIP) on the server side would drive a single client software target, so a standard like SIP could be helpful to developers. The problem with this argument is that every application needs more than just SIP. For example, VOIP also needs the voice bearer and feature components and other software, and these can vary, too. So, even if you want to solve the VOIP or multimedia interoperability problem, you won't do it using SIP alone.

New Choices For Interoperability

In the old days, interoperability mostly meant adhering to the same stack of standards. Today, applications that run on IP can reach any endpoint IP address, just as those that run on HTTP can be displayed on any Web browser. This reduces the need for session- and application-layer interoperability, a need which is further reduced by the power and convenience of the modern desktop and its OS; users can run as many different programs as they like and toggle readily among them. Isn't it obvious that users and their applications don't need to interoperate by following a single signaling protocol?

Recently, there have been some moves toward interoperability of existing programs, but not with a common protocol: For example, in July of 2004, Microsoft, Yahoo and America Online forged an agreement for letting IM and presence information move between their IM services and Microsoft's then-forthcoming Office Live Communications Server 2005. The LCS 2005 would fulfill a bridging function to extend state data from one domain to another.

Application developers are likely to see SIP as a complex control protocol that is optimized for telephony and laden with optional extensions and numerous flavors. If they have a simpler choice (as, for example, Google Talk's open Jingle protocol, which is based on XMPP), then why would they choose SIP?

Remember, it was not the application developers who conceived SIP, or pushed SIP toward the goal of interoperable voice stacks. Rather, it was the standards developers, who seem to think that, by combining a bunch of services, each of which has very little usage, into the burgeoning SIP family, they will somehow drive the world to a common SIP, and that everyone else—including Asterisk, BitTorrent, Google, Microsoft and Skype—will eventually hop on board the SIP bandwagon. But what possible motivation could these

SIP standards developers think everyone will eventually jump on the SIP bandwagon

IETF Takes On SIP's Broken Behavior

Complex protocols cause interoperability problems, and SIP is no exception. Just consider the many ways you can set up a SIP call, and the numerous possibilities to represent information in SIP: parameter positions, blanks surrounding a colon, splitting headers over multiple lines, etc. These choices require heavy-duty interpretation and much more sophisticated parsers than earlier signaling protocols.

There is much more to SIP's interoperability and complexity challenges, however, as demonstrated recently in discussions at the 67th IETF conference (San Diego, November 2006), where seven sessions, as well as a special lunch session, dealt with SIP's "broken behavior." This term was often applied to SIP activity, especially in discussions about early media (the ability of SIP user agents to communicate before a SIP call is established), clipping, security and NAT/firewall traversal.

To solve the early media and clipping problems, presenters at the conference suggested a TCP/IP-inspired solution, slow start. Resolving the security issues, which are more complex, remained a work in progress. SIP uses separate security mechanisms for signaling than for the media, and it can distribute the media (SRTP) master key via SIP signaling (SIP signaling itself is secured through TLS and S/MIME). The idea of two separate security mechanisms increases overall complexity, but the biggest problem discussed at the conference was using SIP to distribute the SRTP master key, because SIP signaling is visible to all recipients, including those on any forked path(s).

SIP security issues get even more interesting when you consider that IMS plans to use SIP to establish IPSec tunnels. Because the IMS specifications for SIP security are so densely intertwined with other IMS functions, it is nearly impossible to accurately assess potential interoperability and security issues. At this point, however, it looks like SIP signaling will have to integrate tightly with the code that generates and distributes keys and with the IPSec code, resulting in even more complexity. (For more about IMS security, see *BCR* January 2007, p. 59–62)

Another of SIP's broken behaviors—the hard-coding of ports or IP addresses at the application level (used in INVITES, VIA and other SIP methods)—interacts poorly with NATs and firewalls. Additional protocols such as Simple Traversal of UDP through NAT (STUN), Traversal Using Relay NAT (TURN) and Interactive Connectivity Establishment (ICE) are under development, but they are not fully baked (TURN and ICE are not RFCs and STUN is being updated).

These broken behaviors bring more critical scrutiny to network-based SIP "redirection," in which proxy servers decide what to do with SIP requests when, for example, the recipient's phone is not answered and a timer goes off. Redirect servers add complexity, complicate interoperability and—like NATs, B2BUAs and other proxies—they clearly violate the Internet's longstanding end-to-end principle.

Conclusion

Finally, the bolted-on aspects of SIP are worrisome. These include lawful intercept, emergency services, provisioning and management. Readers with development experience know that building in function is always more efficient than adding it on later, which usually results in less-than-ideal solutions and increased complexity. Already, some SIP extensions (e.g. INFO, PRACK) have created significant angst and uncertainty in the developer community.

Sometimes the standards crowd seems to forget that programmers are people, and the best thing to do is to make a protocol as simple and as unambiguous as possible. For example, Skype doesn't have the NAT/firewall obstacles that SIP does. Skype just tunnels through, while the IETF is still struggling to solve SIP's NAT/firewall and other security issues.

In the end, all this complexity seems likely to reduce SIP's value, as developers turn elsewhere and interoperability becomes a distant dream. It seems that, if something is useful—and SIP was when it first started—people want to further its utility and extend its value. The IETF is learning that this is not always a good idea. In fact, SIP has become an extreme example of wanton extensibility, and the results are not very pretty□

Building to the SIP protocol or having a SIP client just doesn't look very important

successful services and their developers have that would make them want to switch to SIP?

It's (Still) All About The Applications

Application developers care much, much more about getting onto the Internet quickly, by using all the new stuff/trends/technology/protocols and doing it their own way, than they care about the inner workings of SIP, a 10-year-old signaling protocol (work started in the mmusic WG at the IETF in 1995) that promises something they don't even want (interoperability at the signaling protocol level with other applications). The popular new applications are not using, wanting, or waiting for SIP. They simply don't care about SIP.

A lot of the innovation on the Internet today has to do with software that forms new applications and supports new usage. Software is being re-conceptualized, generally around Web services with trends towards mash-ups and other quick solutions. For example, as reported in the Nov. 10, 2006 issue of *USA Today*, Internet creativity without SIP is continuing at a rapid pace:

- Yahoo is wrapping IM into its e-mail program (IM/e-mail mash-up), which will enable Yahoo members to see their mail contacts on-line and IM them directly. It "makes email a more social experience" according to Yahoo senior vice president Brad Garlinghouse (see www.usatoday.com/tech/news/2006-11-09-web-two-oh_x.htm).
- AOL IM (AIM) program no longer has to be downloaded to be used. Members can sign in directly at the AIM.com website and use a Web-based version to communicate with friends.
- MySpace members sign in, design their own sites, and spend hours online talking with friends (70 million monthly visitors in September 2006).

The fact is, applications, not protocol selections, drive usage. Exciting Internet-based applications like YouTube, Google Earth and MySpace aren't using SIP to initiate their sessions, and it wouldn't matter to them if SIP got yet another extension, even if that extension were aimed at their applications.

Nor do users care if their favorite applications use SIP. If my preferred video chat program doesn't interoperate with yours, no big deal. We don't even try to make them work together. Either I download yours, or you download mine and we are good to go.

This brings up my final question: Is it really necessary for applications and devices to interoperate via a common control method, like SIP? I think not. Instead, it looks to me like signaling-based interoperability is superfluous: Users and the applications they use are controlling their interactions. They just don't need SIP.

As for the future, I suspect that many applications (especially the P2P ones) will work just fine without SIP. And using SIP for sensor applications would certainly be overkill. Will my refrigerator application ever have the need to cooperate with my garage-door opener application? If I even think that might one day be useful, why should I use SIP now? Just in case?

Conclusion

None of the most popular emerging voice, data and new P2P applications uses SIP. Everywhere, developers are waking up to the fact that SIP requires difficult management, security and operation controls. Even operators and enterprise IT departments are not enamored of the complex product architectures that SIP requires in order to implement the simplest features. This is a major motivation behind H.325. Operators and enterprise IT departments want more granular feature control to better manage and control services.

Is the IETF aware they may have extended SIP past the point of usefulness? Does the IETF fiddle while SIP burns and loses its appeal due to complexity creep and an increasing lack of use/interest by Internet developers?

Building to the SIP protocol or having a SIP client just doesn't look very important. In fact, it may be far more important to allow devices to load all the clients they want and let the end user choose the service he or she wants. If you are going to maximize the value of the Internet to the end user, then you have to allow the end user access to all the varied and interesting applications on the Internet. This means the world will remain multi-client and multi-protocol, and SIP will not be successful in controlling it all□

Companies Mentioned In This Article

3GPP (www.3gpp.org)
AOL Instant Messenger (AIM—www.aim.com)
Asterisk (www.asterisk.org)
ETSI (www.etsi.org)
Google (www.google.com)
IETF (www.ietf.org)
IMTC (www.imtc.org)
ITU (www.itu.int)
Meebo (www.meebo.com)
Microsoft (www.microsoft.com)
MSN (www.msn.com)
Multiservice Switching Forum (www.msforum.org)
MySpace (www.myspace.com)
Nokia (www.nokia.com)
Skype (www.skype.com)
Verizon (www.verizon.com)
Yahoo (www.yahoo.com)
YouTube (www.youtube.com)