Leonid Consulting

SIP Tutorial

Leonid Consulting

V1.4

Leonid Consulting

Contents

Leonid Consulting

Tables and Diagrams

Leonid Consulting

Introduction

This tutorial provides a practical introduction to session initiation protocol with a focus on current non-IMS commercial network topologies. The tutorial pre-supposes a basic understanding of IP but provides references and brief explanations to help the reader along.

Leonid Consulting

SIP

A Brief Introduction

The following tutorial provides a practical introduction to the Session Initiation Protocol (SIP) and building VoIP networks with it. The first section provides an overview of how SIP operates and the second an overview of some of the most common commercial deployment models.

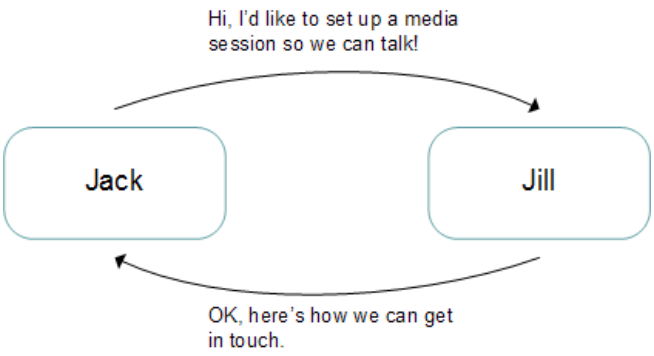
You're a telecom guy interested in this IT stuff? SIP is a signalling/control plane protocol like ISDN or SS7 that sets up media/bearer sessions. While it's specified by the IETF, it is not as strictly prescribed and vendors frequently implement their own proprietary or semi-proprietary extensions.

You're an IT guy interested in this telecom stuff? SIP is a standard published your friends at the IETF. It borrows message codes from HTTP and its partner protocol for describing media session, SDP, borrows its media description capabilities from MIME.

What is SIP?

SIP sets up and manages media sessions over IP[1], operating in a request/response model like HTTP.

Figure 1 The Basic Idea



In the abstraction above, Jack sends Jill a request, Jill processes the response, possibly invoking certain functions (not shown), and issues a response back to Jack.

Beyond the basics, SIP also accommodates pretty much all of the calling features you're used to having in a residential and business environment: call waiting, call hold, hunt groups, voicemail, etc.

Who maintains SIP?

The IETF[2] maintains SIP through RFC's. Below is the brief summary of the more important RFC's:

[1] This tutorial assumes a basic working knowledge of IP. You can piece together a lot of all this without it, but if you want to know more about IP just google it. Wikipedia is great and if you really want to go deep Cisco offers the leading study courses in IP.

[2] Internet Engineering Task force- publishes RFC's (request for comment) which define most prevalent standards in the Internet/IT Communications space.

Table 1 SIP RFC's

RFC	Description
3261	Current primary specification for SIP.
2543	Predecessor of RFC 3261. Obsolete from the standpoint of the IETF.
3262	Provides for some tuning around session setup, mostly to improve mapping in gateways that convert between SIP and legacy protocols like SS7.
3264	This RFC specifies the media session part, SDP (Session Description Protocol). Remember from a couple of paragraphs ago? Looks like MIME?
3265	Describes a notification request/response framework. This was needed for items like message waiting indicator on voicemail, presence items (buddy lists on your IM), and enterprise features like busy lamp field (a light/symbol you see on a phone's display when the line is tied up).s
3266	IPv6 support.

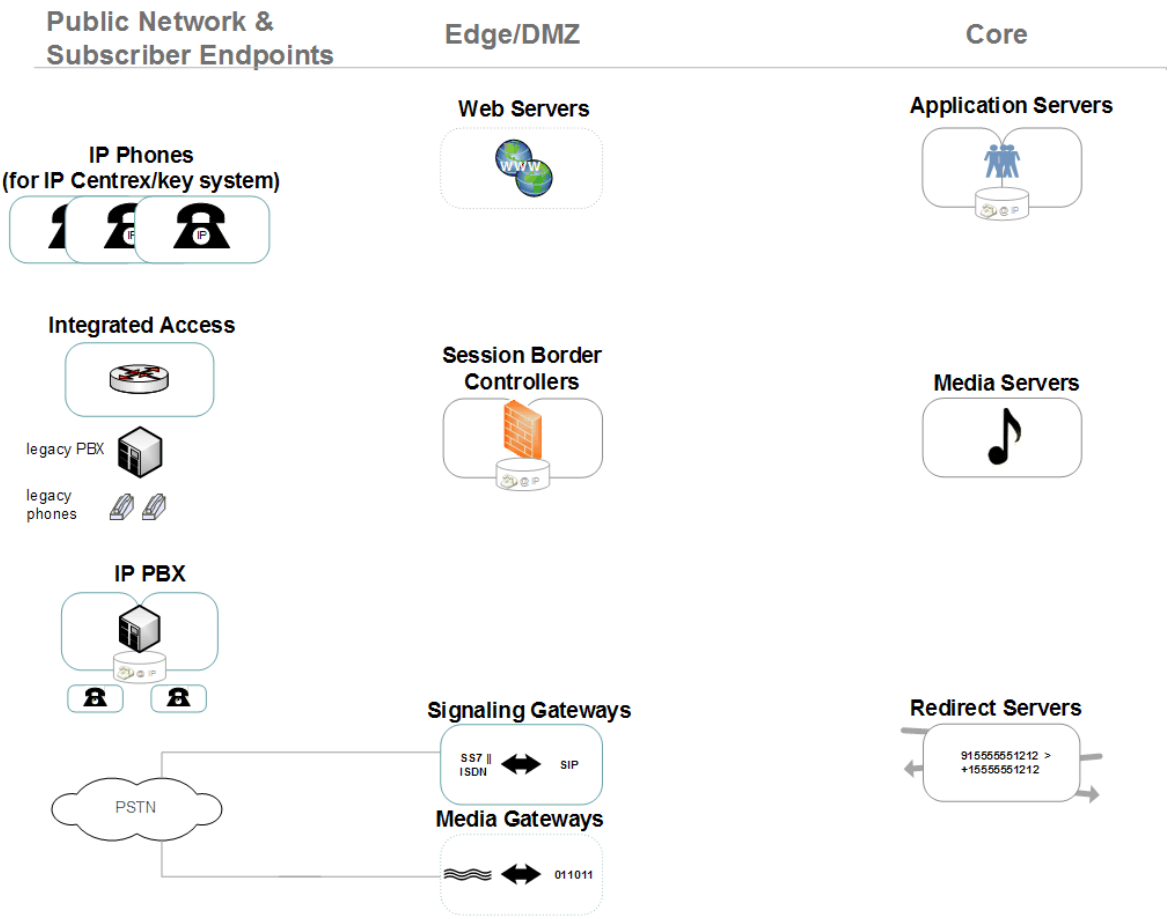
Should you go read all these if you really want to do this SIP thing? For most system designers and engineers, the answer is probably not. Certainly not "cover to cover". The introductions are good, and, of course, there is a lot of information in there, but most people use them as a reference. There's a section that follows on further reading which provides some ideas on where you may want to go for more detail. And we still have a few things to cover here!

Leonid Consulting

What are the elements of a SIP topology?

This section lays out the foundation elements of the SIP topology. Below is a typical non-IMS topology[3]. We'll refer to this topology in the description of the elements in a SIP topology (which we'll do now), for the end to end services flows (later), and in the section on SIP Networking (next section).

Figure 2 non-IMS Topology



The topology divides into three major sections

1. Public Network and Subscriber Endpoints: this is the PSTN and the devices at an individual customer's premise.
2. Edge/DMZ: the idea of an edge/DMZ is that these elements are directly accessible to the "outside world" and they control access to the more sensitive elements in the core.
3. Core: these are the core processing elements of the network.

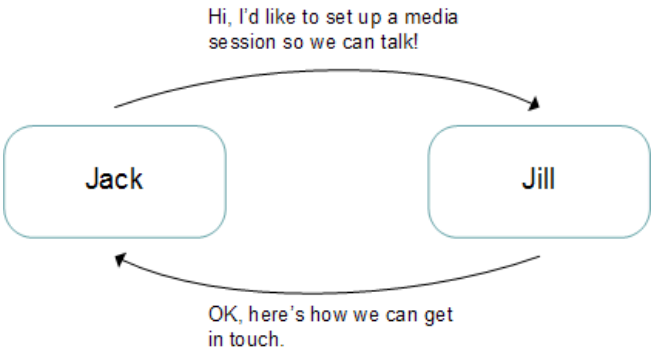
[3] IMS (IP Multimedia Subsystem) is a standard by the 3GPP. Originally designed for wireless networks, it has become increasingly popular with landline operators. It prescribes an alternative topology and service flow from the one described it this section. It does use many of the same elements. IMS' chief goal is to completely abstract the services layer from the access layer. The last section of this tutorial provides an IMS overview and sources for more information.

Leonid Consulting

Now that we have a basic orientation we're going to back into the details of what these things do and how we look at them in the SIP framework. We'll do this by looking at each of the functional building blocks in SIP, and then review the functional roles of all the elements for that functional block.

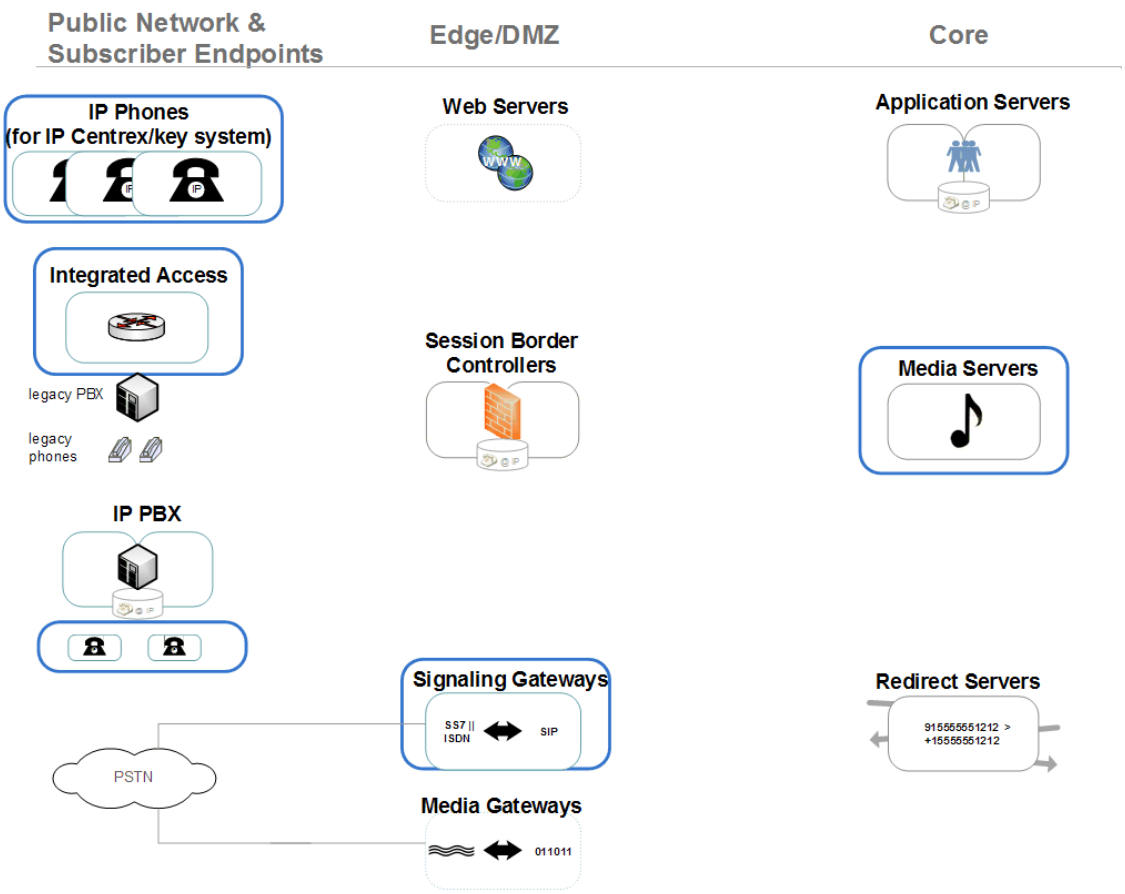
The first and most prevalent element is the SIP user agent. At Leonid, we show the user agent as a simple rectangle.

What a user agent (UA) does is simple: it initiates or terminates SIP dialogues/session. For those of you not familiar with the client/server model, when a SIP UA initiates a request for a SIP session we call it a user agent client (UAC). When it receives a request we call it a user agent server (UAS). If we go back to our Jack and Jill abstraction, Jack's the client and Jill is the server:



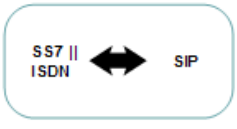
The elements with a blue rectangle around them are UA's and we'll describe their functional role below:

Figure 3 SIP User Agents



The most common functional element that is a SIP UA is a VoIP Phone. This is a phone that has an Ethernet jack and runs an IP stack as well as SIP UA process. In practice, these are desktop sets, like the Cisco 7000 series, or they're little boxes that you plug a plain old phone into and connect to your modem or router (the kind you get from Vonage, etc.).

But it's not just phones that are UA's, so are media servers and gateways. Media servers record and play back media. For example, if you're a subscriber on a SIP service and someone gets your voicemail, it would be a media server that would play that announcement and then record the message you leave.



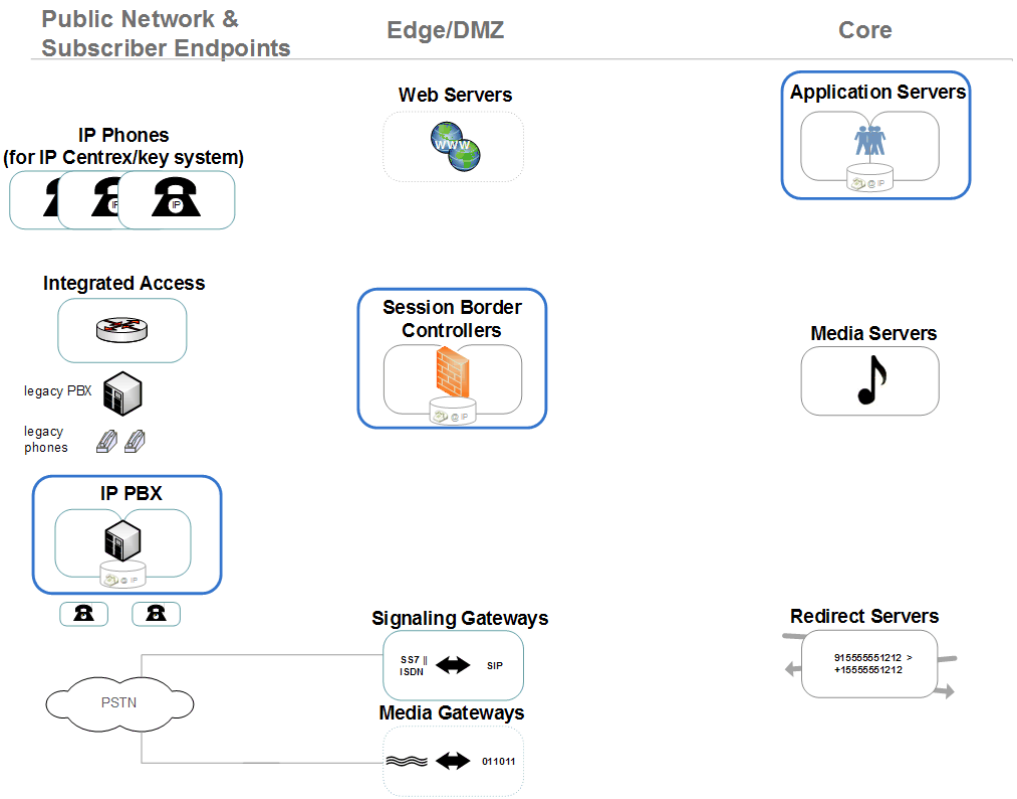
Gateways translate between SIP in the signalling plane and RTP on the media plane to, for instance, SS7 in the media plane and TDM voice bearers in the media plane. It's only the signalling gateway, shown above, that is a SIP UA. The media gateway is just a source/destination for media.

It's indicated as a media target by SIP messages (more on how that works shortly) but itself does not do SIP.

The next element is a back to back user agent (B2BUA). A back to back user agent always initiates a new dialogue on behalf of the client, so it's always acting as a user agent server and client. We'll get into more detail on the reasons why these are necessary and how they're used in the next section. At Leonid, we represent a B2BUA with a double rectangle.

The items with a blue rectangle around them in the below diagram of our topology are B2BUA's:

Figure 4 Topology with Back Back to Back UA's



Moving from left to right, the first functional example of a B2BUA is an IP PBX, which we represent with the icon below. An IP PBX functionally does the same thing as a digital PBX: it completes calls and implements features within its domain, and initiates calls to exterior networks through "trunk" interfaces. For outbound calls to the PSTN, the IP PBX receives SIP messages from the IP phones subtending on it and then initiates another set of messages (a dialogue) out to the exterior network to complete the call. This is the basic action of a B2BUA. The cylinder in the below middle is a Registrar, and we'll cover that in the next section.

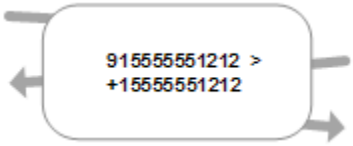
A session border controller (SBC) is also a B2BUA. A session border controller acts as an application-aware firewall, protecting the core elements and performing NAT traversal. It hides the internal network from the outside world by initiating separate dialogues to the access and internal/core networks. There's a later section dedicated to NAT traversal (skip ahead if you're curious) but suffice to say for the moment that SIP messages contain IP addresses that may not be routable at the destination and the SBC sorts that out.

The application server handles services. A full subscriber database resides there and when you, for instance, go on to the web to change your call forward number, you're changing it in the database on the application server. It decides where to forward to the call. And this is

why it's a B2BUA rather than a UA: it receives a request and then it may decide to respond to it any number of ways depending on the configured application logic for that number/subscriber.

A registrar receives registration requests (a SIP method you'll read about shortly) and records them in a database. The registrar is then contacted by other endpoints to reach subscribers at their registered addresses. The registrar is most frequently co-located in a the functional units: application server, session border controller, and IP PBX.

The last functional elements is a redirect server. It always receives a request and responds with a redirect message. Unlike all the elements we've reviewed to date, it is not call stateful. In other words, it responds to a query but it is not aware of individual calls/sessions between UA's. You ask it a question, it responds. After that it remembers nothing.



The last element is a Proxy. While proxies are not prevalent in most service provider networks, they are still the basic unit of analysis in most of the RFC's mentioned earlier here. A proxy is a network element in SIP that forwards requests on behalf of a UA. Unlike a B2BUA, it is only allowed to make very minor modifications to the SIP headers, which are prescribed by RFC 3261. A network element in a service provider network generally needs to hide topologies, change request parameters or even the basic call logic. This is why most network elements that are not UA's are B2BUA's. However, you will still see the term proxy mentioned frequently, often in functional terms. For instance, you might see a specification from a vendor that describes when their unit acts like a B2BUA and when it acts like a proxy, meaning that in some cases it is proxy-like, simply passing through messages.

How does SIP learn addresses?

An elemental question about any protocol is how you find your way around. In SIP there are three primary address elements you'll need to understand for starters:

1. SIP URI

   This is who you are in SIP, regardless of where you are. It's kind of like an email address. In fact, it looks like an email address: while a typical address in a commercial system is something like 4153696000@voipdomain.com, your SIP URI could be anything@anydomain.com.

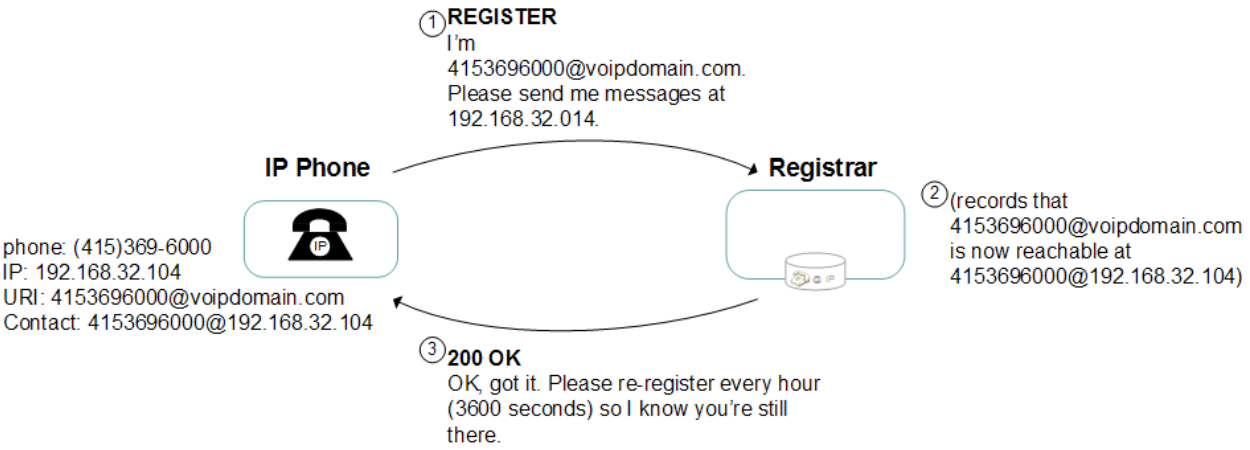
2. Contact Address

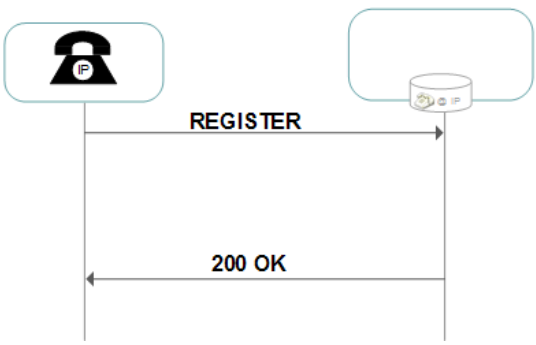
   This is who you are where you are. For example, if I register from my IP phone right now (we'll get into the registration thing in a minute) my contact address would be 4153696000@192.168.32.104. 192.168.32.104 is the address of my IP SIP phone.

3. Media Address

This is where I want to receive media, RTP packets with voice (or video or ....anything really). In my case, this is the same address for my IP phone, 192.168.32.104. It receives media and signalling on the same interface.

The essential address learning method in SIP is a REGISTER message. This is sent by a UA to a registrar to let the registrar know where it should be contacted. Below we see the basic action between UA and sip server (which might be, for instance, an application server or SBC; more on that shortly):





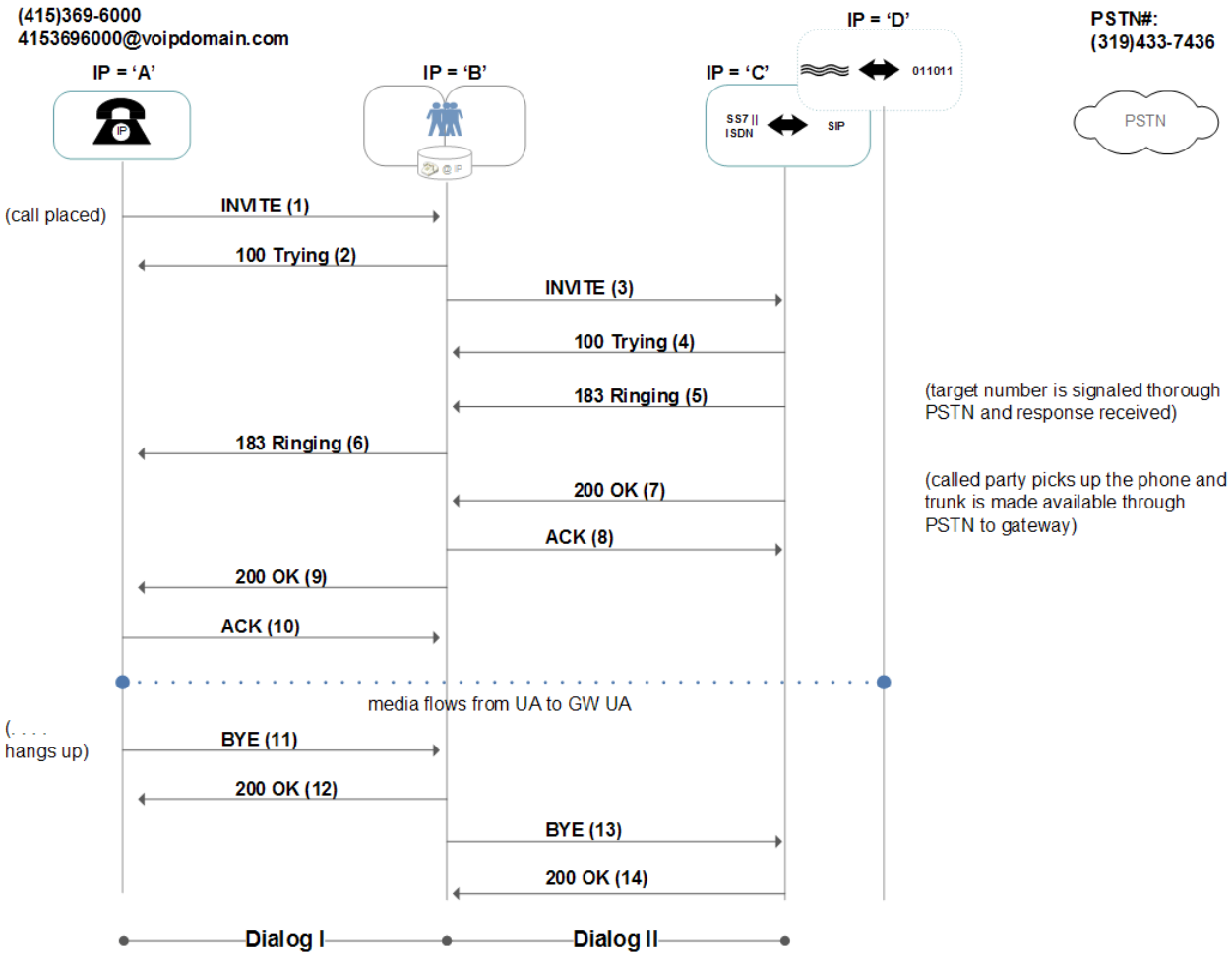
The SIP UA contacts it's registrar, the registrar responds with a positive response, recording the UA's contact IP address, as long as it can find the UA's SIP URI in its database of valid addresses. The call is show here in the form of a ladder diagram, which we'll be using going forward.

The media address comes in to play with the UA initiates or responds to a SIP session, which we'll review in the next section.

How does SIP negotiate sessions?

By way of review, the basic function of SIP is to set up media sessions over IP. SIP handles the set up of the media through a SIP signalling dialogue, and SDP, which is carried in SIP messages, describes the media session. Let's start with a basic example: a user with a SIP phone (415-369-6000) calls a PSTN number 319-433-7436. The legs of the call are the SIP UA/phone, then an application server, which in turn contacts a gateway on behalf of the user to translate the call out to the PSTN.

Figure 5 Basic SIP Session



Yes, we're jumping right in now!! The first thing to know about the message sequence above is the difference between methods and response. The INVITE and the BYE are methods, while the 183 and the 200 are responses. There are two SIP dialogs here since the application server is a B2BUA. Remember, a UA initiates and terminates individual SIP dialogs. A B2BUA will always initiate a new SIP dialog on behalf of the client. By message, here's a blow by blow:

1. The user picks up the phone, dials, and the UAC for 415-369-6000 sends out an INVITE.
2. The application server, which is a B2BUA responds with a conditional response, 100 Trying, to let the UAC know it got the message. The basic purpose of the 100 Trying is to let the UAC (remember, that's UA *client*, the requestor) know that it got the INVITE. Otherwise, the UAC would resend the INVITE. To date, SIP travels over UDP which is an unacknowledged transport protocol, so SIP has to provide its own message reliability.
3. The application server sends an INVITE out to the PSTN gateway, the signalling gateway, specifically. Why? Because this call needs to be sent out to the PSTN and the AS knows that the gateway is the place to make that happen. In the context of this INVITE, is the AS a UAC or UAS? If you answered client, you're right! It's making the request now, so it's the client. Notice the notation on the bottom of the diagram: there

are two dialogs here. The first is between the IP phone and the AS, the second is between the AS and the gateway. These are distinct SIP transactions where all the headers in the messages could be changed from dialog 1 to dialog 2. More on SIP headers after this. We'll call dialog 1 the access call leg and dialog 2 the network call leg.

4. The PSTN gateway responds to the AS with a 100 Trying conditional response. See message 2 explanation above for more context.

5. The PSTN gateway's PSTN/SS7 interface has been working to contact the called number and it's gotten back a message that the endpoint has been found and their phone is being rung! This gets translated by the gateway over to a SIP message called 183 (or 180) ringing, which is a conditional response, just like the 100 Trying we saw earlier.

6. The AS relays the good news over to the IP phone in the context of dialog 1.

7. Really good news: the called party has answered! Trunks have been cut through on the TDM side to bear the media and on the SIP side a 200 OK is issued by the gateway giving the final view on how the media should flow. A 200 OK is a successful/final response. The media session is described in this message by a MIME-like protocol called SDP which is carried in SIP messages. It has quite a few parameters but the three most typically important are: the IP address and UDP port for the media and the codec[4]. In this case the IP address and the port are on the media gateway (IP = 'D'). Key fact: the media socket described by the SDP does not have to be the same as the addresses involved in the SIP transactions. Why? One of the good features of SIP is that it allows you to separate media from signalling.

8. The AS sends an ACK back against the 200 OK. This is a SIP method that in this context means "got the go ahead and I'm ready to go as well".

9. The AS relays the good news over to the IP phone in the context of dialog 1. At this point, RTP is flowing and the callers can hear each other. The media socket on the access side happens to be the same as the signalling interface (it's just an IP phone with one IP address).

10. The UA sends back an ACK against the 200 OK in the context of dialog 1.

11. Whatever these two had to discuss is complete and the calling party (originator) hangs up. Now we see a whole new SIP method, a BYE, going from the IP phone out to the AS to take down the call.

12. The AS responds with a successful/final 200 OK.

13. The AS relays the news of their parting to the gateway in the context of dialog 2.

[4] Digital codecs are different ways of encoding analog wave forms, like your voice, into digital representations. Just Google "voice codecs" and you'll find some great explanations.

14. The gateway responds with a successful/final 200 OK.

Below is a quick summary of common SIP methods and what they do.

Table 2 SIP Methods

Name	Description
REGISTER	As we saw before, this is the message a UA sends when it wants to tell its registrar its IP address (so that it can be contacted)
INVITE	As you can see above, this is the basic call setup message the UA uses to initiate a SIP session.
CANCEL	This is used to cancel an INVITE before the call is set up. For instance, if our caller above changed his mind about the call before we got the 200 OK response and the call was set up, he would have sent a CANCEL rather than a BYE.
BYE	Once a call is set up, this is what you send to end the session (after you hang up)
OPTIONS	This is a message to query the capabilities of an endpoint. Message for message, you're most likely to see it used as a kind of SIP keepalive message. For example, an SBC will periodically send OPTIONS messages to an application server to make sure it's up. An application server might do the same thing to a gateway.
SUBSCRIBE & NOTFIY	This is a framework created for SIP to allow for event notification. Message waiting indicator (a light on your phone that illuminates when you have a message) is an example. Also, if you want to monitor the phone lines of other users, for instance on an attendant phone, you would use this.

Below is a quick summary of SIP response codes by series and what they do. These are sent in response to the SIP methods above.

Table 3 SIP Response Codes

Name	Description
1xx	Provisional/informational response. The server is performing additional work to finalize the request. Typical examples are the 100 Trying you saw above after the INVITE and the 180/3 Ringing response.
2xx	Successful/Final response. The request has been accepted and processed. The most typical example is the 200 OK you saw, for instance, when the call was connected on the TDM side and the gateway responded on its IP/SIP interface to set up IP media.
3xx	Redirect. The most typical example is a 302 redirect where a redirect server provides an alternate contact to a UAC from its database.
4xx	Client error. It was something you said: something about a client request is not right with the server. One common

	example is a 404 not found. You'll get this if you don't have the URI's provisioned exactly the same on the IP customer premise equipment (phones, etc.) and the servers (application server, gateways, etc.).
5xx	Server failure. The server can't process a request that otherwise seems valid to it. It's kind of like the server admitting it's got things wrong. If a gateway doesn't have a trunk available on a route, it might return a 503 service unavailable.
6xx	Global failure. It's not as bad as it sounds, but it's not great either. It means there's something generally wrong with the call flow. A gateway or SBC might send a 600 Busy Everywhere if it detects a call loop.

If that wasn't tedious enough, now we're going to go through each SIP message individually. Kidding! What we'll do is go through a few key items on a few messages and then the rest are in here for your reference. What I'd encourage you to do when you're done is do a test calls, retrieve logs off your own equipment, and make sure you can make sense of the call flow.

Below are all the messages for the service flow above with notes and annotations. *All the IP addresses have been replaced with the annotation from the diagram with representative letters (SIP phone = A; AS = B; Signalling GW = C; Media GW = D) in bold type.*

Message1:

- the SIP body has name:value pairs separated by colons for each header
- notice the request URI (first line)
    - this is the destination of the request
- notice the From, To, and Contact
    - these are the basic routing fields in SIP
    - they will remain the same for the balance of the dialog, regardless of the source/destination of the message
    - the from and the to specify the SIP URI's of the calling and called parties
    - the contact specifies the address at which the device will listen for messages
- the SDP body has name=value pairs separated by equal signs
- notice the media socket and list of preferred codec's
    - the IP address is on the o line and is 'A'; the RTP port is on the m line and is 51734
    - the preferred codec's are on the a lines and start with PCMU

INVITE sip:3194337436@voipdomain.com SIP/2.0
Via: SIP/2.0/UDP **A**:5060;branch=z9hG4bKvgdqan101g2h8bcmn440.1
From: Mister Tester <sip:4153696000@voipdomain.com>;tag=SDd30cb01-3057875635
To: <sip:3194337436@voipdomain.com>
Contact: <sip:4153696000@A:5060;transport=udp>
Call-ID: SDd30cb01-0ff359ae5db152c76a88570aba53adb3-vrvvfv3
CSeq: 58167 INVITE
Max-Forwards: 69
Content-Type: application/sdp
User-Agent: X-Lite release 1103m
Content-Length: 303
Route: <sip:**A**:5060;lr>

v=0
o=4153696000 410585765 410585812 IN IP4 **A**
s=X-Lite
c=IN IP4 **A**
t=0 0
m=audio 51734 RTP/AVP 0 8 3 98 97 101
a=rtpmap:0 pcmu/8000
a=rtpmap:8 pcma/8000
a=rtpmap:3 gsm/8000
a=rtpmap:98 iLBC/8000
a=rtpmap:97 speex/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15

Message2:

SIP/2.0 100 Trying
Via:SIP/2.0/UDP **A**:5060;branch=z9hG4bKvgdqan101g2h8bcmn440.1
From:"Mister Tester"<sip:4153696000@voipdomain.com>;tag=SDd30cb01-3057875635
To:<sip:3194337436@voipdomain.com>
Call-ID:SDd30cb01-0ff359ae5db152c76a88570aba53adb3-vrvvfv3
CSeq:58167 INVITE
Content-Length:0

Message3:

INVITE sip:13194337436@C:5060;user=phone;transport=udp SIP/2.0
Via:SIP/2.0/UDP **A**;branch=z9hG4bK-BroadWorks.A-CV5060-0-593706313-337295797-1184450902673-
From:"Mister Tester"<sip:4153696000@A;user=phone>;tag=337295797-1184450902673-
To:<sip:13194337436@C:5060;user=phone>
Call-ID:BW180822672140707-1869724241@**A**
CSeq:593706313 INVITE
Contact:<sip:**A**:5060>

Remote-Party-ID:"Mister Tester"<sip:4153696000@**A**;user=phone>;screen=yes;party=calling;privacy=off;id-type=subscriber
RPID-Privacy:party=calling;id-type=subscriber;privacy=off
Proxy-Require:privacy
Allow:ACK,BYE,CANCEL,INFO,INVITE,OPTIONS,PRACK,REFER,UPDATE,NOTIFY
Supported:100rel
Accept:multipart/mixed,application/sdp
Max-Forwards:10
Content-Type:application/sdp
Content-Length:286

v=0
o=BroadWorks 42916 1 IN IP4 68.239.81.114
s=-
c=IN IP4 A
t=0 0
m=audio 51734 RTP/AVP 0 8 3 98 97 101
a=rtpmap:0 pcmu/8000
a=rtpmap:8 pcma/8000
a=rtpmap:3 gsm/8000
a=rtpmap:98 iLBC/8000
a=rtpmap:97 speex/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15

Message4:

SIP/2.0 100 Trying
Via: SIP/2.0/UDP **A**;branch=z9hG4bK-BroadWorks.A-CV5060-0-593706313-337295797-1184450902673-
From: "Mister Tester"<sip:4153696000@**A**;user=phone>;tag=337295797-1184450902673-
To: <sip:13194337436@**C**:5060;user=phone>;tag=102116B0-11EE
Date: Sat, 14 Jul 2007 22:08:22 gmt
Call-ID: BW180822672140707-1869724241@**A**
Server: Cisco-SIPGateway/IOS-12.x
CSeq: 593706313 INVITE
Allow-Events: telephone-event
Content-Length: 0

Message5:

- notice here that the SDP has the address of D, the media GW, rather than the signalling GW, C

SIP/2.0 183 Session Progress
Via: SIP/2.0/UDP A;branch=z9hG4bK-BroadWorks.**A**-CV5060-0-593706313-337295797-1184450902673-
From: "Mister Tester"<sip:4153696000@**A**;user=phone>;tag=337295797-1184450902673-
To: <sip:13194337436@C:5060;user=phone>;tag=102116B0-11EE
Date: Sat, 14 Jul 2007 22:08:22 gmt

Call-ID: BW180822672140707-1869724241@**A**
Server: Cisco-SIPGateway/IOS-12.x
CSeq: 593706313 INVITE
Require: 100rel
RSeq: 8222
Allow: UPDATE
Allow-Events: telephone-event
Contact: <sip:13194337436@C:5060>
MIME-Version: 1.0
Content-Type: multipart/mixed;boundary=uniqueBoundary
Content-Length: 431

--uniqueBoundary
Content-Type: application/sdp

v=0
o=CiscoSystemsSIP-GW-UserAgent 6782 9466 IN IP4 **D**
s=SIP Call
c=IN IP4 **D**
t=0 0
m=audio 17354 RTP/AVP 0 101
c=IN IP4 C
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
--uniqueBoundary
Content-Type: application/gtd
Content-Disposition: signal;handling=optional

Message6:

SIP/2.0 183 Session Progress
Via:SIP/2.0/UDP **A**:5060;branch=z9hG4bKvgdqan101g2h8bcmn440.1
From:"Mister Tester"<sip:4153696000@voipdomain.com>;tag=SDd30cb01-3057875635
To:<sip:3194337436@voipdomain.com>;tag=1997287149-1184450905416
Call-ID:SDd30cb01-0ff359ae5db152c76a88570aba53adb3-vrvvfv3
CSeq:58167 INVITE
Allow:ACK,BYE,CANCEL,INFO,INVITE,OPTIONS,PRACK,REFER,UPDATE,NOTIFY
Supported:100rel
Contact:<sip:A:5060>
Session: Media
Remote-Party-ID:<sip:3194337436@**A**;user=phone>;screen=yes;party=called;privacy=off;id-type=subscriber
Content-Type:application/sdp
Content-Length:208

v=0
o=BroadWorks 42918 1 IN IP4 **D**
s=-
c=IN IP4 **D**

t=0 0
m=audio 17354 RTP/AVP 0 101
c=IN IP4 **C**
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15

Message7:

SIP/2.0 200 OK
Via: SIP/2.0/UDP **A**;branch=z9hG4bK-BroadWorks.**A**-CV5060-0-593706313-337295797-1184450902673-
From: "Mister Tester"<sip:4153696000@**A**;user=phone>;tag=337295797-1184450902673-
To: <sip:13194337436@C:5060;user=phone>;tag=102116B0-11EE
Date: Sat, 14 Jul 2007 22:08:25 gmt
Call-ID: BW180822672140707-1869724241@**A**
Server: Cisco-SIPGateway/IOS-12.x
CSeq: 593706313 INVITE
Allow: INVITE, OPTIONS, BYE, CANCEL, ACK, PRACK, COMET, REFER, SUBSCRIBE, NOTIFY, INFO, UPDATE, REGISTER
Allow-Events: telephone-event
Contact: <sip:13194337436@C:5060>
MIME-Version: 1.0
Content-Type: multipart/mixed;boundary=uniqueBoundary
Content-Length: 431

--uniqueBoundary
Content-Type: application/sdp

v=0
o=CiscoSystemsSIP-GW-UserAgent 6782 9466 IN IP4 **D**
s=SIP Call
c=IN IP4 **D**
t=0 0
m=audio 17354 RTP/AVP 0 101
c=IN IP4 C
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
--uniqueBoundary
Content-Type: application/gtd
Content-Disposition: signal;handling=optional

Message8:

ACK sip:A:5060 SIP/2.0
Via: SIP/2.0/UDP **A**:5060;branch=z9hG4bK2pk7ng301g11ab4js6c1.1
From: "Mister Tester"<sip:4153696000@voipdomain.com>;tag=SDd30cb01-3057875635
To: <sip:3194337436@voipdomain.com>;tag=1997287149-1184450905416

Contact: <sip:4153696000-l3qleqjuh0jff@**A**:5060;transport=udp>
Call-ID: SDd30cb01-0ff359ae5db152c76a88570aba53adb3-vrvvfv3
CSeq: 58167 ACK
Max-Forwards: 69
Content-Length: 0

Message9:

- notice here that on the network side, both dialog 1 and dialog 2 have the SDP of 'D', the media gateway
- B2BUA's may or may not provide media firewalling
    - if there was media firewalling, the media would go from UAC to B2BUA to UAS
    - in general, application servers do not provide this capabilities but SBC's do

SIP/2.0 200 OK
Via:SIP/2.0/UDP **A**:5060;branch=z9hG4bKvgdqan101g2h8bcmn440.1
From:"Mister Tester"<sip:4153696000@voipdomain.com>;tag=SDd30cb01-3057875635
To:<sip:3194337436@voipdomain.com>;tag=1997287149-1184450905416
Call-ID:SDd30cb01-0ff359ae5db152c76a88570aba53adb3-vrvvfv3
CSeq:58167 INVITE
Allow:ACK,BYE,CANCEL,INFO,INVITE,OPTIONS,PRACK,REFER,UPDATE,NOTIFY
Supported:100rel
Accept:multipart/mixed,application/sdp
Contact:<sip:A:5060>
Remote-Party-ID:<sip:3194337436@**A**;user=phone>;screen=yes;party=called;privacy=off;id-type=subscriber
Content-Type:application/sdp
Content-Length:208

v=0
o=BroadWorks 42918 1 IN IP4 **D**
s=-
c=IN IP4 **D**
t=0 0
m=audio 17354 RTP/AVP 0 101
c=IN IP4 C
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15

Message10:

ACK sip:A:5060 SIP/2.0
Via: SIP/2.0/UDP **A**:5060;branch=z9hG4bK2pk7ng301g11ab4js6c1.1
From: "Mister Tester"<sip:4153696000@voipdomain.com>;tag=SDd30cb01-3057875635
To: <sip:3194337436@voipdomain.com>;tag=1997287149-1184450905416
Contact: <sip:4153696000-l3qleqjuh0jff@**A**:5060;transport=udp>
Call-ID: SDd30cb01-0ff359ae5db152c76a88570aba53adb3-vrvvfv3
CSeq: 58167 ACK

vMax-Forwards: 69
Content-Length: 0

Message11:

BYE sip:A:5060 SIP/2.0
Via: SIP/2.0/UDP A:5060;branch=z9hG4bK4p1cr500eg516bcm16k1.1
From: "Mister Tester"<sip:4153696000@voipdomain.com>;tag=SDd30cb01-3057875635
To: <sip:3194337436@voipdomain.com>;tag=1997287149-1184450905416
Contact: <sip:4153696000-l3qleqjuh0jff@A:5060;transport=udp>
Call-ID: SDd30cb01-0ff359ae5db152c76a88570aba53adb3-vrvvfv3
CSeq: 58168 BYE
Max-Forwards: 69
User-Agent: X-Lite release 1103m
Content-Length: 0

Message12:

SIP/2.0 200 OK
Via:SIP/2.0/UDP **A**:5060;branch=z9hG4bK4p1cr500eg516bcm16k1.1
From:"Mister Tester"<sip:4153696000@voipdomain.com>;tag=SDd30cb01-3057875635
To:<sip:3194337436@voipdomain.com>;tag=1997287149-1184450905416
Call-ID:SDd30cb01-0ff359ae5db152c76a88570aba53adb3-vrvvfv3
CSeq:58168 BYE
Content-Length:0

Message13:

BYE sip:13194337436@**C**:5060 SIP/2.0
Via:SIP/2.0/UDP **A**;branch=z9hG4bK-BroadWorks.**A**-CV5060-0-593706315-337295797-1184450902673-
From:"Mister Tester"<sip:4153696000@**A**;user=phone>;tag=337295797-1184450902673-
To:<sip:13194337436@**C**:5060;user=phone>;tag=102116B0-11EE
Call-ID:BW180822672140707-1869724241@A
CSeq:593706315 BYE
Max-Forwards:10
Content-Length:0

Message14:

SIP/2.0 200 OK
Via: SIP/2.0/UDP **A**;branch=z9hG4bK-BroadWorks.**A**-CV5060-0-593706315-337295797-
1184450902673-
From: "Mister Tester"<sip:4153696000@**A**;user=phone>;tag=337295797-1184450902673-
To: <sip:13194337436@**C**:5060;user=phone>;tag=102116B0-11EE
Date: Sat, 14 Jul 2007 22:08:36 gmt
Call-ID: BW180822672140707-1869724241@A
Server: Cisco-SIPGateway/IOS-12.x
Content-Length: 0
CSeq: 593706315 BYE

On the inbound side things more or less work in reverse. There are more call flow examples in the next section, Networking with SIP.

How does SIP avoid loops?

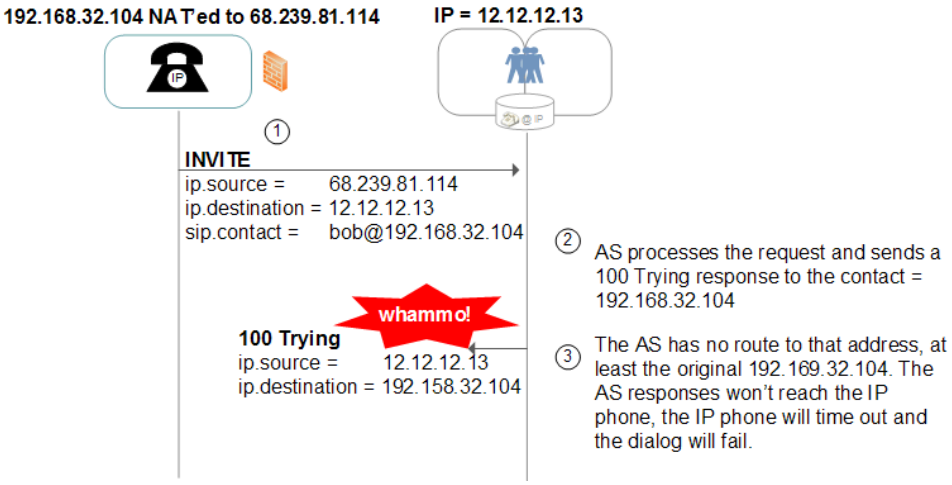
As of RFC 3261, the primary mechanism for loop avoidance is the Max-Forwards parameter, which you can see in the messages in the previous section. The standard starting point for this value is 70 and every UA/proxy in the decrements the value (though in well understood/closed topologies it can be set lower)[5]. If a SIP entity receives the request with a 0 value, it will return a 483 Too Many Hops message to the UAC. Additionally, through use of the Via header and hashes it writes into tags in the via header parameters (all those funny strings you see in the messages), if a proxy/UA receives a looped request it will return a 482 Loop Detected.

What is the NAT problem?

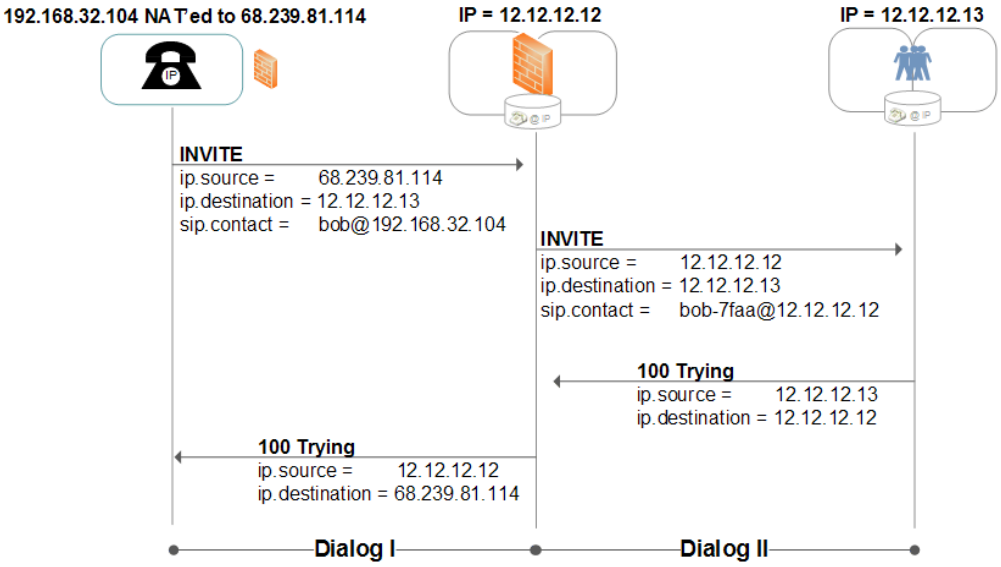
The first thing to understand about the NAT (network address translation) problem is the difference between private and public space. What follows is a brief explanation (skip to next paragraph if you know it). If the topic is new to you, I'd recommend flipping through an IP tutorial somewhere that includes the topic. Within the IP address space, there are public and private addresses. Public addresses are globally unique in the context of the whole Internet. Private addresses are not. Private addresses are allocated for reuse within private networks. If you have a home router, it probably gives you a private IP address for your computer. If you look at the IP of your computer at work, it's probably private. The private address ranges are: 10.0.0.0/8, 192.168.0.0/16, and 172.16.0.0/12. When you have a private IP address and your computer sends an IP packet to the outside world, the router marks the source address to its own. This is the action of NAT. Otherwise, that computer you want to talk to on the outside world would never know how to respond to you since you have a private IP and private IP's are not routeable on the Internet.

SIP uses IP addresses within its own messages, as we saw above. My IP phone had a public address, so no NAT problem (no NAT, ergo, no NAT problem). But suppose it didn't. Suppose I had a private address that was 192.168.32.104. And let's suppose I want to make that call we stepped through above. I send an INVITE to the AS, it's NAT'ed to the public IP of my router, let's say that's 68.239.81.114. The INVITE reaches the AS: no problem whatsoever. Does that AS have any problem processing the INVITE? No, it's just fine. But what's my contact address going to be? It's going to be 192.168.32.104, and that's where the AS is going to try to send me my first conditional response (100 Trying). That 100 Trying will never reach me, and that's because inside this entirely different network where the AS resides, that address is either meaningless or assigned to some totally different computer.

[5] If you're familiar with IP, you've probably noted already that the operation is very similar to the TTL parameter in an IP packet.



The current commercial solution to this problem is to use an SBC (session border controller). You'll also hear these referred to as application layer gateways (ALG's), and that's what it essentially does: provides NAT/gateway service at the application layer. The SBC sits on the public side of the network and listens for SIP requests. When it gets one, it not only inspects the IP header to route the packet, but it also looks inside the SIP message at the contact address and rewrites it to a routeable address before directing it to its next hop (usually an application server or another SBC on another network). Does routeable address mean public address? No. In fact the address could be private. The point is that the SBC will be configured to rewrite the contact to an address that's routeable by the next hop in the service flow. In this case below we see just such action:



Where do I learn more?

The table below summarizes some useful resources for further study and general reference.

Leonid Consulting

Table 4 Study and Reference Resources

Name (Author)	Description
SIP: Understanding the Session Initiation Protocol (Johnston, Alan)	Excellent as both a reference and guide for comprehensive study, this book by one of the authors of RFC 3261 describes core SIP fundamentals.
vendor-specific training (various)	Depending on what equipment you own, equipment vendors offer various courses of study. Both BroadSoft and Acme Packet are noted for having good classroom study. These will introduce some fundamentals and many practical deployment items. BroadSoft training materials are freely available to all BroadSoft customers.
IETF Website	At the type of your fingertips (just Google it), these are useful for specific issues, particularly where standards compliance is an issue.

Leonid Consulting

Appendix A: Exercises

The following are a few basic exercises for you to practice your skills. Many are general, though some are specific to the BroadWorks platform.

Exercise	Description
Call Transit on a Back to Back User Agent (multiple dialogs)	You'll need the logs from either an SBC or Application Server. Make a call. Find the call-ID's for both the access and network side call halves/SIP dialogues. What are they? Observe that you can quickly review the balance of a dialog after getting the call-ID using it to do a search in your browser.
Inbound Call to Voicemail (multiple access, network, dialogs)	An external/PSTN party calls a VoIP subscriber with voicemail. How many unique SIP dialogues to you see? What are the endpoints for in the control (SIP) and media (SDP) plane?
Call Hold (SDP updates)	An external party calls a VoIP subscriber. The VoIP subscriber puts the call on hold. What happens? What was it that put the call on hold?
Busy Lamps in BroadWorks (non-call SIP transactions)	Assign BLF, populate a BLF list, and provision a device. What do you see the device doing?
Bridged Lines (subscribe/notify)	Set up a bridged line appearance between two parties. What do you see happening when the party goes off hook to indicate status to the user sharing their line?