RADVISION

the V²oIP™ experts

SIP: Protocol Overview

NOTICE

http://www.radvision.com

CONTENTS

1

Session Initiation Protocol (SIP)

Introduction

The Session Initiation Protocol (SIP) is a signaling protocol for initiating, managing and terminating voice and video sessions across packet networks. SIP sessions involve one or more participants and can use unicast or multicast communication. Borrowing from ubiquitous Internet protocols, such as HTTP and SMTP, SIP is text-encoded and highly extensible. SIP may be extended to accommodate features and services such as call control services, mobility, interoperability with existing telephony systems, and more.

SIP is being developed by the SIP Working Group, within the Internet Engineering Task Force (IETF). The protocol is published as IETF RFC 2543 and currently has the status of a proposed standard.

This section describes the key constituents of SIP.

SIP Entities

A SIP network is composed of four types of logical SIP entities. Each entity has specific functions and participates in SIP communication as a client (initiates requests), as a server (responds to requests), or as both. One “physical device” can have the functionality of more than one logical SIP entity. For example, a network server working as a Proxy server can also function as a Registrar at the same time.

Following are the four types of logical SIP entities:

User Agent

In SIP, a **User Agent** (UA) is the endpoint entity. User Agents initiate and terminate sessions by exchanging requests and responses. RFC 2543 defines the User Agent as an application, which contains both a User Agent client and User Agent server, as follows:

- **User Agent Client (UAC)**—a client application that initiates SIP requests.
- **User Agent Server (UAS)**—a server application that contacts the user when a SIP request is received and that returns a response on behalf of the user.

Some of the devices that can have a UA function in a SIP network are: workstations, IP-phones, telephony gateways, call agents, automated answering services.

Proxy Server

A **Proxy Server** is an intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. Requests are serviced either internally or by passing them on, possibly after translation, to other servers. A Proxy interprets, and, if necessary, rewrites a request message before forwarding it.

Redirect Server

A **Redirect Server** is a server that accepts a SIP request, maps the SIP address of the called party into zero (if there is no known address) or more new addresses and returns them to the client. Unlike Proxy servers, Redirect Servers do not pass the request on to other servers.

Registrar

A **Registrar** is a server that accepts REGISTER requests for the purpose of updating a location database with the contact information of the user specified in the request.

Messages

Message Types

There are two types of SIP messages:

- Requests—sent from the client to the server.
- Responses—sent from the server to the client.

Requests

Table 1-1 *Request Methods*

Method	Description
INVITE	Initiates a call, changes call parameters (re-INVITE).
ACK	Confirms a final response for INVITE.
BYE	Terminates a call.
CANCEL	Cancels searches and “ringing”.
OPTIONS	Queries the capabilities of the other side.

Method	Description
REGISTER	Registers with the Location Service.
INFO	Sends mid-session information that does not modify the session state.

Responses

Response messages contain numeric response codes. The SIP response code set is partly based on HTTP response codes. There are two types of responses and six classes:

Response Types

- **Provisional** (1xx class)—provisional responses are used by the server to indicate progress, but they do not terminate SIP transactions
- **Final** (2xx, 3xx, 4xx, 5xx, 6xx classes)—final responses terminate SIP transactions.

Classes

- 1xx = provisional, searching, ringing, queuing etc.
- 2xx = success
- 3xx = redirection, forwarding
- 4xx = request failure (client mistakes)
- 5xx = server failures
- 6xx = global failure (busy, refusal, not available anywhere)

Table 1-2 *Response Code Examples*

100	Continue	408	Request time-out
180	Ringing	480	Unavailable
200	OK	481	Call-leg/Transaction does not exist
300	Multiple choices	482	Loop detected
301	Moved permanently	5xx	Server error
302	Moved temporarily	600	Busy
400	Bad request	603	Decline
401	Unauthorized	604	Does not exist
403	Forbidden	606	Not acceptable

Message Parts

SIP messages are composed of the following three parts:

Start Line

Every SIP message begins with a Start Line. The Start Line conveys the message type (method type in requests, and response code in responses) and the protocol version. The Start Line may be either a Request-line (requests) or a Status-line (responses), as follows:

- The Request-line includes a Request URI, which indicates the user or service to which this request is being addressed. Unlike the "To" field (see *Message Samples* below), this address can be re-written by proxies.
- The Status-line holds the numeric Status-code and its associated textual phrase.

Headers

SIP header fields are used to convey message attributes and to modify message meaning. They are similar in syntax and semantics to HTTP header fields (in fact some headers are borrowed from HTTP) and thus always take the format:

```
<name>:<value>
```

Headers can span multiple lines. Some SIP headers such as Via, Contact, Route and Request-Route can appear multiple times in a message or, alternatively, can take multiple comma-separated values in a single header occurrence.

Body (Content)

A message Body is used to describe the session to be initiated (for example, in a multimedia session this may include audio and video codec types, sampling rates etc.), or alternatively it may be used to contain opaque textual or binary data of any type which relates in some way to the session. Message bodies can appear both in request and in response messages. SIP makes a clear distinction between signaling information, conveyed in the SIP Start Line and headers, and the session description information, which is outside the scope of SIP.

Possible body types include:

- SDP—see *Session Description Protocol (SDP)*.
- Multipurpose Internet Mail Extensions (MIME).
- Others—to be defined in the IETF and in specific implementations.

Message Samples

The following samples show the message exchange between two User Agents for the purpose of setting up a voice call. SIP user alice@radvision.com invites SIP user bob@acme.com to a call for the purpose of discussing lunch. Alice sends an INVITE request containing an SDP body. Bob replies with a 200 OK response also containing an SDP body.

Request Message

Request Message line	Description
INVITE sip:bob@acme.com SIP/2.0	Request line: Method type, request URI (SIP address of called party), SIP version.
Via: SIP/2.0/UDP	Address of previous hop.
alice_ws.radvision.com	
From: Alice A. <sip:alice@radvision.com>	User originating this request.
To: Bob B. <sip:bob@acme.com>	User being invited, as specified originally.
Call-ID: 2388990012@alice_ws.radvision.com	Globally unique ID of this call.
CSeq: 1 INVITE	Command sequence. Identifies transaction.
Subject: Lunch today.	Call subject and/or nature.
Content-Type: application/SDP	Type of body—in this case SDP.
Content-Length: 182	Number of bytes in the body.
	Blank line marks end of SIP headers and beginning of body.
v=0	Version of SDP.

Request Message line	Description
o=Alice 53655765 2353687637 IN IP4 128.3.4.5	Owner/creator and session identifier, session version address type and address.
s=Call from Alice.	Session subject.
c=IN IP4 alice_ws.radvision.com	Connection information.
M=audio 3456 RTP/AVP 0 3 4 5	Media description: type, port, possible formats caller is willing to receive and send.

RESPONSE MESSAGE

Response Message line	Description
SIP/2.0 200 OK	Status line: SIP version, response code, reason phrase.
Via: SIP/2.0/UDP alice_ws.radvision.com	Copied from request.
From: Alice A. <sip:alice@radvision.com>	Copied from request.
To: Bob B. <sip:bob@acme.com>;tag=17462311	Copied from request. Includes unique tag to identify call-leg.
Call-ID: 2388990012@alice_ws.radvision.com	Copied from request.
CSeq: 1 INVITE	Copied from request.
Content-Type: application/SDP	
Content-Length: 200	
	Blank line marks end of SIP headers and beginning of the body.
v=0	Version of SDP.
o=Bob 4858949 4858949 IN IP4 192.1.2.3	Owner/creator and session identifier, session version address type and address.
s=Lunch	Session subject.
c=IN IP4 machine1.acme.com	Connection information.
m=audio 5004 RTP/AVP 0 3	Description of media streams the receiver of the call is willing to accept.

Entity Interaction

This section describes the interaction between SIP entities in various common session initiation scenarios.

Session Establishment and Termination

Figure 1-1 shows the interaction between a User Agent Client (UAC) and a User Agent Server (UAS) during trivial session establishment and termination.

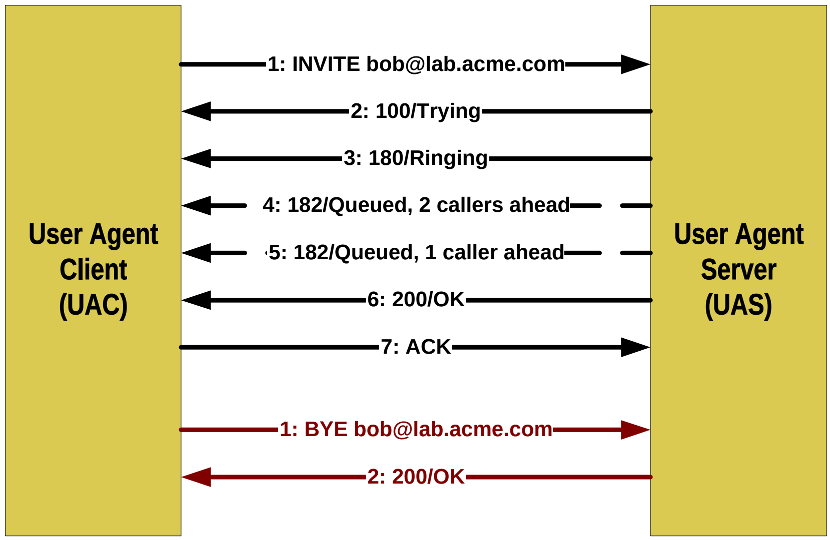

Figure 1-1 *SIP Session Establishment and Call Termination*

Session Establishment

Call Flow

1. The calling User Agent Client sends an INVITE message to Bob's SIP address: sip:bob@acme.com. This message also contains an SDP packet describing the media capabilities of the calling terminal.
2. The UAS receives the request and immediately responds with a 100-Trying response message.
3. The UAS starts "ringing" to inform Bob of the new call. Simultaneously a 180 (Ringing) message is sent to the UAC.
4. The UAS sends a 182 (Queued) call status message to report that the call is behind two other calls in the queue.
5. The UAS sends a 182 (Queued) call status message to report that the call is behind one other call in the queue.

6. Bob picks up the call and the UAS sends a 200 (OK) message to the calling UA. This message also contains an SDP packet describing the media capabilities of Bob's terminal.
7. The calling UAC sends an ACK request to confirm the 200 (OK) response was received.

Session Termination

The session termination call flow proceeds as follows:

1. The caller decides to end the call and "hangs-up". This results in a BYE request being sent to Bob's UAS at SIP address sip:bob@lab.acme.com
2. Bob's UAS responds with 200 (OK) message and notifies Bob that the conversation has ended.

Call Redirection

Figure 1-2 shows a simple call redirection scenario.

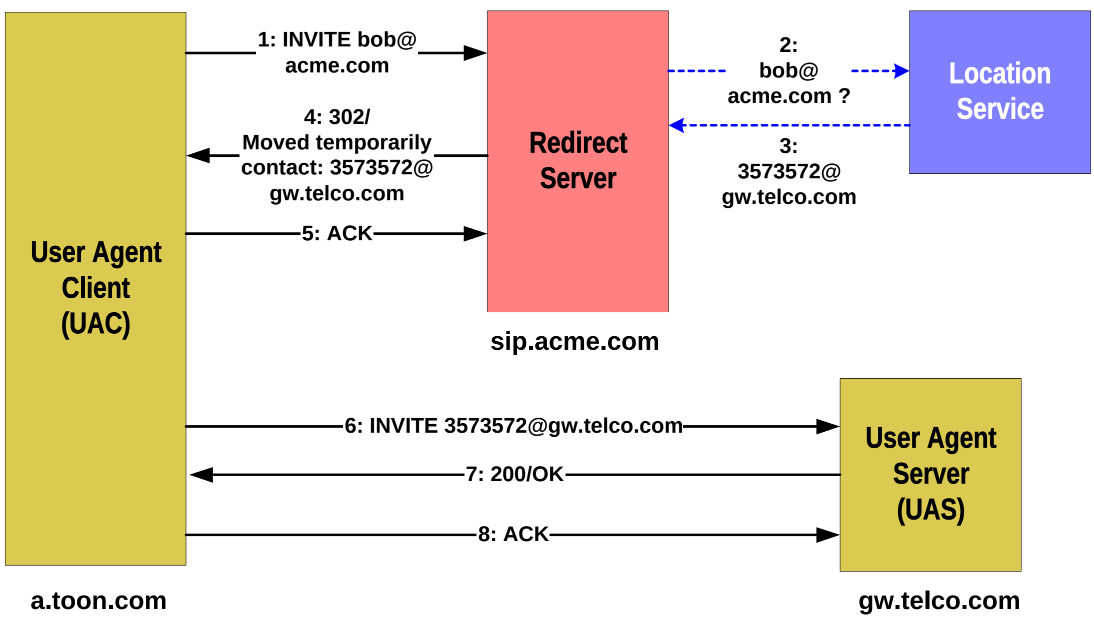

Figure 1-2 *Simple Call Redirection Using a Redirect Server*

CALL FLOW

1. First a SIP INVITE message is sent to bob@acme.com, but finds the Redirect server sip.acme.com along the signaling path.
2. The Redirect server looks up Bob's current location in a Location Service using a non-SIP protocol (for example, LDAP).
3. The Location Service returns Bob's current location: SIP address 3573572@gwtelco.com.
4. The Redirect Server returns this information to the calling UAC using a 302 (Moved Temporarily) response. In the response message it enters a contact header and sets the value to Bob's current location, 3573572@gwtelco.com.
5. The calling UAC acknowledges the response by sending an ACK message.
6. The calling UAC then continues the transaction directly with gw.telco.com by sending a new INVITE.
7. gw.telco.com is able to notify Bob's terminal of the call and Bob "picks up" the call. A 200 (OK) response is sent back to the calling UAC.
8. The calling UAC acknowledges with an ACK message.

CALL PROXYING

Figure 1-3 shows call set-up between two User Agents with the assistance of an intermediate Proxy server.

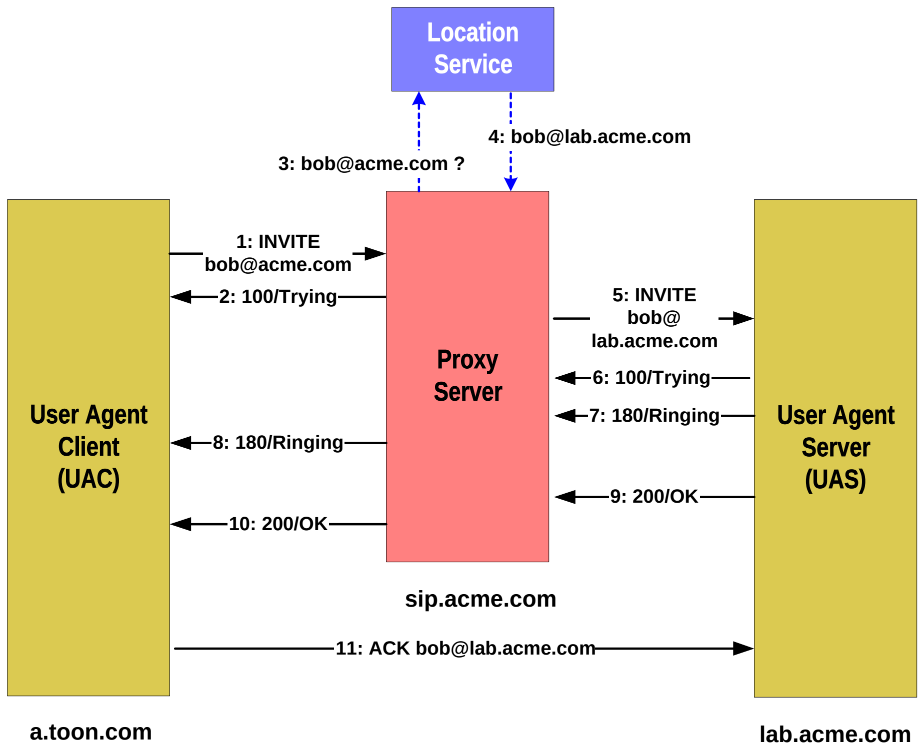

Figure 1-3 *Call Proxying Scenario*

CALL FLOW

1. An INVITE message is sent to bob@ acme.com, but finds the Proxy server sip.acme.com along the signaling path.
2. The Proxy server immediately responds with a 100 (Trying) provisional response.
3. The Proxy server looks-up Bob's current location in a Location Service using a non-SIP protocol (For example, LDAP).
4. The Location Service returns Bob's current location: SIP address bob@lab.acme.com.

5. The Proxy server decides to proxy the call and creates a new INVITE message based on the original INVITE message, but with the request URI in the start line changed to bob@lab.acme.com. The Proxy server sends this request to the UAS at lab.acme.com.
6. The UAS responds first with a 100 (Trying).
7. The UAS responds with a 180 (Ringing) response.
8. The Proxy server forwards the 180 (Ringing) response back to the calling UA.
9. When the call is accepted by the user (for example, by picking up the handset) the UAS at lab.acme.com sends a 200 (OK) response. In this example, Bob's UAS inserts a Contact header into the response with the value bob@lab.acme.com. Further SIP communication will be sent directly to it and not via the Proxy Server. This action is optional.
10. The Proxy forwards the 200 (OK) response back to the calling UAC.
11. The calling UA sends an ACK directly to Bob's UA at the lab (according to the Contact header it found in the 200 (OK) response).

Session Description Protocol (SDP)

SDP is the protocol used to describe multimedia session announcement, multimedia session invitation and other forms of multimedia session initiation. A multimedia session is defined, for these purposes, as a set of media streams that exist for a duration of time.

SDP Packets

SDP packets usually include the following information:

Session information

- Session name and purpose.
- Time(s) the session is active.

Since the resources necessary for participating in a session may be limited, it would be useful to include the following additional information:

- Information about the bandwidth to be used by the session.
- Contact information for the person responsible for the session.

Media information

- Type of media, such as video and audio.
- Transport protocol, such as RTP/UDP/IP and H.320
- Media format, such as H.261 video and MPEG video.

- Multicast address and Transport Port for media (IP multicast session).
- Remote address for media and Transport port for contact address (IP unicast session).