APPLICATION ARTICLE

SIP Out Sample App Article

OVERVIEW

The SIP Out Sample Application is designed to illustrate how easy it is to use the SIP control portion of the RAPID API to make audio and video connections between a WebRTC Client and a SIP Service Network or Trunk.

Contact Sansay for a copy of the companion Sample Application source code that this article is based on.

SOLUTION DIAGRAM

Figure 1 illustrates the topology of the SIP Out Sample Application.

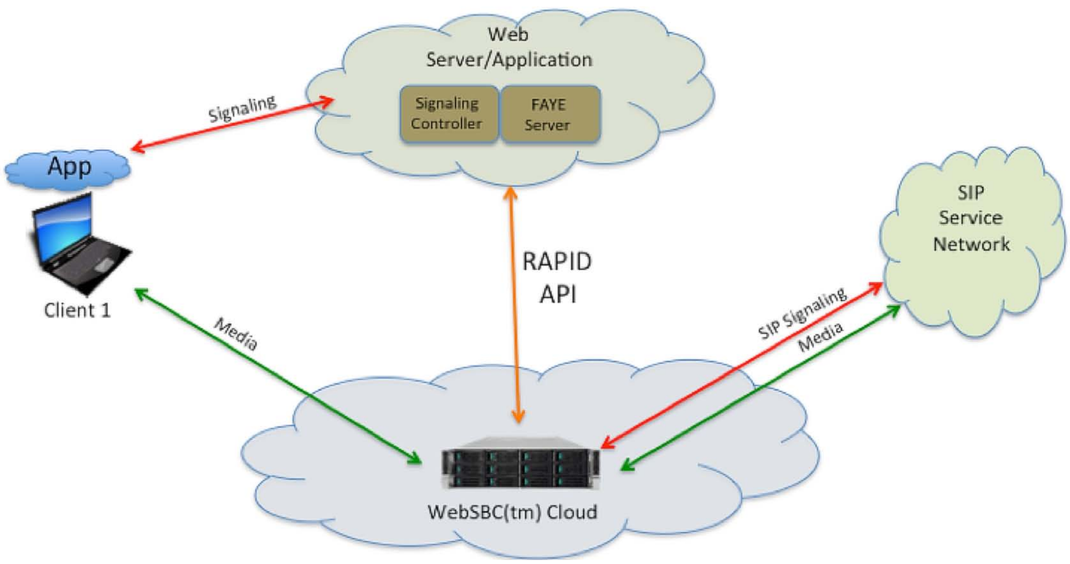

Figure 1

The Web Server and Application are based on a Ruby on Rails platform. The primary Application operation is contained in the Signaling Controller software module (/sansay_examples/app/controllers/signaling_controller.rb). FAYE is used for Websockets support and as an event routing channel between the Signaling Controller and the Client application instances.

OPERATION

There are 2 primary operations provided in the sample code:

1) Registration
2) Connection Establishment
3) Connection Tear Down

ROAP is used as the Session Establishment protocol between the Client and Server Applications. The Sansay RAPID API Document provides a complete description of the API methods supported by the WebSBC.

REGISTRATION

The Registration procedure (Figure 2) is standard HTTP. The Browser requests a page load from the Web Server. The Web Server responds by sending the web page with the WebRTC client software to back to the Browser. The WebRTC Client Application then opens a Websocket connection with the FAYE Server Module.

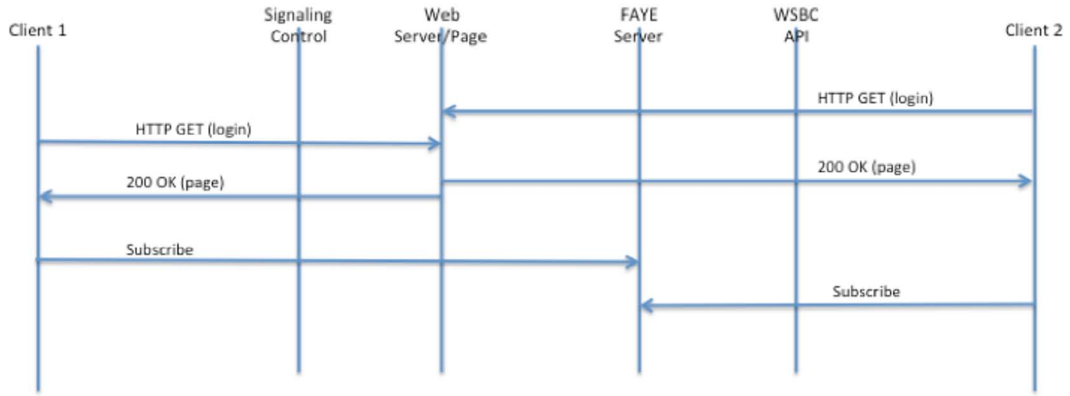

Figure 2

In an actual Application, the initial HTTP Request would likely carry Login information that would be used to identify the Subscriber to the Application. The Application could then maintain a Registration Database and validate the Registration Request. The Sample Code contains place holders for this operation but does not implement authentication

CONNECTION ESTABLISHMENT

Once registered, the Client Application can now initiate a connection with a remote Client. This operation is shown in Figure 3.

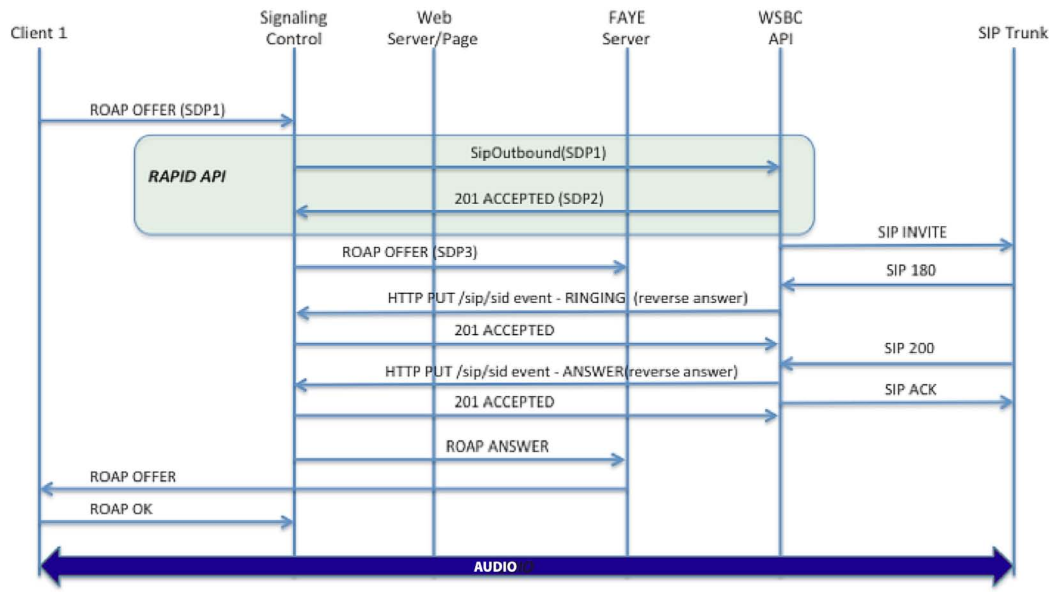

Figure 3

The WebSBC SIP API is used in this Sample. This procedure is covered in the SIP section of the API Document. The RAPID API methods used are highlighted in Figure 3 above. The SipOutbound method passes the SDP of the Client requesting the connection. The WebSBC then initiates a SIP connection establishment procedure on behalf of the requesting Application. In this case the Application is represented by the Signaling Control module in the Sample Application. Call progress events are passed to the Signaling Control Application through HTTP PUT event methods as shown. Once an ANSWER event is received by the Signaling Control Application the originating Client is notified that the connection completed through the FAYE server.

CONNECTION TEAR DOWN

Figure 4 contains 2 procedures that are important. The first is the Session Refresh operation. The Sample Application specifies a Session TTL (time to live) of 10 seconds. The Application must issue a SipSessionRefresh, with the Session ID of the connection to refresh, within the TTL time interval to keep the session alive within the WebSBC.

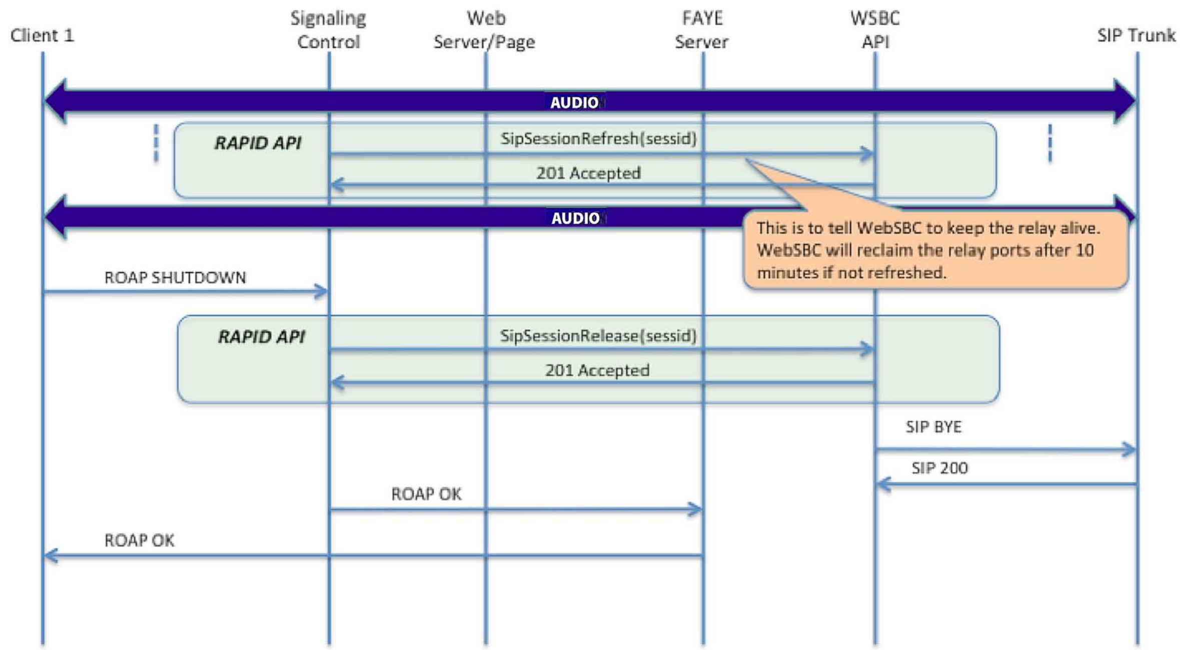

Figure 4

The second procedure in Figure 4 is the Connection Tear Down. Client 1 "hangs up" the connection and issues a SHUT-DOWN command over the ROAP channel. This causes the Signaling Control Module to issue the SipSessionRelease command to the WebSBC to bring down the Connection Session. The WebSBC responds with a 200 ACCEPTED when the operation is complete.

Corporate Headquarters:
Sansay, Inc.
4350 La Jolla Village Dr. Ste. 888
San Diego, CA 92122

Toll Free: +1 888.889.8906
Office: +1 858.754.2200
Fax: +1 858.550.2044

www.sansay.com