



RTCWeb 及其与 IMS 的融合研究

乐利锋, 彭 晋, 段晓东

(中国移动通信研究院 北京 100053)

摘 要: RTCWeb 是一种实现免插件 Web 客户端及通信的技术。相比传统应用客户端, RTCWeb 客户端操作更加便捷, 开发更加灵活, 并且能脱离操作系统和平台的限制, 其对于业务开发者、网络运营商和终端用户都是一种更好的选择。本文首先对 RTCWeb 通信模型及关键技术进行了介绍; 然后针对 RTCWeb 和 IMS 融合提出了构想并给出此构想下的具体解决方案; 最后展望了基于 IMS 的 RTCWeb 系统的未来演进趋势。

关键词: IMS; Web 客户端; 实时通信; 免插件

doi: 10.3969/j.issn.1000-0801.2013.01.003

Research on RTCWeb and Its Merging with IMS

Le Lifeng, Peng Jin, Duan Xiaodong

(China Mobile Research Institute, Beijing 100053, China)

Abstract: RTCWeb is a key technology to realize free plug-in web client. Compared with traditional client, RTCWeb client enables more convenient operation, more flexible development, and detachment from the operating system and platform. It is a better choice for service developers, network operators and terminal users. This paper firstly introduced RTCWeb's communication model and key technologies, then proposed the conception of RTCWeb's merging with IMS and gave the specific solutions under the conception, finally gave a summery and a prospect for future evolution of IMS based RTCWeb system.

Key words: IMS, web client, real-time communication, free plug-in

1 引言

Web 应用已成为互联网应用最成功的模式之一, 其应用范畴已经渗透人类社会生活的各个领域。人们只要通过一款浏览器, 就可以享受到无穷无尽的 Web 服务, 如各种电子商务、网络游戏、社区服务及各种网络交友等。与此同时, Web 技术所带来的优势 (如统一的客户端和较好的可维护性), 使一些传统应用纷纷转型到 Web 模式。Web 应用开发的简单快速也受到开发者青睐, 开发者只需要一次开发就可以在任何地方部署, 代码片段很容易在开发者之间被复制和粘贴, 越来越容易掌握的 JavaScript 库使开发更加快捷。

Web 浏览器一直随着 Web 应用的需要持续加入新的

Web 特性, W3C(world wide web consortium, 万维网联盟) 标准化组织追求的 Open Web Platform^[1]是所有 Web 技术的集合, 不仅包括基础 Web 技术, 如 HTML (hypertext markup language, 超文本标记语言)、CSS (cascading style sheet, 级联样式表) 和 JavaScript 及 HTTP (hypertext transfer protocol, 超文本传输协议) 等; 还包括新的 Web 技术, 如 HTML5^[2], 允许开发者免费使用它所发布的技术。可以预见, 未来 Web 将逐渐成为一种准入门槛较低、推广成本可控的统一应用平台。

基于 Web 的实时通信应用可以吸引越来越多的用户。Facebook^[3]推出了整合 Skype 的视频聊天服务, 首次使用该功能的用户需要安装插件; GoogleTalk^[4]网页版的正常使用需要安装 Google 语音和视频聊天插件, 在浏览器中成



功安装插件后,可以直接在 Gmail^[5]或 iGoogle^[6]中进行视频和话音通话。但这些插件对于不同浏览器需要不同的开发实现,且需要用户安装,具有一定的安全隐患。为了避免引入新插件,一些 Web 网络电话(如 Alicall^[7]、FlashVoIP^[8])采用普及程度比较高的 flash player 插件实现实时通信功能,可重用 flash 插件自有 RTMP (real time messaging protocol, 实时消息传送协议)^[9]信令和媒体传输协议。但 flash 插件方式需要依托第三软件提供商的支持,大规模商用更需要支付相当可观的费用。

如果彻底不引入插件,Web 实时通信客户端在执行效率方面想接近传统客户端或插件客户端的效果,需要浏览器提供更多新特性的支持。为了实现实时通信功能,浏览器至少需要具备会话管理、音视频编解码引擎和媒体传输等功能。会话管理功能允许开发者为上层应用实现呼叫建立和管理;音视频编解码引擎功能允许外设(如麦克风及摄像头)将采集的数据进行编码并发送给网络,或者将接收的媒体进行解码供外设(如音箱或者显示器)呈现;媒体传输功能实现对不同网络的 NAT(network address translation, 网络地址转换)或防火墙穿越,并实现对媒体内容的封装传输。

对于无插件 Web 实时通信应用开发,开发者需要调用浏览器 API (application programming interface, 应用程序编程接口)才能使用其实时通信模块功能。如果缺少对实时通信模块 API 的统一使用规范,不同浏览器厂商对 API 的定义和使用可能出现很大的差异,开发者需要了解不同版本的 API 使用方法,从而可能会影响开发周期。所以,业界希望 Web 浏览器的实时通信能力被标准化的呼声越来越高。IETF(internet engineering task force, 工程任务组)和 W3C 这两大标准化组织从 2011 年开始积极推进基于 Web 浏览器的实时通信 (real-time communication in web-browsers, RTCWeb) 标准化工作^[10,11],力图提出一个能够在 Web 浏览器上实现用户之间话音和视频等实时通信的标准化框架。IETF 的主要工作在于标准化基于浏览器的实时通信的架构和协议;W3C 的主要工作在于标准化浏览器与 Web 应用之间的 API,即上层 Web 应用通过 JavaScript 编程调用这些被标准化的 API,实现对浏览器 RTCWeb 功能的使用,如捕获本地设备的媒体内容、呈现媒体内容、建立点对点媒体通信等。随着 RTCWeb 标准化的逐渐深入和影响力的逐步增强,各大浏览器厂商如 Chrome、Mozilla Firefox、Safari 和 Opera 纷纷宣布支持或部分支持 RTCWeb 功能,RTCWeb 浏览器逐渐步入舞台并占据相当的市场份额。

2 RTCWeb 关键技术

2.1 RTCWeb 通信模型

IETF RTCWeb 工作组建议的典型架构^[12]如图 1 所示。Web 客户端是基于浏览器的融合 JavaScript、HTML 及 CSS 的技术客户端实现,对浏览器和 Web 服务器的会话信令面操作是通过 JSEP 控制实现的;通过 JavaScript 调用 RTCWeb API,实现对浏览器能力的使用;Web 客户端通过 JavaScript 与 Web 服务器传递基于 Web Socket 承载的 RTCWeb 信令消息。RTCWeb 信令协议由 Web 应用定义,可以是标准的协议,如 SIP(session initiation protocol, 会话初始协议)、XMPP (extensible messaging and presence protocol, 可扩展消息处理现场协议)或私有协议,但媒体协商可参考基于 Offer/Answer 模型的 ROAP (RTCWeb offer/answer protocol);Web 服务器之间采用双方认同的协议,如 SIP 或者 XMPP;浏览器之间采用 RTP(real-time transport protocol, 实时传输协议)或 SRTP(secure RTP, 安全实时传输协议)用于媒体传输,同时采用 ICE(interactive connectivity establishment, 交互式连接建立)协议用于防火墙穿越。

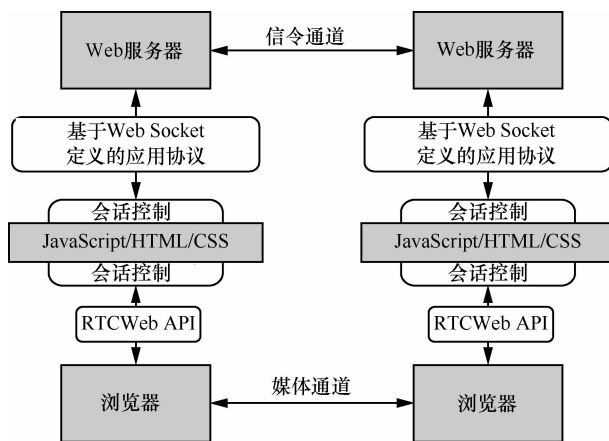


图 1 RTCWeb 通信模型

2.2 RTCWeb 浏览器 API

W3C WebRTC 工作组的关注重点在于浏览器端 API 的标准化^[13],通过调用标准化 API 实现从本地设备(如摄像头、麦克风或网络摄像机)捕获并呈现本地音视频媒体内容、通过 NAT 穿越技术建立点对点连接、基于点对点连接交换媒体内容。这些功能的实现主要由 Stream API 和 Peer Connection 两大类 API 实现。

Stream 抽象了对实际媒体流表示及操作的方法和属性,目前定义了若干重要的接口,主要是 Media Stream。

Media Stream 对象可以表示从远端节点或者发送给远端节点的媒体流;从网络传输 Media Stream 角度看,每个 Media Stream 对象都有一个输入和输出,来自远端节点的 Media Stream(由 Peer Connection 实例化产生)可以看作输入;本地设备产生的 Media Stream(调用 getUserMedia 函数产生)传给远端节点便有一个输出;对于媒体流的呈现,需要专用 URL 来表示一个媒体流并在网页中表现出来(调用 createObjectURL 函数生成)。

Peer Connection 抽象了浏览器之间信令交互及媒体通道建立的方法和属性,通过开通媒体通道,Media Stream 对象表示的媒体流可被送到对端。一个 Peer Connection 对象实现 3 类功能,即 ICE 代理(利用 ICE 方式实现 NAT 穿越,如依赖 STUN 或者 TURN 服务器)、Peer Connection 状态(标识当前连接状态,如“new”、“opening”、“active”等)和 ICE 状态(标识穿越状态,如“new”、“waiting”、“connected”等)。此外,Peer Connection 提供创建媒体协商的方法和属性,如 Offer/Answer 消息构建方法(createOffer/createAnswer)及 SDP 属性设置(setLocal Description/setRemote Description)等。

2.3 RTCWeb 基于 HTML5 呈现方式

W3C WebRTC 工作组对于媒体流的访问和呈现使用 HTML5 video 和 audio 标签^[2],这两个新标签提供了在浏览器中不使用插件播放视频和音频的特性,这正是 HTML5 区别于先前版本的显著特征。非 HTML5 的浏览器一般安装 flash 插件播放媒体,正确播放媒体需要使用<embed>和<object>标签,并且需设置很多参数,媒体标签将会非常复杂。HTML5 对待媒体如同图片,只需要做简单的赋值即可实现对资源的访问:<audio src=“ ”/> <video src=“ ”/>。HTML5 允许为选定的视频流指定播放的参数(如宽度和高度或者播放方式),如<video src=“url” width=“600px” height=“300px” autoplay/>。

RTCWeb 通信在 HTML5 浏览器中呈现音视频的播放需要通过 JavaScript 实现。如图 2 所示,Alice 和 Bob 进行视频通话,Alice 的视频通过 RTP 流到达 Bob 的浏览器,浏览器通知上层 Web 应用;上层 Web 应用向浏览器获取 Alice 视频流的 Bob URL;上层 Web 应用获取 video 标签,并将 URL 赋值给这个 video 标签,更新 video 标签代码,实现媒体播放。

2.4 RTCWeb 媒体协商模型

IETF RTCWeb 工作组建议 RTCWeb 终端间实现媒体建立所需信息协商的协议 ROAP^[14]。ROAP 遵循 Offer/Answer 模型实现浏览器终端之间对 SDP(session description protocol)

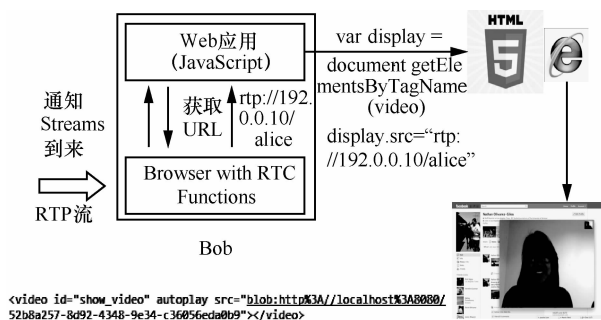


图 2 RTCWeb 基于 HTML5 标签呈现

消息的交换。ROAP 消息内容只是实际传输的 RTCWeb 信令包含的子内容,并不定义实际线上传输的 RTCWeb 信令消息。如图 3 所示,ROAP 目前主要定义了“offer”、“answer”及“OK”等消息,这些消息的产生由上层应用通过 JavaScript 调用 Peer Connection 的若干 API 实现。

2.5 RTCWeb 会话建立方式

RTCWeb 通信系统需要专门的信令状态机实现对多媒

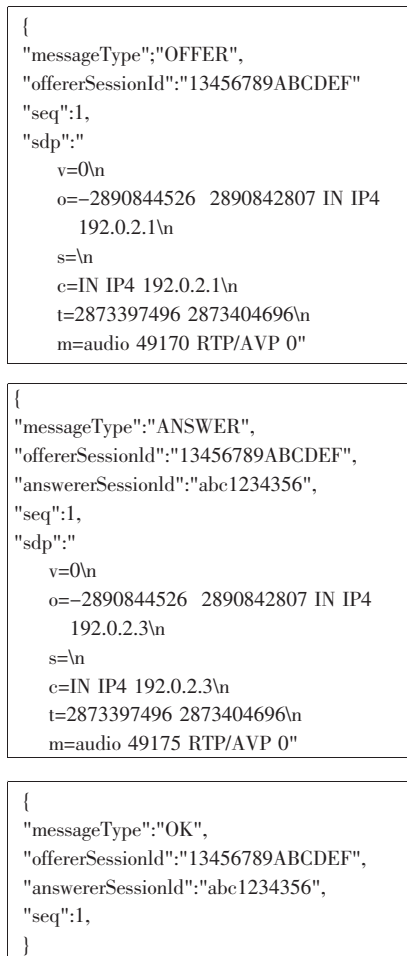


图 3 ROAP 消息展示



体会话信令面的控制,IETF RTCWeb 工作组建议采用一种基于 JavaScript 实现会话建立的协议 (JavaScript session establishment protocol,JSEP)^[15]。JSEP 使得信令状态机从浏览器中解放出来,通过 JavaScript 使控制状态、状态机升级、会话描述修改等操作变得更加灵活和容易。JSEP 能够灵活、简单地处理 Offer/Answer 模型的交互,如图 4 所示。

JSEP 可以和任何会话控制协议相结合,如 JSEP 控制

ROAP 呼叫、JSEP 控制 XMPP 呼叫、JSEP 控制 SIP 呼叫。目前已经出现了 JSEP 控制 SIP 呼叫的雏形 (如 sipML5)^[16],sipML5 信令面采用 SIP over Web Socket,外接专用 RTCWeb 服务器将 SIP 消息从 Web Socket 中解放出来,并发送给 SIP 或 IMS 核心网。

2.6 RTCWeb 信令承载方式

传统上,Web 双向通信是通过 Web 客户端采用 HTTP

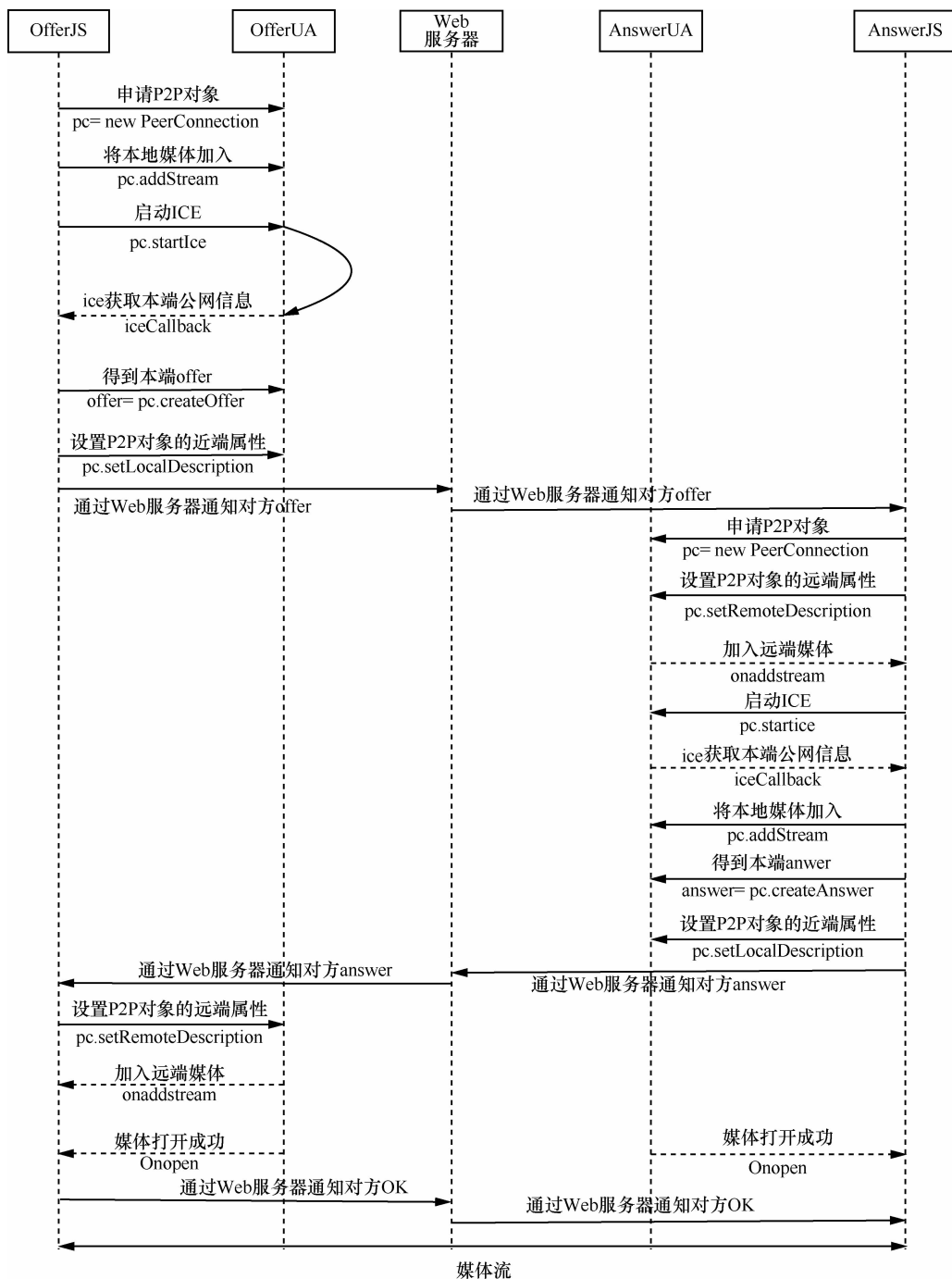


图 4 JSEP 实现流程

轮询方式从 Web 服务器获取消息,这是一种对 HTTP 的畸形使用。基于 HTTP 作为 RTCWeb 信令的传输协议并不合适,主要体现在 3 个方面:第一,Web 服务器采用 HTTP 轮询方式会造成 Web 服务器资源的浪费;第二,HTTP 轮询方式实现实时通信依赖于轮询周期,周期太长会影响实时性,周期太短会耗费资源;第三,Web 客户端和 Web 服务器之间的数据传输需要使用 HTTP 头,这对于实时通信来说是一个比

较大的开销。希望 Web 实时通信能够通过一个 TCP 连接实现双向通信。基于这个想法,IETF 提出了 Web Socket^[17] 协议用于实现双向通信。Web Socket 是一个基于 TCP 及 80/443 端口传输的协议,包含握手和数据传输两部分,握手协议通过带有 Upgrade 字段的 HTTP 请求和 101 的响应交互完成,握手成功后就进入双向长连接的数据传输阶段。如图 5 所示,采用 Web Socket 协议承载 RTCWeb 信令可以提高消息的交互效率,相对 HTTP 轮询而言,实现了真正的双工通信,一个 TCP 长连接相比若干 HTTP 短连接占用资源少,同时数据传输效率更高(Web Socket 数据帧头只有几个字节,而 HTTP 头需要几十个字节)。

3 RTCWeb 与 IMS 融合构想

3.1 IMS 中引入 RTCWeb 客户端的优势

IMS 是一种全新的多媒体业务形式,能够满足用户对更新颖、更多样化多媒体业务的需求。随着 IMS 应用的增加和丰富,IMS 客户端的功能和特性需要不停地变化与演进,客户端软件变得越来越复杂,对 IMS 终端的要求也会更高。目前 IMS 客户端主要包含硬终端和软终端。如果采用 RTCWeb 技术实现 IMS 的免插件的 RTCWeb 客户端,可以进一步丰富 IMS 的终端形态。RTCWeb 客户端和传统的客户端在实现上存在本质区别,如图 6 所示。在传统客户端中,应用与终端平台相关,信令面和媒体面功能是通过调用操作系统 API 实现的,信令面与媒体面在逻辑上分离并通过内部接口通信;在 RTCWeb 客户端中,应用与终端平台无关,信令面和媒体面在逻辑和物理上均分离,信令面由上层应用实现,媒体面由浏览器实现,信令面调用浏览器 API 实现实时通信。

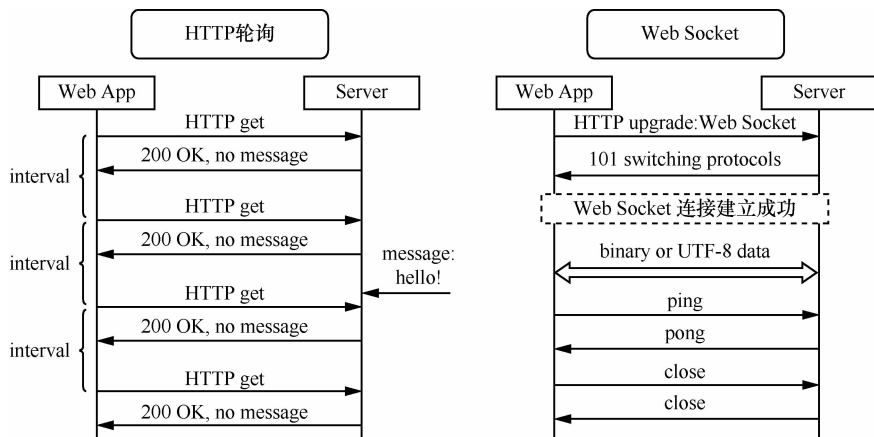


图 5 Web Socket 与 HTTP 实现实时通信流程对比

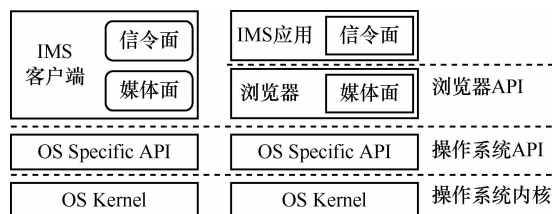


图 6 传统客户端和 RTCWeb 客户端的比较

相比传统 IMS 客户端或插件客户端,RTCWeb 客户端具备更多优势:操作便捷性,基于浏览器环境,实现客户端的免安装(输入网址即可下载 Web 客户端)和免用户干预的升级;开发灵活性,浏览器为上层 Web 应用提供了开发实时通信应用的 API,基于脚本开发客户端应用,周期较短,成本较低;平台无关性,在浏览器中运行,实现了业务与终端平台或操作系统的无关性,有利于跨平台移植,适应于不同终端。

3.2 RTCWeb 与 IMS 融合构想

IMS 增加 Web 客户端形态,使其应用最终能脱离终端平台或操作系统的限制。由于受浏览器支持能力和普及程度的影响,IMS 客户端 Web 化是一个逐步落实的过程。如图 7 所示,第一步,通过对传统客户端软件进行二次开发并向浏览器平台移植,实现基于浏览器的插件客户端,这种客户端借用了浏览器的呈现方式,但逻辑上自成体系,可和 IMS 直接对接;普通插件对于不同浏览器需要不同的开发实现,且需要用户安装,具有一定的安全隐患。第二步,为了避免引入新插件,可以考虑使用普及程度比较高的 flash 插件做客户端,但需要一个接入网关实现 RTMP 协议流的解析和与 IMS 协议的适配。第三步,采用 RTCWeb 技术引入无插件的 Web 客户端,开发者通过



JavaScript 编程实现 Web 应用信令面功能,RTCWeb 信令可以由 Web 应用定义,所以需要一个接入网关实现 RTCWeb 信令和 IMS 协议的适配。

随着终端 Web 化的逐渐深入,承担 RTCWeb 客户端接入功能的网关(RTC-GW)的作用越来越重要。RTC-GW 除了可以接入 RTCWeb 客户端,也可以提供对其他 Web 客户端(如 flash 客户端)的接入,实现客户端通信协议(如 RTCWeb 或者

RTMP)到电信核心网 IMS 协议的适配和转换及媒体转码等操作(为了简化,后文将主要考虑 RTC-GW 提供 RTC 客户端接入功能)。随着 RTCWeb 客户端数量的增长,单个 RTC-GW 无法应对高并发呼叫和媒体处理问题,可以通过多个 RTC-GW 组成协同群组(RTC-GW cluster),实现资源共享。

4 RTCWeb 接入 IMS 的解决方案

RTCWeb 技术有力推动了 IMS 终端形态向无插件 Web 客户端的转型,将来 IMS 用户手持任何终端下载 RTC 客户端即可使用 IMS 业务。本节将重点介绍 RTCWeb 客户端接入 IMS 解决方案及典型业务的实现流程。

4.1 总体结构

如图 8 所示,RTCWeb 系统由 RTC-GW 及 RTCWeb 客户端组成,其中 RTCWeb 客户端由支持 RTCWeb API 的浏览器及相应的 Web App 组成。RTCWeb 客户端可以通过 DNS 获取 RTC-GW 群组的入口地址,入口点一般具有负载均衡器(load balancer)的功能;RTCWeb 客户端向负载均

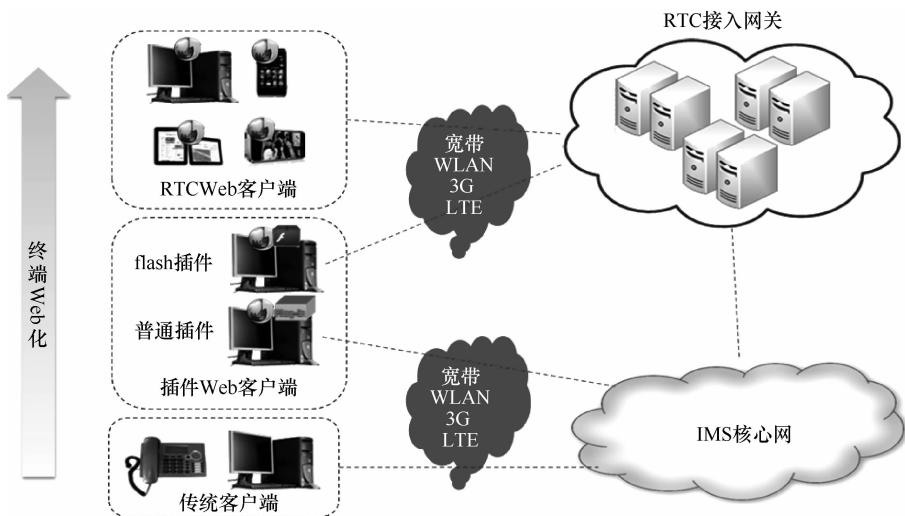


图 7 RTCWeb 客户端接入 IMS 构想

衡器请求可服务的 RTC-GW,负载均衡器采用轮询或者 IP 散列的方式确定一个 RTC-GW,并向 RTCWeb 客户端返回 IP 地址或者域名信息。如果 RTCWeb 客户端位于 NAT 之后,无法与其他客户端建立直连,RTC-GW 可以向 RTCWeb 客户端提供 ICE,帮助客户端实现不同类型的 NAT 穿越。RTCWeb 客户端通过 RTC-GW 实现与 IMS 核心网的通信,其中,RTCWeb 客户端和 RTC-GW 之间形成客户端/服务器方式的通信,包括 RTCWeb 信令和 RTP 媒体交互;RTC-GW 实现了背对背的用户代理(B2BUA),转化 RTCWeb 信令及媒体编解码适配 IMS 信令及媒体编解码,最终接入为 IMS 业务边界控制器(service border controller,SBC)设备,完成 RTCWeb 客户端和 IMS 的互通。

RTCWeb 信令处理/适配及媒体面交换主要涉及了 RTC-GW 以下基本功能。

- 主处理单元(master processing unit,MPU)主要完成 RTCWeb 信令及业务流程控制,如对于 Chrome 浏览器,MPU 处理基于 Web Socket 承载的 RTCWeb 信令。

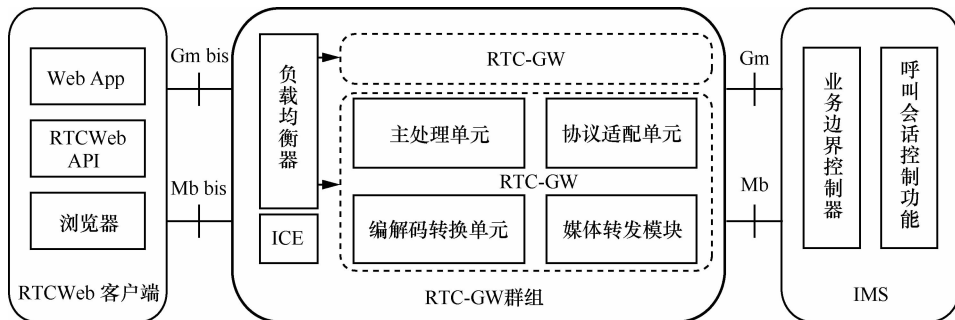


图 8 基于 IMS 的 RTCWeb 系统结构

- 协议适配(protocol adapter,PA)单元主要完成 RTCWeb 信令与 SIP 的协议转换及 SIP 用户代理的信令面相关工作。
- 编解码转换(codec transcoding,CT)单元主要完成媒体格式的编解码转换工作。比如,Chrome 浏览器支持音频编解码 iSAC/iLBC/G.711 和视频编解码 VP8,而 IMS 要求支持音频编解码 G.711/G.723/G.729 和视频编解码 H.263/H.264,一般情况下需要实现浏览器到网络编解码的转换。
- 媒体转发(media forward,MF)模块主要完成 IMS 及 RTCWeb 客户端之间的 RTP/SRTP 媒体接收、发送及解析,如果需要编解码转化需要先将 RTP 分组发送给 CT 处理。

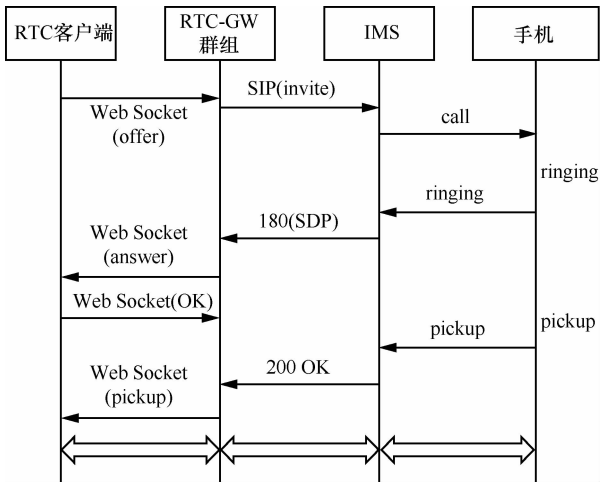


图 9 RTCWeb 用户与传统手机用户之间的呼叫流程

4.2 关键流程

4.2.1 RTCWeb 用户与传统手机用户之间的呼叫流程

RTCWeb 客户端通过 RTC-GW 群组系统接入 IMS,所以 RTCWeb 客户端与 RTC-GW 群组系统之间的交互基于 Web Socket 承载的 RTCWeb 信令,而 RTC-GW 群组系统与 IMS 之间采用 SIP 信令。如图 9 所示,在 RTCWeb 用户呼叫传统手机用户的信令流程中,RTC-GW 群组系统负责在 RTCWeb 客

户端和 IMS 网络之间传递呼叫、振铃及摘机等操作。

4.2.2 RTCWeb 用户之间的呼叫流程

SBC 作为 IMS 的信令和媒体的入口点,将造成两个 RTCWeb 客户端之间通信媒体的迂回。希望在无需合法监听的情况下,RTCWeb 客户端之间的媒体交互能够以 P2P 的方式进行。这个问题可以通过 RTC-GW 群组系统来实现。如图 10 所示,RTC-GW 群组系统在媒体协商 Offer/

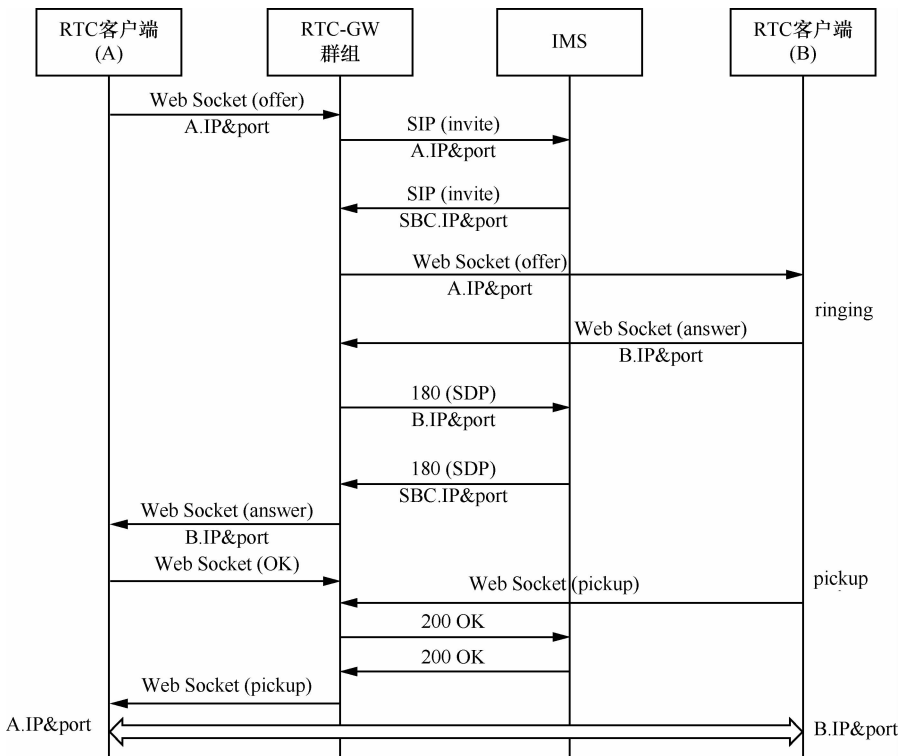


图 10 RTCWeb 用户之间呼叫流程

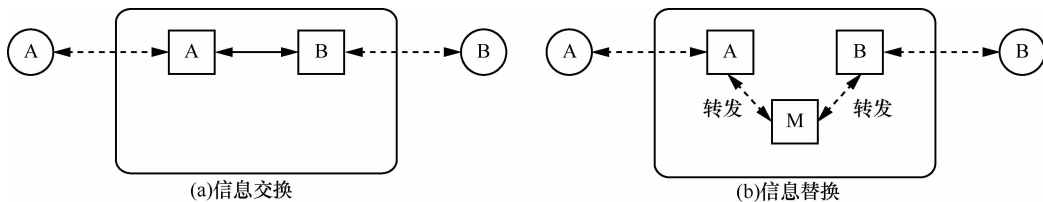


图 11 RTC-GW 群组系统实现媒体地址替换

Answer 交互中,将 SBC 的媒体面 IP 地址和端口隐藏,取而代之的是 RTCWeb 客户端 A 和 B 的媒体面 IP 地址和端口,实现 RTCWeb 客户端之间的媒体直传。

RTC-GW 群组系统如何将 SBC 的媒体面 IP 地址和端口替换成 RTCWeb 客户端的媒体面 IP 地址和端口,实现方法有很多,图 11 列举两种方式仅作参考:第一种方式,RTC-GW A 和 B 分别为 RTCWeb 客户端 A 和 B 提供服务,RTC-GW A 和 B 直接交换媒体面信息,将 SBC 的媒体面信息换成对端 RTCWeb 客户端的媒体面信息;第二种方式,RTC-GW A 和 B 分别为 RTCWeb 客户端 A 和 B 提供接入,并将信令转发给同一个 RTC-GW 提供服务,这样同一个 RTC-GW 便直接将 SBC 的媒体面信息替换成不同的 RTCWeb 客户端的媒体面信息。

5 结束语

RTCWeb 技术成为推进 Web 实时通信的优选方式之一,它具有传统客户端无法比拟的优势,如操作便捷性、开发灵活性及平台无关性等。本文首先对 RTCWeb 通信模型及关键技术进行介绍及分析;在此基础上,提出了采用 RTCWeb 技术实现 IMS 免插件 Web 客户端的想法,并提出相应的 RTCWeb 与 IMS 的融合方案;方案细化了 IMS RTCWeb 客户端和 RTC-GW 接入网关的实现方式;通过本文建议的 RTCWeb 和 IMS 融合方案,可进一步丰富 IMS 的终端形态,简化 IMS 客户端部署方式,提升用户体验。

此外,关于 RTC-GW 群组实现方式,可以结合云计算技术形成一个 RTC-GW 云。通过云计算技术使 RTC-GW 群组具有低成本、易扩展、易管理等优点。RTC-GW 云系统保证 RTC-GW 功能实体之间相互协作,实现负荷分担,对外实现拓扑隐藏,使用了数据多副本容错、同构可互换等措施来提升可靠性。RTC-GW 云还可以对外提供云服务,如 SaaS(software as a service,软件即服务),可能

包括 RTC-GW 的功能或者其子功能,如会话控制、协议适配或编解码转化甚至 NAT 穿越等;PaaS (platform as a service,平台即服务),可能提供 RTCWeb 开发平台,即系统可以对外提供 JavaScript 开发包供第三方网站或者软件进行调用。

参考文献

- 1 Open web platform. http://www.w3.org/wiki/Open_Web_Platform
- 2 HTML5. <http://www.w3.org/TR/html5/>
- 3 Facebook. <http://www.facebook.com/>
- 4 GoogleTalk. <http://www.gtalk.com.cn/>
- 5 Gmail. <http://mail.google.com/>
- 6 iGoogle. <http://www.google.com/>
- 7 Alicall. <http://www.alicall.com/>
- 8 FlashVoIP. <http://www.alicall.com/>
- 9 RTMP specification. <http://www.adobe.com/devnet/rtmp.html>
- 10 W3C WebRTC working group. http://www.w3.org/standards/techs/webrtc#w3c_all
- 11 IETF RTCWeb working group. <http://datatracker.ietf.org/wg/rtcweb/>
- 12 Alvestrand H. Overview: Real Time Protocols for Browser-based Applications. W3C, June 20, 2012
- 13 WebRTC 1.0: real-time communication between browsers. <http://www.w3.org/TR/webrtc/>
- 14 Jennings C, Rosenberg J. RTCWeb Offer/Answer Protocol (ROAP). IETF, October 30, 2011
- 15 Uberti J, Jennings C. Javascript Session Establishment Protocol. IETF, June 4, 2012
- 16 sipML5. <http://www.sipml5.org/>
- 17 RFC 6455. <http://tools.ietf.org/draft-ietf-hybi-thewebsocketprotocol>

【作者简介】 乐利锋,男,博士,中国移动通信研究院网络技术研究所高级工程师、研究员,主要研究方向为分布式技术及 IMS 网络。彭晋,男,博士,中国移动通信研究院网络技术研究所高级工程师、网络架构实验室主任,主要研究方向为分布式技术及移动互联网。段晓东,男,中国移动通信研究院高级工程师、网络所所长,主要研究方向为 IP 技术、分布式技术及移动互联网技术。

(收稿日期:2012-08-07)