



移动应用无需心跳维持长期在线的方案研究

何 峣,张 荣

(中国电信股份有限公司广东研究院 广州 510630)

摘 要:针对微信、QQ 和微博等小流量长在线应用使得无线网络空口信令资源消耗、终端能耗增大的问题,在分析业界同类解决方案的基础上,提出一种全新的无需心跳仍能维持移动应用长期在线的解决方案,彻底解决移动应用由于心跳机制引发的无线空口信令消耗的问题。

关键词:移动长在线应用;心跳;信令风暴

doi: 10.3969/j.issn.1000-0801.2014.01.019

Research on Solution of Mobile Application Always Keeping Online Without Heartbeat

He Yao,Zhang Rong

(Guangdong Research Institute of China Telecom Co., Ltd., Guangzhou 510630, China)

Abstract: Along with the mobile application of the increasingly rich, especially widely used as WeChat, QQ and Weibo, the wireless network air interface signaling resource consumption issues highlighted, while the terminal energy consumption. Based on the analysis of the same industry solution, a new no heartbeat solution which can still maintain a mobile application always online was proposed, the wireless command consumption problem was solved.

Key words: always online mobile application, heartbeat, signaling storm

1 引言

根据百度在 2013 年第一季度的移动应用统计分析,平均每个终端的应用总量(包括操作系统自带、预装和用户下载)已经达到 139.1 个,而用户日均启动的应用数量达到 12.6 个,主要聚焦在即时通信、搜索、购物、微博、阅读等应用。移动终端上越来越多的应用需要与其服务器通过心跳机制保持长期连接,维持应用一直在线的状态,以便应用能第一时间收到服务器的推送信息,且各应用之间的心跳周期都是独立的。

移动通信无线网络在发送数据时需要经过空口连接建立、数据传输、等待休眠、连接拆除的过程。由于心跳机制及应用数据发送时间与网络休眠时间相互分开,由应用

层与网络层独立控制,容易导致空口连接频繁拆建。2011 年中国电信股份有限公司福建分公司校园网络由于 QQ 应用集中,导致网络拥塞,用户体验急速下降。2012 年,NTT DoCoMo 无线网络发生故障,直接原因是一款基于 Android 的应用以每 3~5 min 的频率发出网络连接信令消息,造成空口连接频繁拆建,引发信令风暴。

在当前无线网络业务容量有限的条件下,各个应用以心跳机制分别维持自己的长期在线的方式,是造成无线空口资源消耗增大的重要原因之一。

LTE 网络较目前 3G 网络在理论上容量有较大提升,综合考虑接入能力、前向控制信道能力、反向控制信道能力以及 RRC 连接数,LTE 网络较 3G 网络业务容量理论增长 4~5 倍。在一定程度上,可缓解网络容量对应用承载不

足的问题。但是,2012 年全球移动互联网流量增长达到 70%;2013 年中国移动互联网用户数相比 2012 年同期增长了 18.1%,且仍将保持高速增长。以此发展速度计算,3~4 年后,LTE 网络仍将面临现有 3G 网络的长期在线应用的承载问题,无线信令资源紧张的问题长远来看并不能只通过无线网络技术升级来解决。

2 国内外现状

2.1 标准

OMA (Open Mobile Alliance, 开放移动联盟)、IETF、3GPP 和 W3C 等国际标准组织对移动互联网消息推送机制都进行了相关的研究和标准制定,其中 OMA 制定的相关标准,更贴近移动互联网业界的需求,继 2005 年开始的 OMA Push 以及 2010 年开始的 Notification Channel 标准后,于 2012 年立项的 Always Online Infrastructure (AOI) 项目,则开始定义下一代在线和通知系统,目前 AOI 项目已经完成需求定义、主要架构和关键交互流程的制定,预计 2013 年底将完成所有接口与流程的规范制定。如图 1 所示,AOI 系统架构是在移动应用客户端与应用服务器之间,架设了 AOI 客户端中间件和 AOI 服务器,用于接管原应用客户端与应用服务器之间的消息推送和长连接维持的职责,各个应用服务器先把消息推送给 AOI 服务器,AOI 服务器再把消息推送给 AOI 客户端中间件,最后 AOI 客户端中间件把消息推送至对应的应用客户端。同时 AOI 客户端中间件收敛终端中多个应用的网络连接数量,并统一发送心跳消息,以维持 AOI 客户端中间件与 AOI 服务器之间的长期连接。使用了 AOI 架构后,各个应用客户端就不需要与各自的应用服务器建立相互独立的长期网络连接。

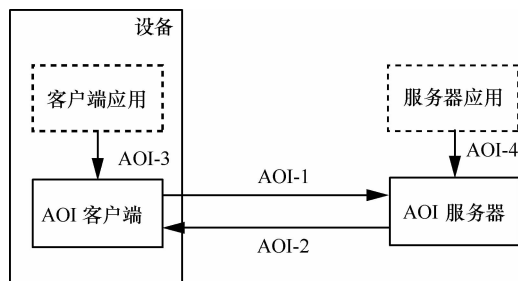


图 1 OMA AOI 标准项目的系统架构

中国通信标准化协会 (CCSA) 也于 2013 年 6 月立项《移动长在线应用系统总体技术要求》进行标准制定工作,现已完成征求意见稿的编制,并计划在接口、终端和平台服务器领域进行标准分册的立项工作。该标准主体技术架

构参照了 OMA AOI 架构,并结合国内运营商自身特点,加入了运营商自身网络通信能力和 QoS 方面的考虑,根据已经开展的业务试验情况,细化了各接口的参数。如图 2 所示,国内标准在原 OMA AOI 架构的基础上,加入国内运营商特有的 WAP 推送机制,针对移动终端 IP 地址内网化的趋势,加入了 AOI 代理的可选模块,从而更好地发挥运营商的优势,将短信与 IP 长连接结合使用,既可节省无线资源又可向用户提供及时可靠的消息传递。在此基础上,将进一步结合策略控制系统、DPI 系统等,对关键业务提供区分式的质量保障,从而进一步提升用户体验。

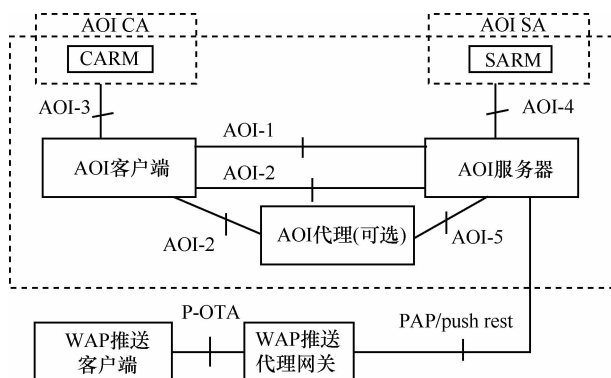


图 2 CCSA 移动长在线应用系统总体架构

2.2 互联网终端与平台厂商方案

苹果、谷歌和微软等主流终端与操作系统平台厂商分别为自有的移动操作系统集成了在线推送通知服务架构。移动应用可以借助这些移动操作系统提供的推送通知服务向其客户端发送通知消息,而无需额外开发长期在线的通知功能。如图 3 所示,苹果公司通过部署 Apple Push Notification Service (APNS) 以及规范相应的 APNS 流程,使得苹果手机上多个应用能够共享手机与 APNS 之间建立的长连接,从而减少心跳分组的数量;通过 APNS 推送消息网关将应用服务器的通知消息发送给终端以降低应用和终端之间的数据交互次数。

如图 4 所示,谷歌从 Android 2.2 版本后开始导入 Android Cloud to Device Messaging (C2DM) 服务架构,C2DM 也是在 Android 操作系统与第三方应用服务器之间架设了 C2DM 服务器,用作推送通知消息的桥梁。Android 操作系统与 C2DM 服务器之间维持着一条长连接。

2.3 电信运营商方案

韩国 SKT 在 2012 年 5 月部署了 Smart Push 系统,通过统一管理心跳信号,及时检测各聊天工具、社交网络程

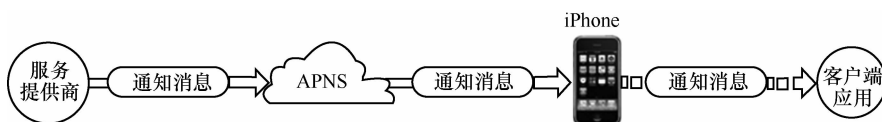


图 3 苹果公司 APNS 系统示意

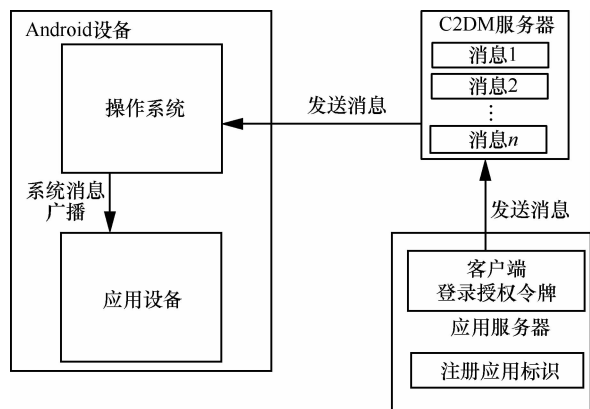


图 4 谷歌公司 C2DM 系统示意

序和智能手机的交互状态,减少不必要连接信号的产生,尽可能避免通信网超负荷。Smart Push 架构类似于 AOI 架构,如图 5 所示,在终端侧安装 Smart Push 客户端中间件,手机应用按照接口规范与 Smart Push 客户端中间件对接,Smart Push 客户端中间件与 Smart Push 服务器保持心跳长连接,应用服务器按照接口规范与 Smart Push 服务器对接。Smart Push 架构目前只传递心跳(上行)和通告消息(下行),已有 Kakao talk、me2day 等近 11 款应用使用该系统。

国内运营商主要通过网络优化以及与应用商协调心跳周期等参数优化的方式缓解长期在线应用对网络资源的消耗。例如 QQ 服务器将一般系统消息由实时发送调整为间隔 60 s 后发送,降低产生空口信令交互的概率,约减少 2% 信令交互次数。

2.4 现有方案存在的问题

无论是标准组织、终端还是平台厂商和运营商,其方案都有共同特点,即都需要在终端侧部署长在线客户端中间件,在网络侧部署长在线服务器,长在线客户端中间件与服务器之间用心跳机制保持一条网络连接。长在线客

户端中间件用于汇聚和收敛应用客户端的长在线网络连接,并向应用客户端分发推送消息,长在线服务器用于汇聚应用服务器的推送消息,并推送至长在线客户端中间件。

这种方案必须要在终端侧部署客户端中间件,给推广带来一定的阻力。

- 由于中间件在终端中主要以后台运行的方式为其其他应用提供服务,而不是直接为用户提供服务,因此用户没有太多的积极性自行安装该中间件。
- 即使把中间件预装在终端中,用户也可通过获取终端操作系统管理员权限的方式,强行卸载预装的软件。
- 移动互联网应用厂商为规避法律责任,把中间件模块集成到自身的应用客户端安装包中的积极性也不会太高。
- 即使在操作系统中集成了中间件,如谷歌的方案,但由于 Android 系统的开放性,集成的功能可能由于终端厂商对原生操作系统的裁剪修改而被去除,加上谷歌业务退出中国的缘故,难以保障中间件与长在线服务器之间的长连接是稳定可靠的。

3 无心跳维持应用长在线的方案

3.1 需求分析

应用客户端与应用服务器之间维持长期连接,主要是为了应用服务器能寻址找到应用客户端,应用客户端能接收到应用服务器推送过来的消息,同时应用客户端也能通过此连接向应用服务器发送消息。当终端接入网络后,获取的不是公网地址而是运营商内部的私网地址时,位于私网与公网出口位置的网关设备,如 WAP 网关和 NAT 设备,会对一定时间间隔内无数据流量的网络连接进行自动

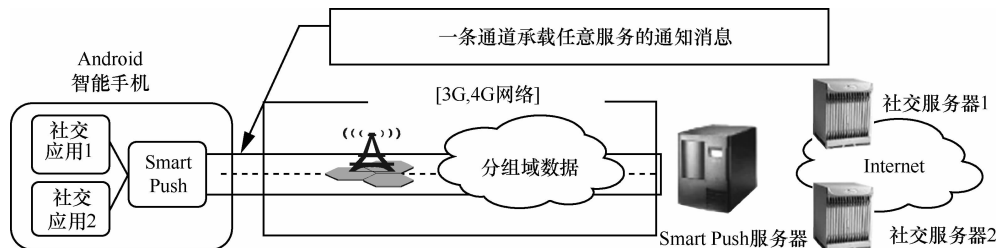


图 5 韩国 SKT Smart Push 系统示意

断开操作,之所以要维持心跳机制,就是为了保持应用客户端与应用服务器之间的这条长期网络连接是活跃的,而不被网关设备断开。

因此,维持移动应用长期在线的内在本质需求,归结为两点:第一,应用客户端有消息要发送给应用服务器时,需要找到应用服务器,并能把消息发送过去;第二,应用服务器有消息要推送给应用客户端时,需要找到应用客户端,并能把消息推送过去。

由于应用服务器部署在公网环境,具备公开的域名或公网 IP 地址,应用客户端自身无论以公网地址还是私网地址接入网络,都可以找到应用服务器。当应用客户端要向应用服务器发送消息时,除了利用上述长期连接的通道外,还可以用短期连接的方式,如 Web 技术的请求+回应的方式。短期连接同样满足送达消息的需求,最关键的一点是不再需要在应用客户端和应用服务器之间维持一条长期的网络连接,减轻了应用服务器维护大量客户端在线状态的负担,正因为这种弱化在线状态的机制,使得应用服务器可以更灵活地实施后台系统的负载均衡策略。至此,利用现有的技术就能很好地解决第一点需求,在此暂称这种为正向短期连接技术。

针对第二点需求,由于应用客户端接入网络后,可能获得公网 IP 地址,也可能获得私网 IP 地址,这取决于接入运营商的策略,使得应用服务器不能完全根据应用客户端登录时的公网 IP 地址+端口连接到客户端,因为应用客户端可能在运营商的私网内,就更不用说推送消息过去了。利用上述的类 AOI 架构,可以满足需求,但存在推广难的问题,难在方案没有给应用开发商足够的自由度和开发度(应用客户端必须在终端内接入到长在线客户端中间件),由于各个应用所使用的通信协议千差万别,在通信通路上增加了一个中间节点,在一定程度上限制了应用开放商的自由度,进而限制了应用特色功能的开发。另一方面,对于运营商来说,依然存在着心跳长期连接,由于心跳导致无线网络空口信令额外消耗的问题,依然没有彻底解决。本文阐述的方案是针对第二点需求,从既要保持通信层面的管控,又要给予应用开发商一定的自由开放度的折中视角提出,并同时解决运营商无线网络空口资源的问题。

3.2 关键技术

解决问题的关键技术,是反向短期连接技术,即应用客户端同时兼任服务器的监听角色,应用服务器同时兼任客户端的发起连接并发送消息的角色,当应用服务器有消

息推送时,主动向应用客户端发起网络连接,应用客户端接受连接请求并建立网络连接后,即可接收应用服务器的消息推送,待推送完成后,应用服务器主动断开这条反向网络连接。如果应用客户端接入网络后获取的是公网 IP 地址,则应用服务器可利用此公网 IP 地址以及登录过程中协商好的端口号,利用反向短期连接技术,方便地找到应用客户端建立网络连接并发送消息。但当应用客户端接入网络后获得的是运营商的私网地址,应用服务器获取到的应用客户端地址,只是应用客户端所在私网的公网出口网关设备的公网地址以及被转换过的端口号,不能直接使用反向短期连接技术,此时需要在运营商私网中部署反向连接代理的节点,反向连接代理分别与各个应用服务器保持长期连接,当应用服务器推送消息时,先把消息发至反向连接代理,反向连接代理由于和应用客户端位于同一私网中,可直接使用反向短期连接技术,向应用客户端发起连接请求,建立通信后把之前缓存的消息发送给应用客户端,发完后断开连接。

正向短期连接技术与反向短期连接技术相配合,就可完全抛弃心跳机制也可维持移动应用长期在线的需求。所述的关键技术,并不需要在终端侧部署类似中间件的模块,只需在私网这种特定环境下,部署代理设备即可,应用开发商只需按照一定规范进行开发,保持了较大的自由度。

反向短期连接技术涉及如下几个问题。

- 如果终端休眠了,岂不是收不到来自应用服务器或反向连接代理的连接请求了吗?
- 应用服务器和反向连接代理由于承担了客户端发起连接的职责,由于每发起一条连接请求就会消耗一个本地端口,每个 IP 地址最多只配有 65 535 个端口,难道一台应用服务器或反向连接代理就只能为这么少的应用客户端提供服务吗?

以下从智能手机终端的硬件层面和操作系统层面进行解答,以 Android 智能手机为例。

- Android 手机有两个处理器,一个是 AP(application processor,应用处理器),另一个是 BP(baseband processor,基带处理器)。AP 是 ARM 架构的处理器,用于运行 Linux+Android 系统;BP 用于运行实时操作系统(RTOS),通信协议栈运行于 BP 的 RTOS 之上。由于通信协议栈运行于 BP,一旦收到数据分组,BP 会将 AP 唤醒,唤醒的时间足够 AP 完成对 BP 收到协议的处理,如处理协议分组的时间很长,



则需要加上 Android 操作系统的 WakeLock 资源锁, 处理完后再释放。其次, 需要注意的是, TCP 可靠连接可以将 AP 唤醒, 但是 UDP 数据分组则不能, 因为系统底层对于 TCP 可靠连接过来的数据, 会产生 AP 中断来唤醒 AP, 但是 UDP 则不会, 因为 TCP 可靠链接是客户端与服务器之间经过握手验证的, 数据来源是可靠的, 但如果 UDP 也能唤醒 AP, 则手机很容易遭到 UDP 数据攻击, 导致手机耗电量大幅度上升。因此, 只要使用 TCP 作为反向短期连接的通信协议, 即使手机休眠了, 也可被重新唤醒。

- 服务器操作系统, 如 UNIX、Linux 和 Windows 等是根据五元组(目的 IP 地址、目的端口、源 IP 地址、源端口和协议号)唯一确定一条网络连接的, 并不只是根据本地 IP 地址+端口确定, 而且在 Socket 编程中可使用 SO_REUSEADDR 特性, 对地址和端口进行复用, 即同一个本地 IP 地址和同一个端口, 只要对端的 IP 地址或端口不同, 就可创建一条唯一的网络连接。由于应用客户端的地址是不同的, 应用服务器或反向连接代理可利用本地端口复用的技术, 支持十万级甚至百万级的网络连接数量。

3.3 总体架构及组网

该方案的适用范围是当移动终端以 2G/3G/LTE 通道接入网络时, 不适用于移动终端以 Wi-Fi 方式或其他方式接入网络的情况, 若移动终端使用 Wi-Fi 或其他方式接入网络, 则移动应用可沿用现有的心跳长期连接机制, 因 Wi-Fi 接入不涉及终端在休眠态与激活态之间切换引发的空口信令消耗问题。

方案的总体架构如图 6 所示, 应用服务器包含正向短连接服务器模块、反向短连接客户端模块和代理长连接模块; 应用客户端包含正向短连接客户端模块和反向短连接服务器模块; 反向连接代理包含代理长连接客户端模块和反向短连接客户端模块。

位于应用服务器的正向短连接服务器模块与位于应用客户端中的正向短连接客户端模块, 是应用客户端向应用服务器方向发送数据时的通道, 用接口 A 表示, 该通道可完成用户注册、登录、终端网络连接状态更新和发送消息等功能。

位于应用服务器或反向连接代理的反向短连接客户端模块与位于应用客户端的反向短连接服务器模块, 是应用服务器向应用客户端方向推送数据时的通道, 用接口 B 表示, 该通道可完成消息推送等功能。

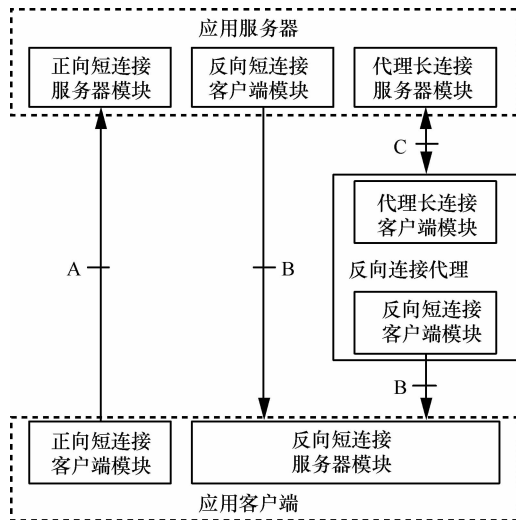


图 6 总体架构

位于反向连接代理的代理长连接客户端模块与位于应用服务器的代理长连接服务器模块, 是应用服务器向位于运营商私网内的应用客户端推送数据时的代理通道, 用接口 C 表示, 该通道可完成消息推送的代理中转功能。反向连接代理作为客户端, 与应用服务器建立长连接网络通道, 并以心跳机制维持该通道, 由于该通道并不处于无线网络中, 而是存在于有线的 IP 承载网, 所以心跳机制并不会影响到空口信令。当应用服务器需要推送数据给应用客户端时, 则先通过接口 C 把数据推给反向连接代理, 反向连接代理再通过接口 B 推送给应用客户端。

以中国电信 CDMA 网为背景介绍该方案的组网示例, 如图 7 所示, 移动终端通过 3G 接入 PDSN 后, 有 3 种路径进入公网, 见图 7 中(1)、(2)、(3): 第 1 种是直接由 PDSN 进入; 第 2 种是先进入运营商的私网, 然后再经过 NAT 地址转换设备进入; 第 3 种是先通过 WAP 代理协议进入运营商的 WAP 网关, 然后 WAP 网关去除代理协议的分组头, 最后数据进入公网。由于第 2 种和第 3 种情况, 移动终端获取的都是运营商的私网地址, 在私网内, 应用服务器无法直接访问, 所以必须在私网中部署反向连接代理, 作为消息推送的中转站, 反向连接代理通过 NAT 设备或 WAP 网关, 可与应用服务器建立长期连接, 同时由于反向连接代理与移动终端位于互通的私网, 所以可以利用反向短连接的技术, 与应用客户端建立起连接, 从而实现公网的消息数据可经过代理中转推送至私网中。

该架构不再要求应用客户端必须接入一个类似终端中间件的模块上, 应用开发商只需遵照一定的规范进行开

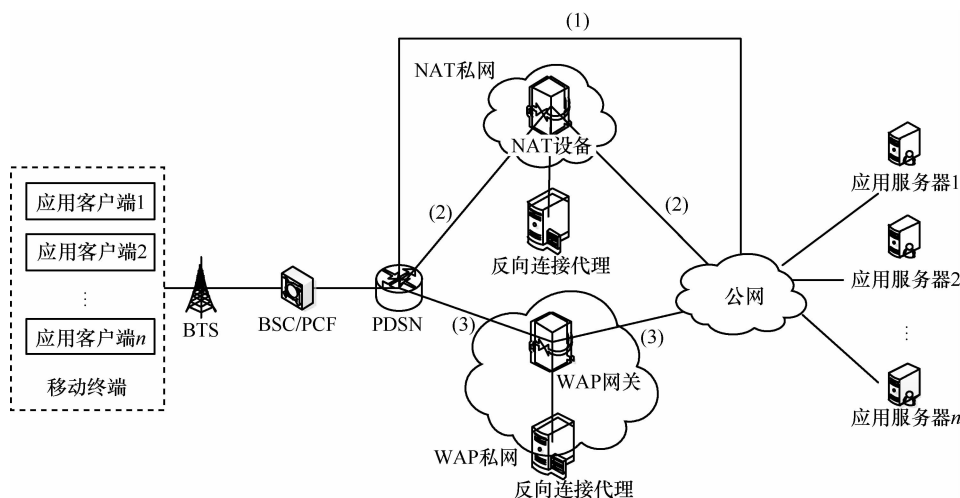


图7 组网示例

发即可,提供了更大的自由度。

4 方案可行性

本文所述方案均建立在成熟技术的基础之上,涉及的软件模块及软件系统,均可基于 TCP/IP 架构,使用现成的 Socket 接口编程实现。其中接口 A 和接口 B,可采用目前业界广泛使用的 Web+rest 方式实现,既稳定可靠,性能方面又可得到各种优秀的开源或付费框架支持;接口 C 由于需要面向各种不同类型的移动应用服务商,需采用兼容性和扩展性都符合要求的协议,成熟的 XMPP 是不错的选择,其兼容性和扩展性得到目前业界广泛的认可。鉴于采用的是成熟技术,从而避免了由于新技术试验而额外消耗的投入成本。

本方案不涉及任何硬件、操作系统、底层协议栈和网络设备等相对基础层面的设施改造,无论终端侧、网络侧或平台侧,采用通用的 IT 和 IP 软硬件基础设施即可实现。反向连接代理可伴随在移动分组网中的 NAT 设备和 WAP 网关的位置进行部署,运营商只需针对反向连接代理的私网 IP 地址,对私网各网段进行路由互通的配置,就可实现反向连接代理的云化集中部署。由于不涉及底层的改造,从而进一步减少了基础设施层面因升级而额外投入的成本。

应用开发商无需改动现有的心跳机制,因为现有的机制仍适用于 Wi-Fi 网络环境,只需增加对本方案的软件支持,并在终端应用侧增加对终端当前接入的网络类型进行判断即可,当终端以 3G 接入则启用该方案的软件逻辑,当以 Wi-Fi 接入则启用原有心跳的软件逻辑。以兼容演进而非激进变革的方式进行,最大程度地实现软件重用和利旧,从而使软件开发的投入尽量少。

因此,从技术成熟度、系统可实现性、产业链合作的便利性和投入成本等多方面分析,本方案在技术和经济上均可行。

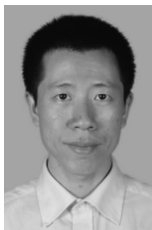
5 结束语

移动互联网应用无心跳维持长期在线的方案可以较好地解决运营商当下面临的困难,减少长期连接对无线资源的额外消耗,同时降低向应用开发商推广的阻力,给予了更大的自由度。反向连接代理则更适合由运营商提供,其承担起移动互联网业务可感知的智能管道节点角色。

参考文献

- 1 百度无线数据报告. <http://developer.baidu.com/report>
- 2 iPhone Push. <http://baike.baidu.com/view/7794523.htm>
- 3 OMA always online infrastructure (AOI) draft version 1.0. <http://www.openmobilealliance.com/>

[作者简介]



何晓,男,中国电信股份有限公司广东研究院工程师,主要从事 IPQoS、移动互联网领域的产品研发工作。

张荣,女,中国电信股份有限公司广东研究院高级工程师,主要从事 IP 网络技术、策略控制技术与移动互联网领域的研究工作。

(收稿日期:2013-12-02)