千万级高性能长连接Go服务架构实践

原创 欢迎关注的 百度Geek说 2024年01月22日 18:02 上海

点击蓝字，关注我们

作者 | glstr

导读
introduction

移动互联网时代，长连接服务成为了提升应用实时性和互动性的基础服务。本文主要介绍了百度系内基于golang实现的统一长连接服务。主要从统一长连接功能实现和性能优化等角度，描述了统一长连接服务在设计、开发和维护过程中面临的问题和挑战，重点介绍了解决相关问题和挑战的解决方案和实践经验。

全文7631字，预计阅读时间20分钟。

01 GEEK TALK
摘要

移动互联网时代，用户对服务的实时性、互动性有了更高的要求，因此能够极大提升服务实时性、互动性的长连接服务，成为了移动互联网应用的刚需。长连接，顾名思义，是应用存活期间和服务端一直保持的网络数据通道，能够支持全双工上下行数据传输。其和请求响应模式的短连接服务最大的差异，在于它可以提供服务端主动给用户实时推送数据的能力。

不过，长连接作为基础服务，要做到低延时、高并发、高稳定性，对服务的开发和维护有较高的要求，如果每个业务都维护自身的长连接服务，一方面有较大的重复开发和维护成本，另一方面长连接服务功能迭代、服务稳定性、专业性很难跟上业务诉求。

因此，统一长连接项目通过打造完整的端到服务端的长连接服务系统，给业务提供一套安全、高并发、低延迟、易接入、低成本的长连接服务能力。

02 GEEK TALK
统一长连接服务

统一长连接服务主要目的是给业务提供一套安全、高并发、低延迟、易接入、低成本的长连接服务系统。主要愿景包括：

1.满足百度体系内APP主要场景如直播、消息、PUSH、云控等业务对长连接能力的诉求，提供安全构建、维护长连接和数据上下行能力；

2.保障服务的高并发、高稳定性、低延迟，保障长连接服务的专业性和先进性；

3.支持长连接多业务长连接复用，减少APP建立和维护长连接的成本和压力；

4.支持业务快速接入长连接，提供给业务简单清晰的接入流程和对外接口。

03 GEEK TALK
问题和挑战

为了构建能够满足业务诉求的长连接服务，统一长连接服务在设计、开发和维护过程中，我们面临着一些问题和挑战需要去解决，其主要包括以下两个方面。

3.1 功能实现

统一长连接服务与接入业务的边界关系是长连接业务架构设计的首要问题。与业务专用的长连接服务不同，统一长连接服务要实现的目标是多业务方共用一条长连接。因此在设计时既要考虑到不同业务方、不同业务场景对长连接服务的诉求，同时，也要明确长连接服务的服务边界，避免过多介入业务逻辑，限制后续长连接服务的迭代和发展。

通常，业务对长连接服务的主要诉求包括三个方面：

1.连接建立、维护、管理；

2.上行请求转发；

3.下行数据推送。

数据上下行过程中，需要能够支持不同业务数据协议上的差异；此外，根据不同的业务类型，对下行数据推送模式、推送量级有着不同的要求。以长连接服务常见的业务方：消息、直播、PUSH为例。

1.消息场景：主要是私信和有人数限制(500-1000)的群聊，推送通知的模式主要是单播和批量单播，推送频率和并发度，依赖于私信和群里消息的发送频率。

2.直播消息场景：直播消息是一个组播场景，组播成员数和直播间在线人数相关，峰值百万甚至千万，推送消息频率高。

3.PUSH场景：PUSH 场景是对一个固定人群下发消息，推送模式是批量单播，推送频率相对而言比较低。

业务	支持能力	推送场景	推送预计UPS
消息	请求转发 下行推送	单播/批量单播	万级
直播	请求转发 下行推送	组播	千万级
云控	请求转发 下行推送	批量单播	百万级
PUSH	请求转发 下行推送	批量单播	百万级

综上，统一长连接服务，要实现的服务能力如下：

1.连接建立、维护、管理；

2.上、下行数据转发，区分不同业务、兼容不同业务消息协议；

3.下行推送上，支持单播、批量单播、广播。

3.2 性能优化

统一长连接服务因为要给百度体系APP提供长连接能力的服务，要做到高并发、高可用、高稳定性。具体到长连接服务本身，其主要包括以下几个方面。

3.2.1 建联qps、延时、成功率、连接维持

长连接在app打开同时需要完成建立、并在app存活期间保持连接存活。因此，长连接服务要支持万级别的建联qps和千万级别在在线连接维持，并支持横向扩容。此外，连接建立作为长连接服务的基础，建联的成功率和延时重中之重。

3.2.2 上行请求qps、延时、成功率

连接建立完成后，需要将业务请求转发给业务侧，这个依赖于用户规模和请求频率，一般至少要支持到几十乃至百万级别，并可以支持横向扩容。

3.2.3 下行请求qps，延时、成功率

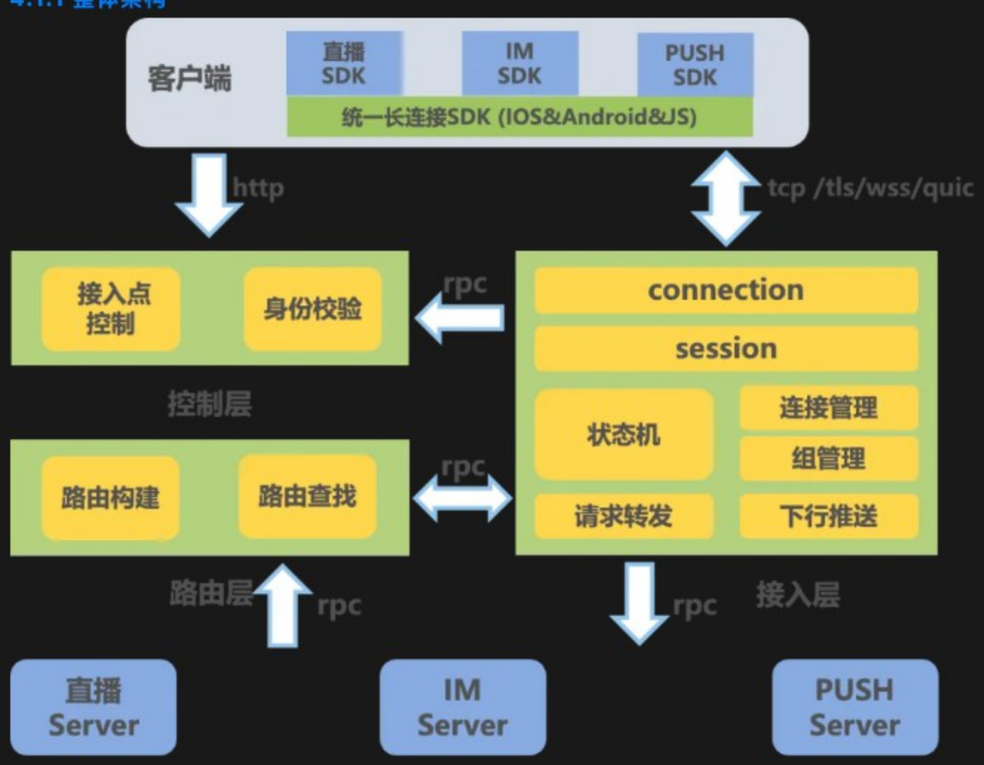
下行请求根据业务场景的不同，分为批量单播和组播，且不同业务对应请求qps要求不一样，一般批量单播需要支持到百万级ups，组播要支持到千万级ups，且支持横向扩容。

04
GEEK TALK
整体介绍

下面简单介绍下，为了完成上述目标，统一长连接服务所做的一些方案设计和实践经验。

4.1 整体介绍

4.1.1 整体架构



统一长连接服务整体架构如图所示，整个服务包括统一长连接SDK、控制层、接入层、路由层四个部分组成。统一长连接SDK归属于客户端，控制层、接入层、路由层归属于服务端。每个组成部分在整体系统中的扮演的角色和功能如下。

1.统一长连接SDK：统一长连接SDK归属于客户端，负责连通业务SDK和长连接服务端，其主要职责包括：

- 请求控制层，获取能够标识设备合法身份的token、长连接接入点和长连接接入协议；
- 同长连接接入层建立、维护长连接，在连接状态异常时，主动触发连接重连，维护端上连接的稳定；
- 转发各业务SDK请求到长连接服务；
- 接受长连接下发的数据并将数据转发给指定的业务SDK。

2.控制层：长连接建联之前的一个前置服务，主要用来验证接入设备的合法性和决定设备的接入策略，其主要职责包括：

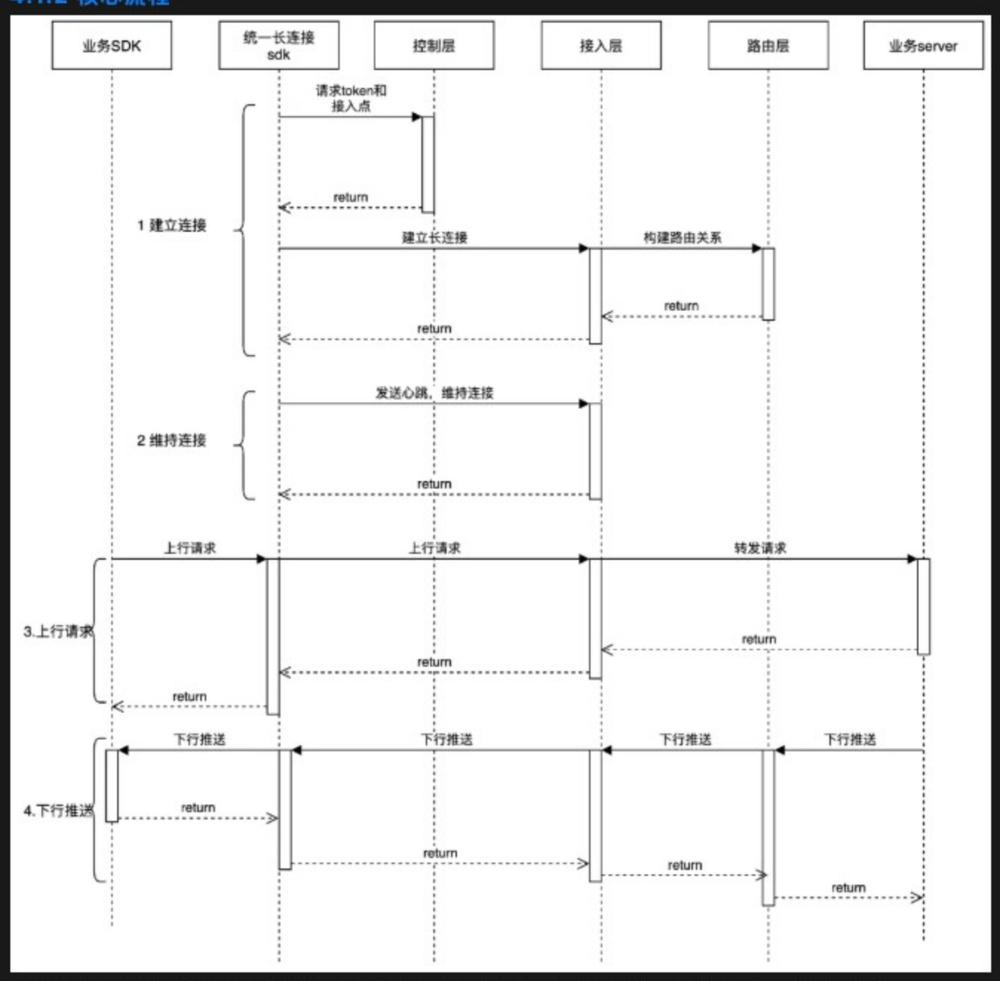
- 生成和验证标识设备合法性的token；
- 根据客户端属性下发对应的接入点；
- 负责小流量控制策略等。

3.接入层：接入层作为统一长连接核心服务，承担了连接介入，连接维护、请求转发、下行推送等主要功能，是长连接核心逻辑和主要压力的承担者，其主要职责包括：

- 对端通讯：负责与长连接SDK建立、维护、释放长连接；
- 连接管理：负责连接管理、构建连接ID->连接信息的映射关系
- 组管理：负责连接组的管理，构建组ID-> 连接信息的映射关系
- 上行转发：接受并转发业务请求到业务后端，接受业务后端的返回，并写回给长连接SDK；
- 下行推送：接受业务的推送请求，写到对应的长连接SDK。

4.路由层：负责构建设备标识和连接标识的映射关系，在业务指定设备标识进行推送的时候，提供设备标识查询连接标识的能力。

4.1.2 核心流程



长连接生命周期内主要有四个核心流程：建立连接、维持连接、上行请求、下行推送。

1.建立连接：由长连接SDK发起，先通过控制层获取该设备的合法标识token和接入配置(接入点、接入协议)，然后和接入层开始建联长连接，成功则长连接建立完成；

2.维持连接：主要是通过长连接SDK定时发起心跳来保证长连接活跃；

3.上行请求：上行请求由业务SDK发起，长连接SDK封装后发送给接入层，接入层根据请求来源发送给指定的业务Server；

4.下行推送：下行推送由业务Server发起，经由路由层根据设备标识确定连接标识，然后将请求转发到对应的接入层，写入到设备指定连接上，经由长连接SDK转发给业务SDK。

4.2 功能实现

4.2.1 连接状态

长连接由于连接生命周期较长，在周期内连接可能会因为各种网络情况、数据传输异常导致连接发生状态变化，同时也为了防止恶意设备模拟正常的客户端对长连接服务进行攻击，需要有一套机制能够让服务端验证长连接状态是合法有效的，同时对于处于异常状态的连接，能够触发其重连并快速恢复。

统一长连接通过引入状态机的方式构建了该机制。即明确定义长连接在生命周期内可能存在的各种状态、每种状态可以触发的操作，以及状态间相关转移的场景等。比如连接在可以发送数据前，需要通过登录验证连接合法有效，认证通过后的连接才能视为有效连接并支持上下行数据传输；登录后的连接如果发生异常情况，比如数据格式异常、网络状态异常，会触发连接失效触发端上重新建联登录等等。引入这种的机制好处是：简化长连接状态流转的开发逻辑，长连接生命周期里面每种状态以及状态间转移关系，触发的操作都是明确定义的，避免了线上因为各种未知原因导致连接处于不可知状态，导致长连接异常甚至无法恢复。

4.2.2 多业务支持

统一长连接一个主要愿景是支持多业务复用一条长连接，即同一条连接上，能够兼容不同业务的数据协议，且在上下行业务数据传输时候能够区分不同业务的请求转发给指定业务。

这个主要是通过长连接私有数据协议来支持，长连接数据协议是长连接SDK和长连接接入层交互使用的数据协议，采用的二进制私有协议，协议主要分为三部分：

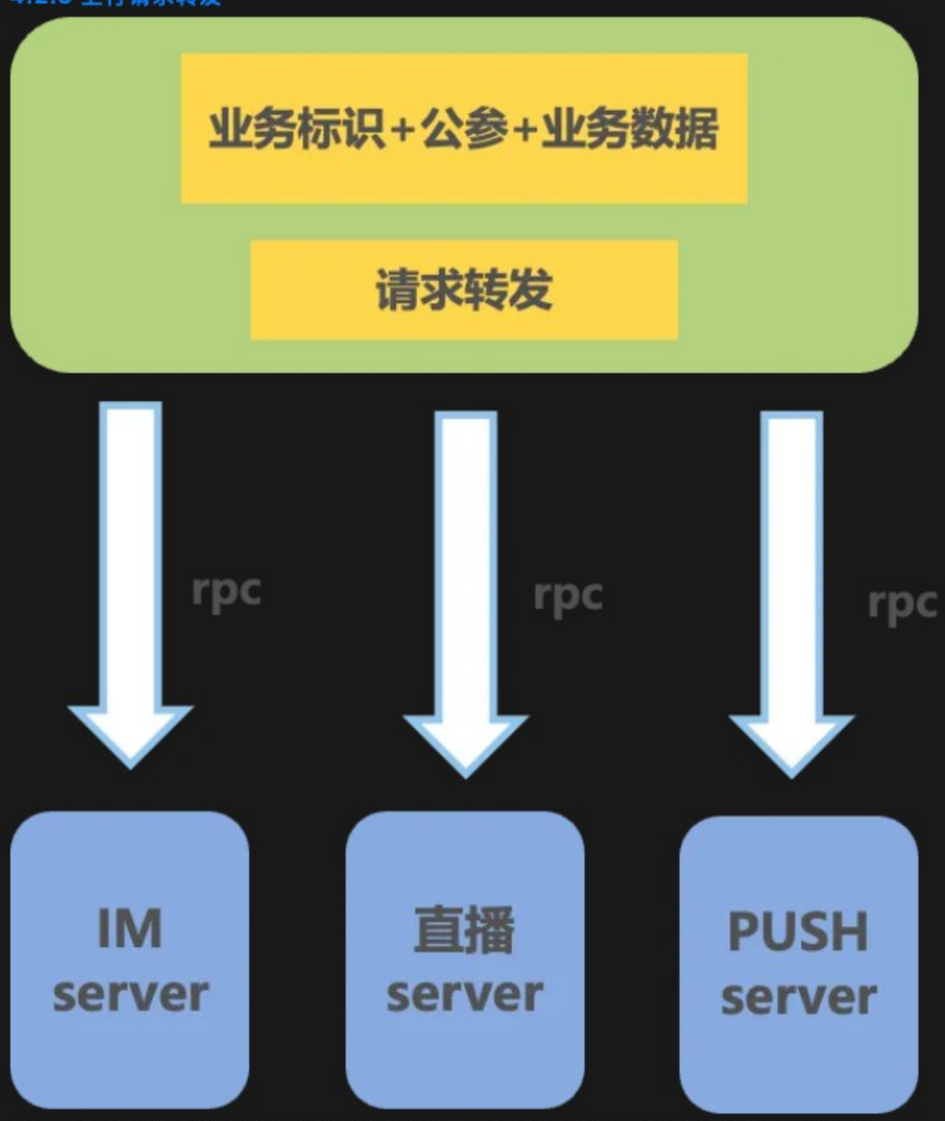
1.**协议头**：包括协议标识、协议版本等；

2.**公参**：设备标识、应用标识、业务标识、请求元数据等；

3.**业务数据**：业务自定义数据，用来兼容不同业务的数据协议。

通过解析协议公参里面的业务标识，长连接SDK和长连接接入层能够确认业务数据对应的业务方，并根据业务标识将请求做对应转发。业务的请求数据放在业务数据部分，协议有业务侧指定，长连接服务只做转发，不介入业务具体细节。

4.2.3 上行请求转发



接入层根据业务标识确认业务数据来源后，会通过RPC请求将业务数据转发给业务server，然后将业务Server的返回写回给端上。上行请求转发除了会转发业务上行请求数据外，也会讲长连接公参数据一并带给业务Server。除此外， 如果业务有诉求，在连接状态发生变化，比如连接断开等，长连接接入层也可以将该信号实时通知业务Server，以便业务Server 根据状态信号变化做进一步操作。

4.2.4 单播&多播推送

下行推送，根据业务需要，主要分为两类，单播推送和组播推送，以下对比了单播推送和组播的差异。

推送方式	推送场景	用户包稳定性	推送频率	实时性要求	实际场景
批量单播推送	给一组用户分批推不同或者相同消息	用户包基本不变，或则变化频率低	中	中(全量用户完全推完分钟级)	PUSH、IM
组播推送	给一组用户推相同的消息	用户包变化快	高	高(秒级)	直播

单播推送：服务端主动推送时候，比如一个业务要给某个设备推消息，由于接入层是多实例部署的，首先需要知道这个设备与哪个长连接实例相连，其次需要知道这个设备与这个实例内哪条长连接相关联，那么这个实例地址和对应的长连接一期就构成了这个设备当时的连接信息，给某个用户推消息，本质上就是通过用户设备找到这个设备对应的连接信息的过程，也就是设备ID -> 连接信息（实例ip+连接ID）映射关系的过程，路由层负责构建和维护设备和连接信息的一个模块。

这个对业务的主要成本是确定需要推送用户的设备ID：

> （1）对于一些业务本身的业务场景是设备维度的，那就可以直接通过接口进行推送；
>
> （2）对于一些业务本身的业务场景是用户维度的，一个用户可以有多个设备，那么业务侧需要做一个用户->设备的映射关系，给用户推消息，需要做用户->设备，然后设备->连接信息的两层转换。

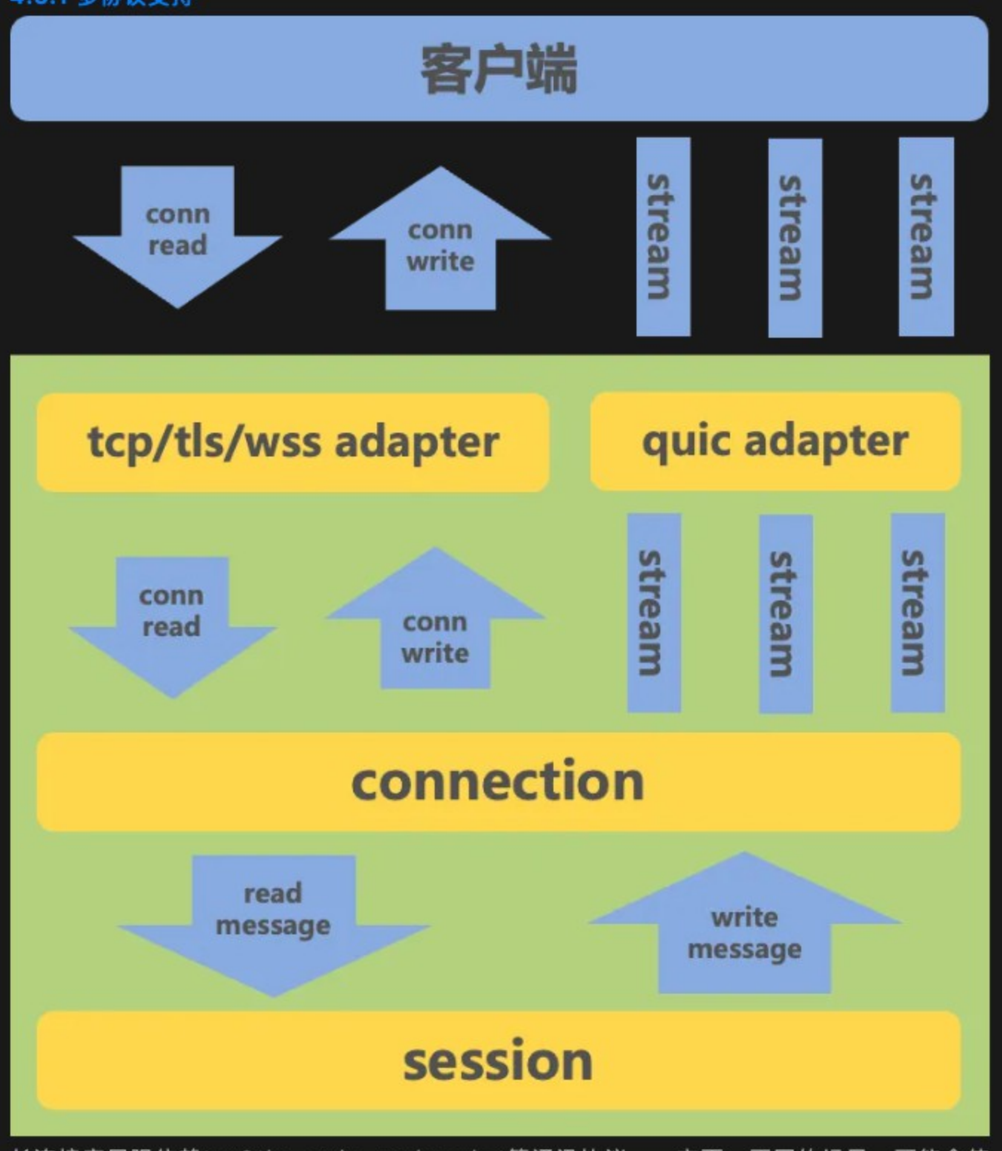
组播推送：此外，由于下行推送的时候，在某些场景下，会存在需要给大批量用户推送相同消息的场景，比如直播(聊天室)。路由层会维护一个连接组信息，连接组ID->组内连接信息的映射。连接组的创建、连接加入和退出连接组，主要由对应的业务场景来控制，路由层只提供对应的接口能力，在连接组建立好后，向对应的连接组推消息，长连接服务会自动将消息拆分发给组内每一条连接。

使用连接组，业务需要做的事情：

> （1）连接组创建；
>
> （2）客户端主动加入和退出连接组；
>
> （3）根据连接ID，推送消息。

4.3 性能优化

4.3.1 多协议支持



长连接底层强依赖tcp&tls、quic、websocket等通讯协议，一方面，不同的场景，可能会使用到不同的通讯协议，比如NA端一般倾向于tcp和tls，小程序和web端倾向于websocket，一方面，现有协议的迭代和新协议的出现，也会给长连接的性能和通道质量带来优化。因此，为了适配不同场景下的通讯协议，同时也为了能够快速探索通讯协议迭代对长连接服务质量的提升。统一长连接做了对不同通讯协议的兼容。

即为了支持多种通讯协议，接入层对连接概念做了划分，将连接分为了两层，connection 和 session。

> **connection层**: 和具体的通讯层协议交互，封装不同通讯协议的接口和逻辑差异，比如tls、websocket、quic 等等。同时给上层session 层提供统一的数据接口，包括连接建立、数据读取、写入、连接信息获取等。新协议的接入，只需要在connection做相应适配，不影响session层的长连接业务逻辑。
>
> **session层**：长连接业务级别连接概念，维护长连接业务连接状态，维护连接状态机，支持请求转发、下行推送等业务逻辑，本身依赖connection层提供的数据接口和实际的通讯协议进行交互。但是不感知具体的通讯协议差异。

同时，端上在建联时，控制层会根据客户端当前的情况，比如端类型(NA、小程序、Web)、当前的网络类型(4G、5G)、设备质量情况等，下发不同的建联协议和对应的接入点，端上根据下发的建议协议和接入点，结合端上自身情况，选用合适的接入协议和接入点进行建联。

这样设计的主要优势在于：

1.长连接业务逻辑和通讯协议做了隔离，通讯协议的更新和新增，不影响长连接状态机、请求转发、下行推送等业务逻辑，简化了兼容多通讯协议的实现难度，实现一套架构支持多通讯协议接入。

2.客户端可以根据实际情况，采用不同的通讯协议接入，通讯协议带来的通道质量优势能够很好的体现在长连接服务质量上。

4.3.2 请求转发组&下行任务组

连接建立完成后，接入层主要面临着三个压力来源：连接维持、请求转发、下行推送。按照单个实例需要支持百万连接来考虑，假定单个连接心跳为1分钟1次，则连接维持需要支持1.6w qps心跳请求；请求转发依赖于业务请求频率，通常百万在线至少需要支持1-2w qps 上行请求；下行推送，根据推送场景的不同，通常组播场景下需要支持5-10w ups下行。综上，单个实例如果要支持百万连接，通常需要支持3-4w qps 上行，5-10w 的ups下行。
统一长连接层接入层服务是使用golang来实现的，按照golang 常用的网络模型，一条连接会有对应两个goroutine，一个用来读数据和处理数据，一个用来写数据。这个模型存在两个问题：

1.统一长连接是多业务复用一个连接，连接上会存在同时有多个请求上行，一个goroutine读和处理数据，如果一个请求处理比较慢，会导致后续其他请求处理排队的情况；

2.每个连接至少需要2个goroutine,如果单实例需要支持一百万连接，则单个实例常态下会有200万goroutine, 而且，长连接服务，连接的建立和释放非常频繁，这个会导致goroutine的建联和释放也非常频繁，进而给服务的gc带来巨大的压力。

统一长连接通过引入请求转发组和下行任务组来解决上述两个问题。

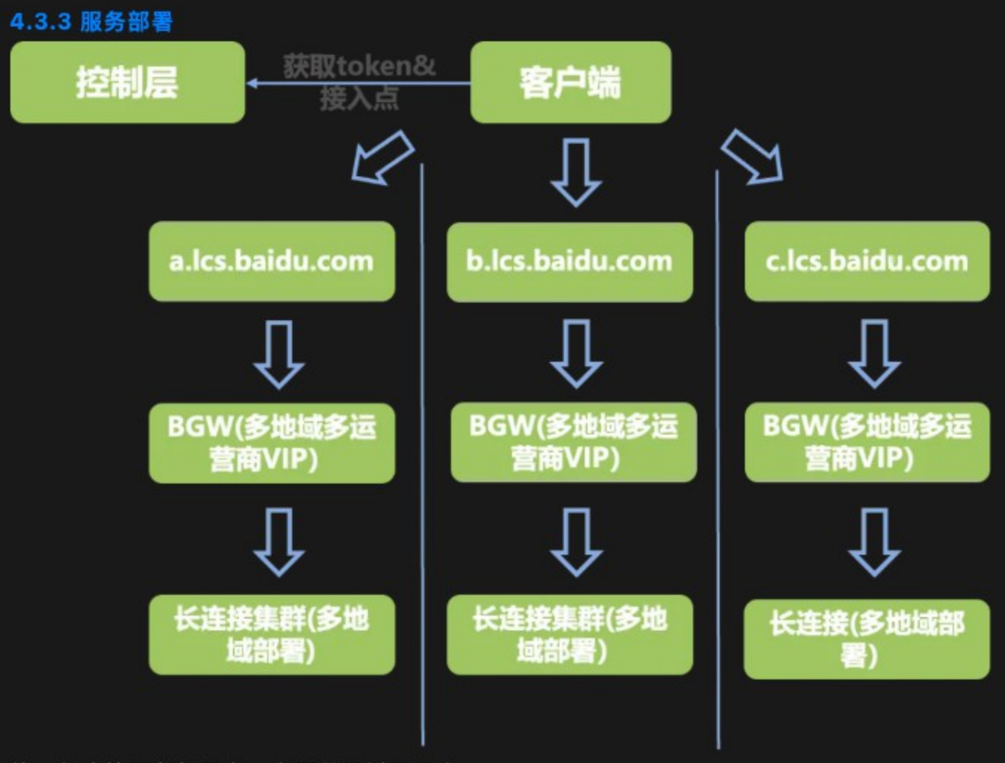
1、请求转发组

接入层在实例启动的时候，会根据支持的业务上行qps，初始化对应的请求转发组，连接在读取完请求数据后，会根据请求属于那个业务将请求转给对应的请求转发组，由请求转发组内的goroutine完成后续请求；不同业务的请求转发组是不同的goroutine池，避免业务间相互影响。连接本身的读goroutine只是负责读取数据，转发请求给请求转发组，避免请求处理存在排队的情况。

2、下行任务组

下行数据写入的时候，会有一个公共的任务组，每个连接在任务组中会有一个固定的处理协程，有数据下行的时候会讲写入数据的任务发到下行任务组。这个主要目的是，由于单个实例需要维护大量的在线长连接，每个长连接通常需要两个协程，一个读协程、一个写协程，如果单实例支持50w连接，也就会单实例存在百万协程，这个对服务gc和资源都会造成一定的压力。同时，一个实例维持的所有连接通常不会同时都下行写消息，因而，可以通过维护一个下行任务组，任务组里面维护动态数量的协程数，每个连接绑定任务组里面一个协程，有下行任务时将任务发送到任务组里面，有任务组里面的协程负责将任务下行，减少实例服务压力。

4.3.3 服务部署



统一长连接服务部署上，主要做了以下几点：

1.长连接在国内三大运营商的华东、华北、华南地域均部署了接入点；部分业务需要支持海外业务，增加了香港机房的独立接入点入口；

2.根据业务量级和重要性，分大小集群部署，每个集群对应不同的域名，不同业务通过控制层下发域名分流到对应集群。主要目的针对于重点业务，提供独立部署的能力；对于次级业务，提供混部服务，降低成本和提高资源利用率；

3.针对接入层每个实例，将实例的配置和能够支持的连接数，控制在10w-20w量级，避免单个实例支持连接数过多，实例发生服务抖动时，对整个集群服务产生较大影响；单个实例支持的连接数有限制，减少实例内常态维护的goroutine数，减轻gc压力。

4.4 业务接入

业务接入统一长连接通常涉及以下几个步骤：

1.评估需要接入的能力：评估需要接入的长连接能力，比如下行推送中，是要接批量单播还是组播；是否要支持上行请求等，根据接入能力不同，需要对接不同的接口；

2.评估用户量级：用户量级涉及到资源评估以及是否需要单独进行集群部署等等；

3.端上接入长连接SDK：客户端需要接入长连接SDK，接入上下行收发消息接口；

4.服务端接入上下行接口：服务端需要根据接入能力，适配不同的服务端接口；

5.申请资源服务上线。

05
GEEK TALK
总结与规划

5.1 总结

目前统一长连接已支持了千万级长连接并发在线，支持过百万级ups 批量单播和组播消息下行，拥有实时横向扩容能力。服务第一次完成上线至今，长连接服务稳定，经历过多次重大活动高并发推送的考验，没有出现过任何影响其他业务服务质量的case。总的来说，统一长连接项目从立项、开发到最后上线运维，服务质量整体上是符合预期的。从统一长连接项目整个项目流程，主要总结有以下三点经验：

1.需求：分析需求时候，要明确需求和业务的边界，也就是明确什么应该长连接的能力，什么是业务逻辑，坚持长连接服务不深入介入业务逻辑这个原则，保证长连接服务与业务的逻辑解耦，确保长连接服务结构的稳定。

2.设计：需求明确的情况下，技术的方案的选择以简单满足要求为优先，长连接服务本身逻辑并不复杂，其主要在于服务的稳定性和高性能，巧妙而复杂的方案在长连接的场景下，并不一定能够适用；

3.运维：追求单实例性能的同时，也要在性能和运维之间做取舍，单实例性能再强，能够支持的连接数再多，通常也比不上分拆多个小实例，带来的稳定性和资源利用率的提升大。

5.2 规划

统一长连接服务经历数次迭代后，目前基本功能已经趋于稳定，进一步对长连接服务进行改善和优化，主要集中在以下几个方向：

1.精细化：进一步完善长连接全链路网络质量数据统计和分析能力的建设。

2.智能化：端上建联、接入点接入、心跳频率等能够根据实际环境进行自动调整；

3.场景拓展：探索长连接支持更多的业务场景。

END

推荐阅读

百度搜索Push个性化：新的突破
数据交付变革：研发到产运自助化的转型之路
百度搜索exgraph图执行引擎设计与实践
百度搜索&金融：构建高时效、高可用的分布式数据传输系统
“踩坑”经验分享：Swift语言落地实践

一键三连，好运连连，bug不见👇

#长连接 3　#golang 2　#高并发 1

长连接 · 目录

〈上一篇 · 百度IOS端长连接组件建设及应用实践