# 一、通信系统方案整体架构

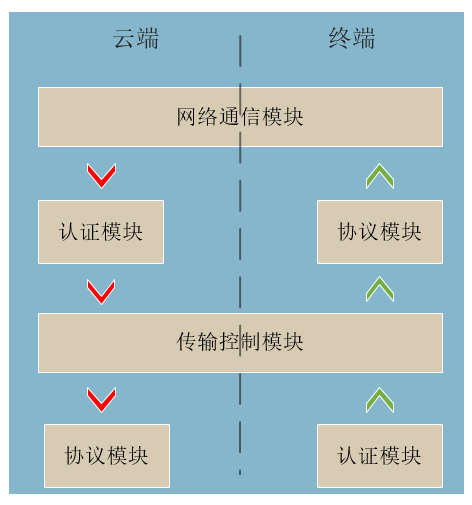
## 1.需求背景

借助5G高带宽、低延时的特性，结合云端大容量、高性能的运算及存储资源，终端的运算和存储能力逐渐迁移到云端，本地和云端只需进行交互，便可实现远程操控云端系统，畅想云端无限存储、计算资源的效果。

通信系统是云侧和端侧协同的关键技术，同时可为云手机、云游戏、云办公等场景赋能，提供更流程、更安全、体验更佳的5G云服务。

## 2.整体结构

整体结构图，概述方案。



如上图所示，本方案由客户端和服务端两个整体部分，而这两个部分由分别由四个模块组成，分别为：网络通信模块，认证模块，传输控制模块，协议模块。

**网络通信模块：**此部分是端云交互的基础，所有的数据都通过这个模块来实现安全可靠的传递，另外网络传输效率有时候会变得很低，那么此模块同时也是具有做相应的拥塞控制的策略的。

**认证模块：**有了数据，就要保证数据发送的安全性，以及用户的管控，那么此模块就是用来协定端云双方是否都是拥有使用资格的。

**传输控制模块：**由于网络状态的不确定，有时当前的传输链路顺畅，有时就会变得很拥塞，虽然在网络通信模块里面已经包含了相关的网络拥塞控制，但是当拥塞持续发生时，中间过程的一些优化就显得捉襟见肘了，需要从根源上来减少链路中数据包的输入，以缓解链路拥塞带来的一些列迟滞性问题。

**协议模块：**为了让端云更好更有效的行使能力，那么就需要一系列既定的协议来应对各种不同的问题和处理不同的事物。那么此模块就是来保证端云更好更快的运行的。

# 二、技术调研

现有的拥塞控制方案：

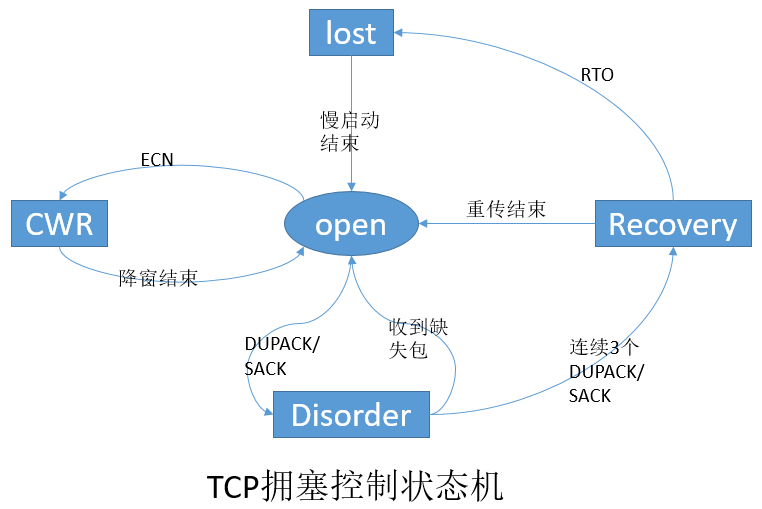
TCP拥塞控制：

TCP拥塞控制有两个重要的方法：流量控制、阻塞控制。

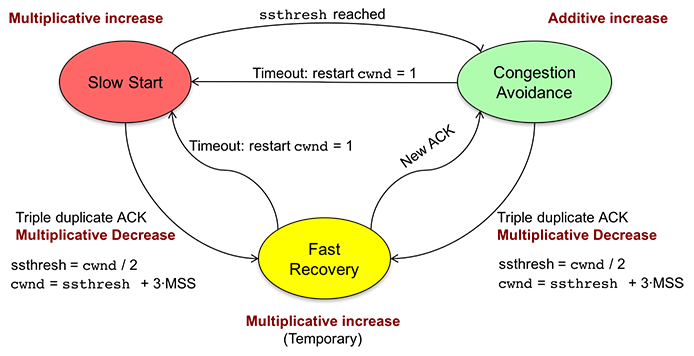
TCP协议通过滑动窗口来进行流量控制，它是控制发送方的发送速度从而使接受者来得及接收并处理。而拥塞控制是作用于网络，它是防止过多的包被发送到网络中，避免出现网络负载过大，网络拥塞的情况。

在阻塞控制上有四种控制算法：慢启动，拥塞避免，拥塞发生时和快速恢复。

状态机与控制算法转换框图如下：



算法转换框图：



慢启动：

所谓慢启动，也就是TCP连接刚建立，一点一点地提速，试探一下网络的承受能力，以免直接扰乱了网络通道的秩序。整个过程如下：

1) 初始化拥塞窗口cwnd大小为1，即可以传一个MSS大小的数据。

2) 每当收到一个ACK，cwnd大小加一，呈线性上升。

3) 每当过了一个往返延迟时间RTT，cwnd大小直接翻倍，乘以2，呈指数上升。

4) 还有一个ssthresh（slow start threshold），是一个指数上升的上限，当cwnd >= ssthresh时，就会进入“拥塞避免”

拥塞避免

如上面最后一步，当拥塞窗口大小cwnd大于等于慢启动阈值ssthresh后，就进入拥塞避免。过程如下：

1) 收到一个ACK，则cwnd = cwnd + 1 / cwnd

2) 每当过了一个往返延迟时间RTT，cwnd大小加一。

 过了慢启动阈值后，拥塞避免算法可以避免窗口增长过快导致窗口拥塞，而是缓慢的增加调整到网络的最佳值。

拥塞发生时

TCP拥塞控制默认认为网络丢包是由于网络拥塞导致的，所以一般的TCP拥塞控制算法以丢包为网络进入拥塞状态的信号。对于丢包有两种判定方式，一种是超时重传RTO[Retransmission Timeout]超时，另一个是收到三个重复确认ACK。

 超时重传是TCP协议保证数据可靠性的一个重要机制，其原理是在发送一个数据以后就开启一个计时器，在一定时间内如果没有得到发送数据报的ACK报文，那么就重新发送数据，直到发送成功为止。

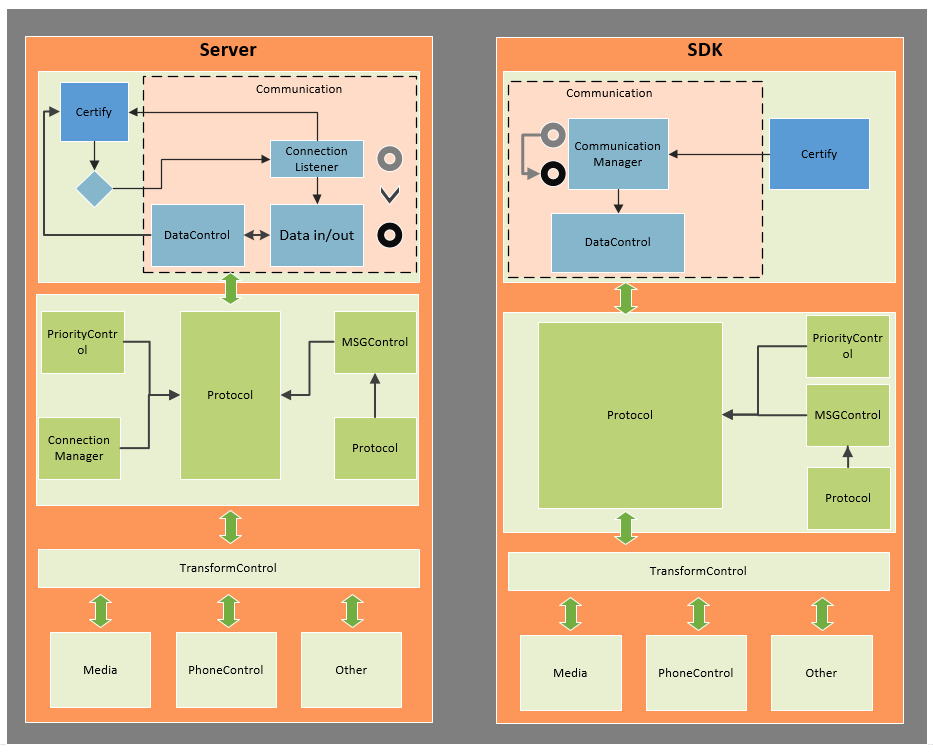
 如果发送端接收到3个以上的重复ACK，TCP就意识到数据发生丢失，需要重传。这个机制不需要等到重传定时器超时，所以叫做快速重传，而快速重传后没有使用慢启动算法，而是拥塞避免算法，所以这又叫做快速恢复算法。

快速恢复

 最为早期的TCP Tahoe算法就只使用上述处理办法，但是由于一丢包就一切重来，导致cwnd又重置为1，十分不利于网络数据的稳定传递。

 所以，TCP Reno算法进行了优化。当收到三个重复确认ACK时，TCP开启快速重传Fast Retransmit算法，而不用等到RTO超时再进行重传。

# 三、实现方案



如上图，为本系统的整体架构图，由上文分为四个模块：网络通信模块，认证模块，传输控制模块，协议模块。

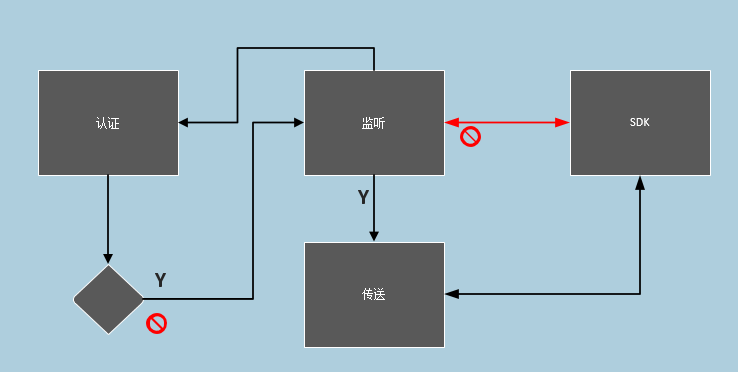
## 3.1网络通信模块

此模块分为两个部分，一个为数据传输，另一个为拥塞估计。数据传输为保证数据高效安全得在两边传输，拥塞估计为传输控制模块做前期准备。

### 3.1.1数据传输

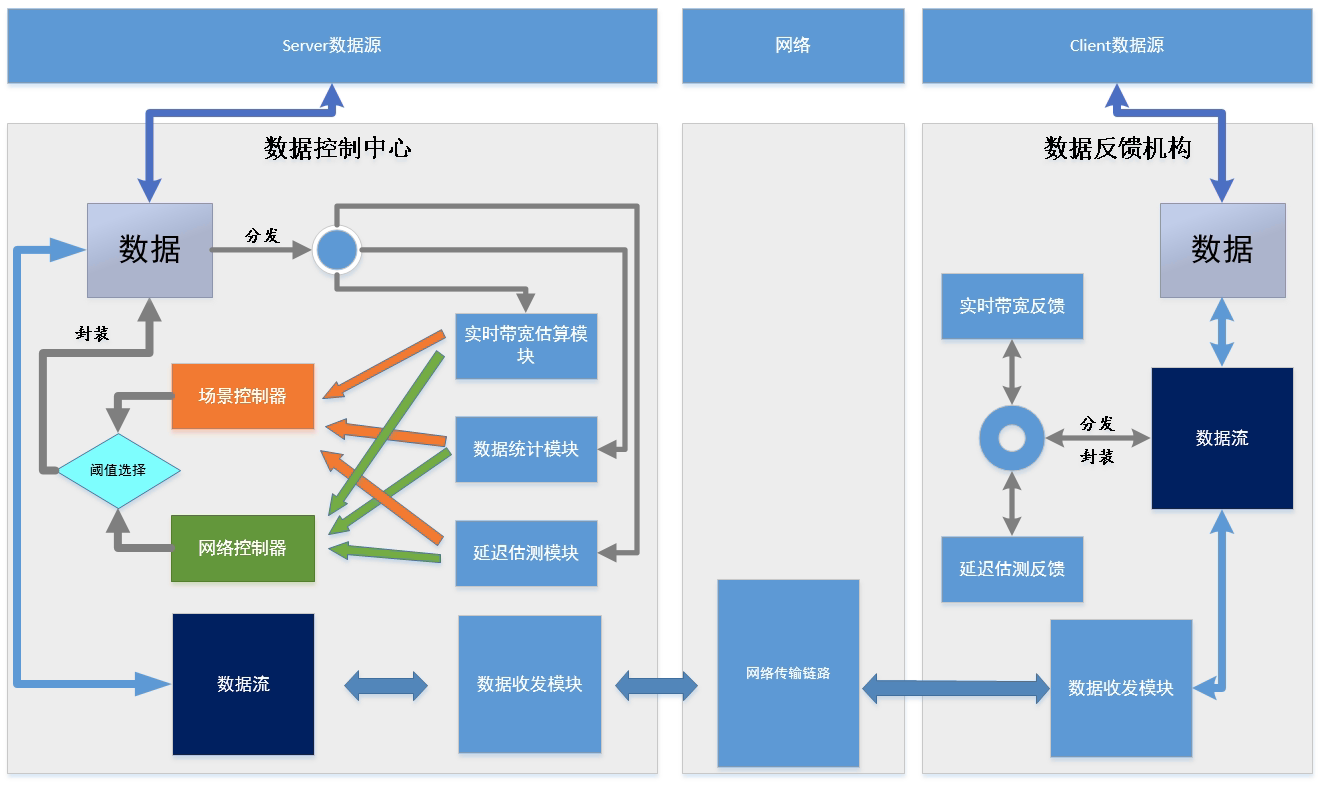
此部分又有两个模块组成。一个为常驻内存的TCP监听，另一个为数据传送。最开始由SDK发送TCP的链接请求并把相应的认证信息发送出去，也就是图中灰色的圆圈，当服务端收到请求时拿到SDK的认证信息时，传送给服务端的认证模块，如果认证通过，则回复SDK链接成功的标志，并把此链接转给数据传送模块，接着自己继续回归TCP监听状态以备新的链接能够得到及时的应答。如果认证失败，则会立即中断此链接，同样的也会回归监听状态。

数据传送依赖TCP相关的拥塞控制算法，以此来保证数据传输的及时性，如下图。



### 3.1.2拥塞估计

整个云手机运行过程中很多方面都会或深或浅的影响它的整体效果或者端到端的时延，那么在拥塞估计方面就显得举足轻重了。整个框架如图所示：



其中：实时带宽估计模块和实时带宽反馈互为系统，延迟估计模块和延迟估计反馈一起。数据统计为独立的模块。

1）实时带宽估计：

此模块作用是提供当前IP链路的实时传输带宽预测值，每秒更新一次。所谓实时传输带宽就是当前链路单位时间内的带宽总消耗。假如说当前在图中的数据区单位时间内产生了1M字节的数据，只要这些数据在下一个单位时间里面全部传送完毕，那么就说当前链路的实时传输带宽为1M字节，但是如果只传送了800K字节的数据，那么就只有800K字节。

2）延迟估测：

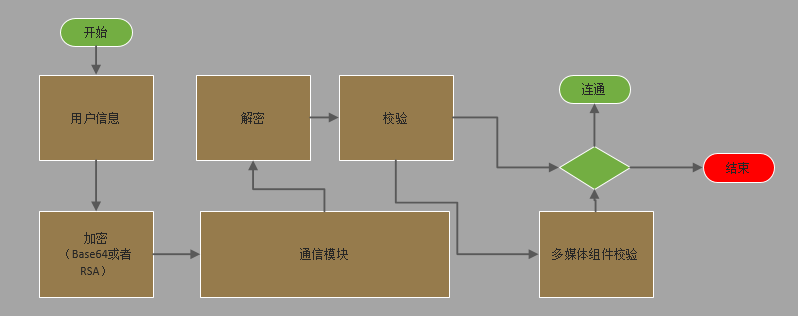
系统会数据流中的每一个数据包上做上标记，当延迟反馈端收到这个标记的时候则会就近把反馈标记加到即将要发送到服务器的数据中，以此来立即响应服务器的延迟估算标记。最后服务器收到反馈就会通过时差来最终确定传输延迟。

3）数据统计：

此模块作用为统计所有占用网络资源的数据。

## 3.2认证模块

这一部分用来做用户校验和连接校验的，如下图：



SDK需要连接Server的时候，首先把自己的用户信息通过Base64或者RSA加密借助通信模块发送到Server端。当server收到信息是，做反向动作：解密，校验，当校验成功，并且用户当前的链接不是多媒体组件（注：SDK想要访问Server都必要先经历初次访问），那么就会把用户信息存入Server，并开始正常访问Server。如果用户的当前连接为多媒体组件的链接，那么就需要和之前成功访问并且存下来的数据做校验，只要校验通过才可以开启多媒体组件。

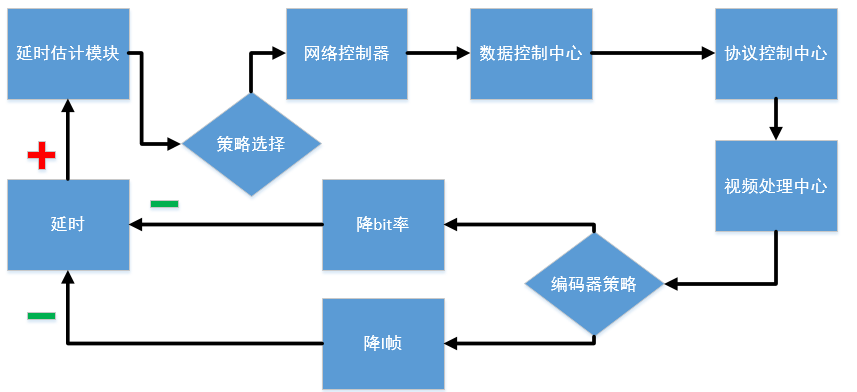
## 3.3传输控制模块

此模块是针对网络通信组件出现拥塞以及业务优先级来确定的。

### 3.3.1拥塞处理

触发此部分那么就说明当前网络传输效率已经不能满足当前的数据传输需求了，那么就需要启动相应的策略来处理当前拥塞状况。

首先需要从网络通信模块中获取当前的拥塞状态。当现在的拥塞状态可以触发拥塞处理时，那么整个的处理逻辑如下所示：



由此当拥塞处理被触发的时候，通过延迟模块，可以得到当前实时的传输延时，当延迟大于某个阈值的时候，那么就说明当前链路中某地方发生了拥堵，那么此时就需要告诉数据控制中心需要降低本链路的带宽，数据中心会把拥塞状态发给协议控制中心，协议控制中心要求视频处理中心降低视频输出带宽，视频处理中心通过动态调整编码参数降低码率或减少I帧输出。

### 3.3.2业务优先级

因为云手机是个复杂的系统，整个端到端数据中存在各种业务，但是并不是每个业务都是需要及时传输的，那么由此产生了业务的优先级。因为使用场景的不同，可以把业务大体归类到如下几个场景中：

控制 > 音视频 > 文件

其中控制包含：Touch，Sensor，GPS。音视频就是有Android相关的视图和音频业务。文件包含：文件同步，Camera。由此，在每个业务上打上标签，同一场景中各业务的优先级一样，不同场景按照分每秒产生的数据排序。即每一秒都会把新产生的数据按照优先级排序，已经排好的数据（前一秒的数据）则不再变更传输优先级。

## 3.4协议模块

为了让端云更好更有效的行使能力，那么就需要一系列既定的协议来应对各种不同的问题和处理不同的事物。那么此模块就是来保证端云更好更快的运行的。

也就是通信双方都能清楚的知道对方所要表达的是什么，想要做什么。其次在保证沟通的前提下，尽可能让传输的数据小。