

13 | 在WebRTC中如何控制传输速率呢？

2019-08-13 李超

从0打造音视频直播系统

[进入课程 >](#)



讲述：李超

时长 14:40 大小 13.45M



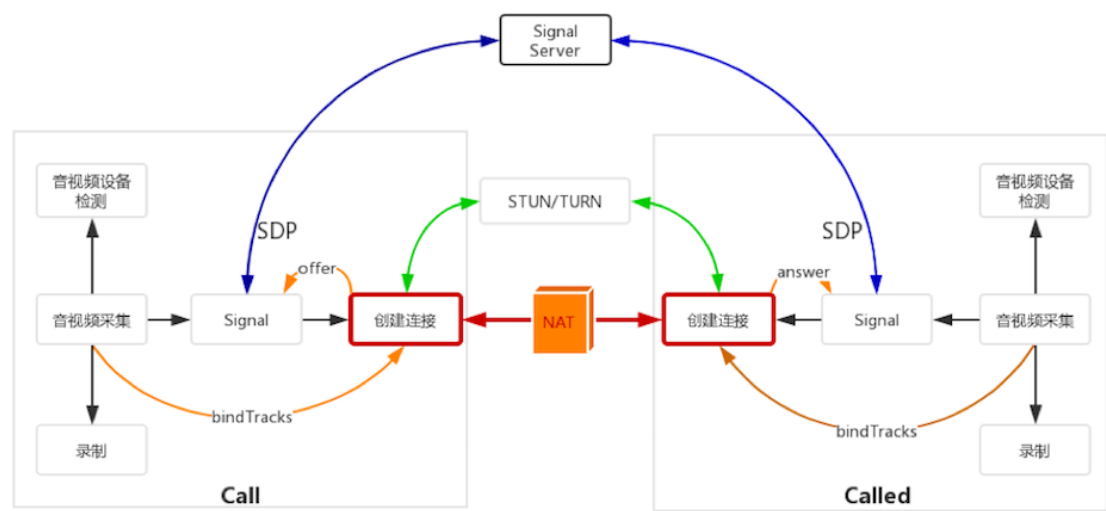
在上一篇 [《12 | RTCPeerConnection：音视频实时通讯的核心》](#) 一文中，我向你介绍了 RTCPeerConnection 对象是如何在端与端之间建立连接的，以及音视频数据又是如何通过它进行传输的。而本文则更进一步，向你介绍如何使用 RTCPeerConnection 来控制音视频数据的传输速率。

通过 RTCPeerConnection 进行传输速率的控制实际上还是蛮简单的一件事儿，但在学习相关知识的时候，你不仅要能知其然，还要知其所以然。对于本文来讲，就是你不但要学习如何控制传输速率，同时还应该清楚为什么要对传输速率进行控制。

其实，之所以要进行传输速率的控制，是因为它会对**音视频服务质量**产生比较大的影响，对于音视频服务质量这部分知识，接下来我就与你一起做详细探讨。

在 WebRTC 处理过程中的位置

在此之前，我们依旧先来看看本文的内容在整个 WebRTC 处理过程中的位置。



WebRTC 处理过程图

通过上图你可以知道，本文所讲的内容仍然属于**传输**的范畴。

音视频服务质量

像上面所说的，虽然通过 `RTCPeerConnection` 在端与端之间建立连接后，音视频数据可以互通了，但你还应对传输速率有所控制。之所以要对传输速率进行控制，主要是为了提高音视频服务质量。

举个简单的例子，假设你的带宽是 1Mbps，你想与你的朋友进行音视频通话，使用的视频分辨率为 720P，帧率是 15 帧 / 秒，你觉得你们通话时的音视频质量会好吗？

咱们来简单计算一下，根据经验值，帧率为 15 帧 / 秒、分辨率为 720P 的视频，每秒钟大约要产生 1.2 ~ 1.5Mbps 的流量。由此可知，你和你朋友在 1M 带宽的网络上进行通话，那通话质量一定会很差。因为你的“马路”就那么宽，却要跑超出它宽度的数据，这样超出带宽的数据会被直接丢弃掉，从而造成大量视频帧无法解码，所以最终效果一定会很差。

由此可知，如果你不对音视频传输速率进行限制的话，它一定会对音视频服务质量产生严重的影响。除了传输速率，还有哪些因素会对音视频质量产生影响呢？下面我从网络质量和数据两个方面列举了一些对音视频服务质量产生影响的因素：

网络质量，包括物理链路的质量、带宽的大小、传输速率的控制等；

数据，包括音视频压缩码率、分辨率大小、帧率等。

以上这些因素都会对音视频的服务质量产生影响。有这么多因素会对音视频服务质量产生影响，那在真实的场景中，你该怎么去区分它们呢？或者说怎么判断服务质量是不是由于传输速率问题引起的呢？

要想判断出是哪些因素引起的音视频服务质量变差，你就必须要知道这些因素的基本原理，下面我们就简要地对这些因素做些介绍。

1. 物理链路质量

物理链路质量包括三个方面，即丢包、延迟和抖动。下面我们来看看它们是怎样影响服务质量的吧！

丢包。这个比较好理解，如果物理链路不好，经常出现丢包，这样就会造成接收端无法组包、解码，从而对音视频服务质量产生影响。

延迟。指通信双方在传输数据时，数据在物理链路上花费的时间比较长。对于实时通信来说，200ms 以内的延迟是最好的，这样通话双方的感觉就像是在面对面谈话；如果延迟是在 500 ms 以内，通话双方的体验也还不错，有点像打电话的感觉；如果延迟达到 800ms，还能接受，但有明显的迟滞现象；但如果延迟超过 1 秒，那就不是实时通话了！

抖动。指的是数据一会儿快、一会儿慢，很不稳定。如果不加处理的话，你看到的视频效果就是一会儿快播了、一会儿又慢动作，给人一种眩晕的感觉，时间长了会非常难受。不过对于 WebRTC 来讲，它通过内部的 JitterBuffer（可以简单地理解为一块缓冲区）就能很好地解决该问题。

2. 带宽大小

带宽大小指的是每秒钟可以传输多少数据。比如 1M 带宽，它表达的是每秒钟可以传输 1M 个 bit 位，换算成字节就是 $1\text{Mbps}/8 = 128\text{KBps}$ ，也就是说 1M 带宽实际每秒钟只能传输 128K 个 Byte。

当带宽固定的情况下，如何才能让数据传输得更快呢？**答案是充分利用带宽**。这句话有点抽象，它实际的含义是**把带宽尽量占满，但千万别超出带宽的限制**。这里还是以 1M 带宽为

例，如果每秒都传输 1M 的数据，这样传输数据的速度才是最快，多了、少了都不行。每秒传输的数据少了，就相当于有 100 辆车，本来每次可以走 10 辆，10 趟就走完了，可你却让它一次走 1 辆，这样肯定慢；而每秒传输多了，就会发生网络拥塞，就像每天上下班堵车一样，你说它还能快吗？

3. 传输速率

在实时通信中，与传输速率相关的有两个码率：音视频压缩码率和传输控制码率。

音视频压缩码率指的是单位时间内音视频被压缩后的数据大小，或者你可以简单地理解为压缩后每秒的采样率。它与视频的清晰度是成反比的，也就是**压缩码率越高，清晰度越低**。我们可以做个简单的对比，你应该清楚音视频编码被称为**有损压缩**，所谓的有损压缩就是数据被压缩后，就无法再还原回原来的样子；而与有损压缩对应的是**无损压缩**，它是指数据解压后还能还原回来，像我们日常中用到的 Zip、RAR、GZ 等这些压缩文件都是无损压缩。对于有损压缩，你设备的压缩码率越高，它的损失也就越大，解码后的视频与原视频的差别就越大。

传输码率是指对网络传输速度的控制。举个例子，假设你发送的每个网络包都是 1500 字节，如果每秒钟发 100 个包，它的传输码率是多少呢？即 $100 \times 1.5K = 150K$ 字节，再换算成带宽的话就是 $150KB \times 8 = 1.2M$ 。但如果你的带宽是 1M，那每秒钟发 100 个包肯定是多了，这个时候就要控制发包的速度，把它控制在 1M 以内，并尽量地接近 1M，这样数据传输的速度才是最快的。

当然，如果你的压缩码率本来就很小，比如每秒钟只有 500kbps，而你的带宽是 1Mbps，那你还有必要对传输码率进行控制吗？换句话说，一条马路可以一起跑 10 辆车，但你现在只有 3 辆，显然你就没必要再控制同时发车的数量了。

4. 分辨率与帧率

你应该很清楚，视频的**分辨率**越高，视频就越清晰，但同时它的数据量也就越大。我们还是来简单计算一下，对于 1 帧未压缩过的视频帧，如果它的分辨率是 1280×720 ，存储成 RGB 格式，则这一帧的数据为 $1280 \times 720 \times 3 \times 8$ （3 表示 R、G、B 三种颜色，8 表示将 Byte 换算成 bit），约等于 22Mb；而存成 YUV420P 格式则约等于 11Mb，即 $1280 \times 720 \times 1.5 \times 8$ 。

按照上面的公式计算，如果你把视频的分辨率降到 $640 * 360$ ，则这一帧的数据就降到了原来的 $1/4$ ，这个效果还是非常明显的。所以，**如果你想降低码率，最直接的办法就是降分辨率。**

当然，对**帧率**的控制也一样可以起到一定的效果。比如原来采集的视频是 30 帧 / 秒，还以分辨率是 $1280 * 720$ 为例，之前 1 帧的数据是 22M，那 30 帧就是 $22 * 30 = 660\text{Mb}$ 。但如果改为 15 帧 / 秒，则数据就变成了 330Mb，直接减少了一半。

但了解音视频压缩原理的同学应该知道，通过减少帧率来控制码率的效果可能并不明显，因为在传输数据之前是要将原始音视频数据进行压缩的，在同一个 GOP (Group Of Picture) 中，除了 I/IDR 帧外，B 帧和 P 帧的数据量是非常小的。因此，**减少帧率的方式就没有降低分辨率方式效果明显了。**

到这里我已经向你介绍了，在 WebRTC 实时通信中对音视频质量产生影响的因素有哪些，以及这些因素对音视频服务质量产生影响的基本原理，了解这些原理会更有利于你判断音视频服务质量变差的原因，从而决定是否要使用控制传输速率的方法来解决音视频服务质量的问题。

那接下来我们言归正转，看一下在 WebRTC 中具体该如何控制传输速率。


传输速率的控制

通过上面的介绍，我想你现在应该很清楚，**可以通过以下两种方式来控制传输速率**。第一种是通过**压缩码率**这种“曲线救国”的方式进行控制；第二种则是更直接的方式，通过控制**传输速度**来控制速率。

第二种方式虽说很直接，但是也存在一些弊端。假设你有 10M 的数据要发送，而传输的速度却被限制为 5kbps，那它就只能一点一点地传。**需要注意的是，由于 WebRTC 是实时传输，当它发现音视频数据的延迟太大，且数据又不能及时发出去时，它会采用主动丢数据的方法，以达到实时传输的要求。**

所以说控制传输速率虽然有两种方式，但实际上，WebRTC 只允许我们使用第一种压缩码率的方式来主动控制速率，而第二种方式是它在底层自己控制的，为了保障实时性，一旦数据无法及时发送出去的话就会进行主动丢包。

了解了上面这些知识之后，下面我们就来看看在 WebRTC 下如何才能控制视频流的码率。具体操作如下面代码所示：

 复制代码

```
1 ....
2
3 var vsender = null; // 定义 video sender 变量
4 var senders = pc.getSenders(); // 从 RTCPeerConnection 中获得所有的 sender
5
6 // 遍历每个 sender
7 senders.forEach( sender => {
8     if(sender && sender.track.kind === 'video'){ // 找到视频的 sender
9         vsender = sender;
10    }
11 });
12
13 var parameters = vsender.getParameters(); // 取出视频 sender 的参数
14 if(!parameters.encodings){ // 判断参数里是否有 encoding 域
15     return;
16 }
17
18 // 通过 在 encoding 中的 maxBitrate 可以限掉传输码率
19 parameters.encodings[0].maxBitrate = bw * 1000;
20
21 // 将调整好的码率重新设置回 sender 中去，这样设置的码率就起效果了。
22 vsender.setParameters(parameters)
23     .then(()=>{
24         console.log('Succesed to set parameters!');
25     }).catch(err => {
26         console.error(err);
27     })
28
29 ...
```

上面的代码中，首先从 RTCPeerConnection 中获取视频的发送者，即 kind 为 video 的 sender；然后取出 sender 中的 parameters 对象，其中的 maxBitrate 属性就是用于控制传输码率的；将你期望的最大码率设置好后，再将 parameters 对象设置回去，这样 WebRTC 就可以控制某路流的码率大小了。

通过上面的代码你还可以看出，在 WebRTC 中速率的控制是使用压缩码率的方法来控制，而不是直接通过传输包的多少来控制的。从另外一个角度你也可以得到这样的结论，因为 maxBitrate 属性来自于 sender 的 encoding 对象，而 encoding 对象就是进行编码时使用的参数。

小结

通过本文的讲解，你应该可以看出在 WebRTC 中控制传输速率其实是非常简单的事情，只要向 `RTCPeerConnection` 中的 `sender` 设置一个最大码率就可以控制某一路流的传输速率了。

另外，在本文中我还向你简要介绍了物理链路的质量、带宽的大小、码率、分辨率和帧率这几个影响音视频服务质量的重要因素，并详细分析了每个因素是如何影响到音视频服务质量的。

这里需要注意的是，在 WebRTC 中，通过上述的方式只能对每一路音视频流进行码率的控制，而不能进行整体的统一控制。所以，如果你的应用同时存在多路音视频流，而你又想控制一个总的码率，就只能一路一路地控制了。

思考时间

实际上，在 WebRTC 中除了通过 `sender` 的 `maxBitrate` 控制码率外，还可以通过 SDP 来控制传输速率，你是否可以找到这个方法呢？

欢迎在留言区与我分享你的想法，也欢迎你在留言区记录你的思考过程。感谢阅读，如果你觉得这篇文章对你有帮助的话，也欢迎把它分享给更多的朋友。



从 0 打造音视频直播系统

手把手教你打造实时互动音视频直播系统

李超

新东方音视频直播技术专家
前沪江音视频架构师



新版升级：点击「 请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

上一篇 12 | RTCPeerConnection：音视频实时通讯的核心

下一篇 14 | 如何打开/关闭音视频？

精选留言 (8)

写留言



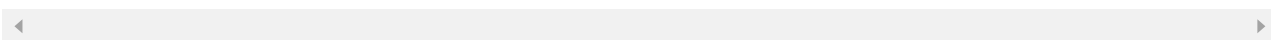
刘丹

2019-08-13

```
senders.forEach( sender => {  
  if(sender && sender.track.kind === 'video'){ // 找到视频的 sender  
    vsender = sender;  
  }  
});...
```

展开 ∨

作者回复: 这里只是一个例子，一般情况下，一个客户端只有一路视频。



1



tommy_zhang

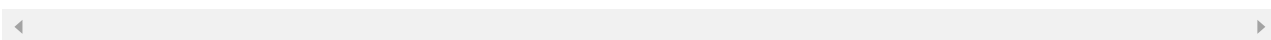
2019-08-16

```
m=video 9 UDP/TLS/RTP/SAVPF 96 97 98 99 100 101 127  
c=IN IP4 0.0.0.0  
b=AS:500
```

老师好，在android端我修改SDP,添加了b=AS:500，带宽没有限制住。是什么原因？

展开 ∨

作者回复: 不同的分辨率有最低码流，如果低于这个值也是不行的



tommy_zhang

2019-08-16

我在android端，在SDP中添加b=AS:500，带宽没有限制住。m=video 9 UDP/TLS/RTP/SAVPF 96 97 98 99 100 101 127

c=IN IP4 0.0.0.0

b=AS:500

展开 ▾



诸葛亮了

2019-08-16

怎样能保证分辨率保持不变呢

展开 ▾

作者回复: 分辨率设置好后, 如果你不主动改变它, 它一直都不会变



鼠辈

2019-08-15

也就是说 1M 带宽实际每秒钟只能传输 128K 个 Byte。应该是125吧

作者回复: 是 128K , $1\text{Mbit} = 1024\text{Kbit} = 1024/8 \text{ KByte} = 128\text{KByte}$



诸葛亮了

2019-08-14

在 3.传输速率 下有这么一句话“当然, 如果你的压缩码率本来就很小, 比如每秒钟只有 500kb...” 。当中的“如果你的压缩码率本来就很小” 是不是应该是“如果你的传输码率本来就很小” 啊?

作者回复: 是压缩码率, 压缩码率小代表的数据源小



hao1111205

2019-08-14

通过 SDP 来控制传输速率, 是否可以通过修改SDP里的采样率来实现?

作者回复: 修改采样率也等于是将源的大小变小, 也起到了控制传输速率的效果, 当分辨率太大, 压缩码率下不来的时间就需要降分辨率



许童童

2019-08-13

能否这样理解，码率越大，视频越清晰。

展开 ▼

作者回复: 是的

💬 2

