调试libRTMP代码来分析RTMP协议

配置好的VS2010可调试的libRTMP工程如下：

<https://github.com/jiayayao/librtmp>

该工程可以使用VS调试RTMP协议内部的代码，并且对RTMP协议部分做了详细的注释。

RTMP Server可以采用Nginx-RTMP module的方式，具体可以搜索一下。

RTMP\_ConnectStream()时接收到的packet的type依次是：

0x05: Set server bindwidth（BW = 5000000）

0x06: Set client bindwidth（BW = 5000000）

0x01: Set in chunk size（4096）

0x20:Invoke <\_result>

(object begin)

(object begin)

Property: <Name: fmsVer, STRING: FMS/3,0,1,123>

Property: <Name: capabilities, NUMBER: 31.00>

(object end)

(object begin)

Property: <Name: level, STRING: status>

Property: <Name: code, STRING: NetConnection.Connect.Success>

Property: <Name: description, STRING: Connection succeeded.>

Property: <Name: objectEncoding, NUMBER: 0.00>

(object end)

(object end)

0x20:Invoke <\_result>

(object begin)

Property: NULL

(object end)

0x20:Invoke <onStatus>

(object begin)

Property: NULL

(object begin)

Property: <Name: level, STRING: status>

Property: <Name: code, STRING: NetStream.Publish.Start>

Property: <Name: description, STRING: Start publishing>

(object end)

(object end)

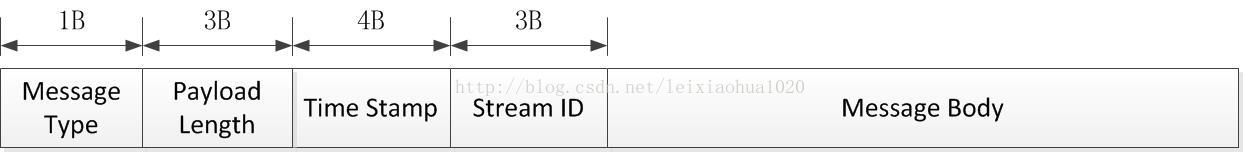
服务器接收到“connect”消息后，会返回\_result给客户端，客户端接收到是connect的response后，会发送“createStream”命令到服务器。

服务器接收到“createStream”消息后，会返回\_result给客户端，客户端接收到是“createStream”命令返回的response后，会发送“publish”命令到服务器。

RTMP协议是一个互联网TCP/IP五层体系结构中应用层的协议。RTMP协议中基本的数据单元称为消息（Message）。当RTMP协议在互联网中传输数据的时候，消息会被拆分成更小的单元，称为消息块（Chunk）。

1 消息

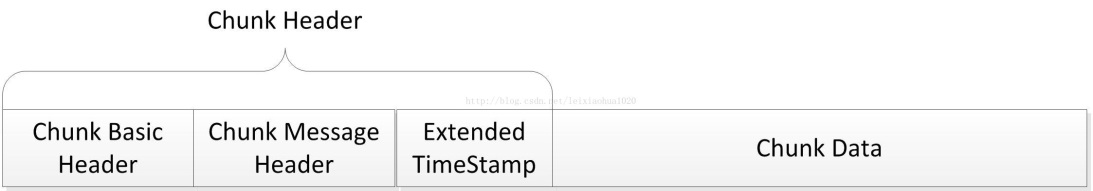
消息是RTMP协议中基本的数据单元。不同种类的消息包含不同的Message Type ID，代表不同的功能。RTMP协议中一共规定了十多种消息类型，分别发挥着不同的作用。例如，Message Type ID在1-7的消息用于协议控制，这些消息一般是RTMP协议自身管理要使用的消息，用户一般情况下无需操作其中的数据。Message Type ID为8，9的消息分别用于传输音频和视频数据。Message Type ID为15-20的消息用于发送AMF编码的命令，负责用户与服务器之间的交互，比如播放，暂停等等。消息首部（Message Header）有四部分组成：标志消息类型的Message Type ID，标志消息长度的Payload Length，标识时间戳的Timestamp，标识消息所属媒体流的Stream ID。消息的报文结构如图3所示。



消息

2 消息块

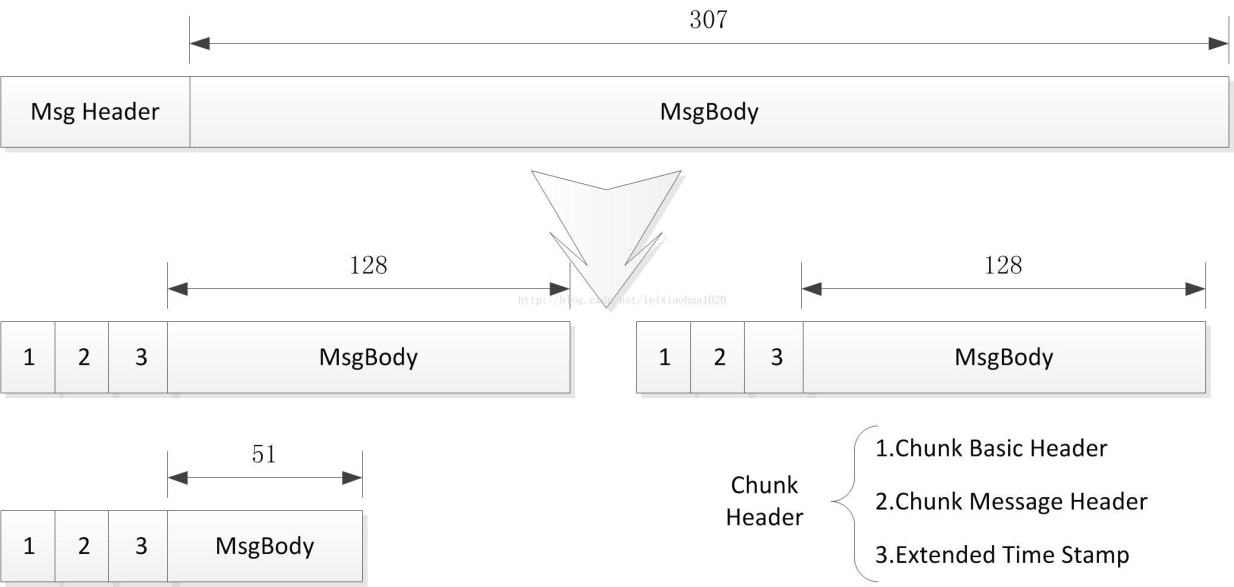
在网络上传输数据时，**消息需要被拆分成较小的数据块**，才适合在相应的网络环境上传输。RTMP协议中规定，消息在网络上传输时被拆分成消息块（Chunk）。消息块首部（Chunk Header）有三部分组成：用于标识本块的Chunk Basic Header，用于标识本块负载所属消息的Chunk Message Header，以及当时间戳溢出时才出现的Extended Timestamp。消息块的报文结构如图4所示。



消息块

3 消息分块

在消息被分割成几个消息块的过程中，消息负载部分（Message Body）被分割成大小固定的数据块（默认是128字节，最后一个数据块可以小于该固定长度），并在其首部加上消息块首部（Chunk Header），就组成了相应的消息块。消息分块过程如图5所示，一个大小为307字节的消息被分割成128字节的消息块（除了最后一个）。



RTMP分块

RTMP传输媒体数据的过程中，发送端首先把媒体数据封装成消息，然后把消息分割成消息块，最后将分割后的消息块通过TCP协议发送出去。接收端在通过TCP协议收到数据后，首先把消息块重新组合成消息，然后通过对消息进行解封装处理就可以恢复出媒体数据。

播放一个RTMP协议的流媒体需要经过以下几个步骤：握手，建立连接，建立流，播放。RTMP连接都是以握手作为开始的。建立连接阶段用于建立客户端与服务器之间的“网络连接”；建立流阶段用于建立客户端与服务器之间的“网络流”；播放阶段用于传输视音频数据。

RTMP握手过程如下：

1. 客户端向服务器发送C0、C1块，服务器收到后发送S0和S1块；
2. 客户端收到S0和S1后，向服务器发送C2块；服务器收到C2块后发送S2块；
3. 客户端和服务器分别收到S2和C2后，握手建立完成。

RTMP默认的chunk size是128字节。

消息是RTMP协议的基本数据单元，消息的格式如下：

| Message Type | Payload length | Time stamp | Stream ID | Message Body |

Message Type(1 byte)：表示消息类型，1-7内的消息ID用于协议控制；8表示音频，9表示视频；

Payload length(3 bytes)：表示负载的字节数；

Time stamp(4 bytes)：表示时间戳；

Stream ID(3 bytes)：标识消息流。

虽然消息是RTMP协议的基本数据单元，但是RTMP传输数据是将Message拆分成Chunk发送，并且拆分时，拆分的Chunk Header中包含的Basic Header标识该块从属于哪个Message，接收端也会根据这个id来将chunk组装成Message。

Chunk包括Chunk Header和Chunk Data，如下所示：

| Chunk Basic Header | Chunk Message Header | Extended Timestamp | Chunk Data |

Chunk Basic Header的长度可能是1，2，或3个字节，这取决于块流ID，其中chunk type的长度是固定的（占2位，注意单位是位，bit）。

Chunk Message Header有多种类型，类型0：

类型1：

类型2：

类型3：