Adobe公司的实时消息传输协议

摘要

此备忘录描述了 Adobe公司的实时消息传输协议(RTMP)，此协议从属于应用层，被设计用来在适合的传输协议（如TCP）上复用和打包多媒体传输流（如音频、视频和互动内容）。

目录

简介

Adobe公司的实时消息传输协议(RTMP)提供了一套全双工的可靠的多路复用消息服务，类似于TCP协议[RFC0793]，用来在一对结点之间并行传输带时间戳的音频流，视频流，数据流。通常情况下，不同类型的消息会被分配不同的优先级，当网络传输能力受限时，优先级用来控制消息在网络底层的排队顺序。

当前文章描述了实时消息传输协议(RTMP)的语法和实现。

术语

当前文章中的这些关键字"MUST""MUST NOT""REQUIRED""SHALL""SHALL NOT""SHOULD""SHOULD NOT""RECOMMENDED""NOT RECOMMENDED""MAY""OPTIONAL"的说明请参照[RFC2119]中对应的描述。

贡献者

Rajesh Mallipeddi，Adobe公司前员工，是这份规范的原始作者，并且提供了大部分的原始文本。
Mohit Srivastava，Adobe公司员工，为这份规范的完善做出了很大贡献。

定义

有效数据

RTMP包中的数据，例如音频采样、压缩后的视频数据。有效数据的格式和说明不在本文中详述。

备注：有效数据格式的定义请参考FLV定义。

包

一个数据包包括固定的包头和有效数据。一些底层协议可能需要封装包的定义。

端口

传输协议使用端口来区分同一个主机上的多个目标地址。TCP/IP协议根据正整数(端口号)来识别端口。OSI传输层使用的传输选择器(TSEL)相当于端口。

传输地址

一个网络地址和端口的组合，用来标识传输层的一端，例如IP地址和TCP端口。数据包从源地址传输到目的地址。

消息流

一个信息流的逻辑通信信道。

消息流ID

每条消息都有一个与之相关联的ID，用来区分这条消息属于哪个流。

块

消息的一个分片。消息在被发送到网络之前，被分割成更小的部分。块可以确保在多个流之间，使用基于时间戳的方式，端到端的交付所有消息。

块流

一个逻辑通信信道，允许块在一个特定的方向上流动。块流可以从客户机传输到服务器，也可以反向传输。

块流ID

每个块都有一个与之相关联的ID，用来区分块属于哪个流。

多路复用

将独立的音频或视频数据组合为连贯的音视频数据流，使同时传输多个视频和音频成为可能。

多路分解

多路复用的逆向过程，将混合的音视频数据拆分成独立的音频和视频数据。

远程过程调用(RPC)

允许客户端或服务器请求对端调用程序或子程序。

元数据

描述数据。电影的元数据包括标题、时间、创建日期、等等。

应用实例

当客户端发送连接请求到服务器时，会在服务器端建立一个应用实例。

动作消息格式(AMF)

一种紧凑的二进制格式，用来序列化AS对象。AMF有两个版本：AMF0、AMF3。

注: AMF0代表早期的flex对象，AMF3代表flash对象；

字节序,对齐,和时间格式

所有的整形字段使用网络字节序的方式传输，第零字节位于第一位，第零比特是双字节或字段的标志位。这种节字排序方式通常称为大端。传输顺序的描述详见IP协议[RFC0791]。除非特别声明，本文中的数字一律为十进制。

注: 网络字节序为大端排序方式。

除非特别说明，RTMP协议中的数据都以字节对齐。例如，一个16比特数字可能位于奇数偏移字节。涉及到追加数据的，追加字节应该为零。

RTMP的时间戳，是以毫秒为单位的整型数，使用相对时间。通常，每个流以零作为时间戳的起始，但这不是必需的，只要他们的基准时间一样即可。注意，这意味着，任何同步传输的多个流(特别是单一主机)的时间戳需要在RTMP协议外做一些额外处理。

时间戳是一个32位整型数，使用周期为49天17时2分47.296秒。由于，流允许连续传输数年时间，RTMP应用程序在处理时间戳时，应该使用序列号算法[RFC1982]。例如，应用程序假设所有相邻的时间戳都在1到2^32毫秒之间，比如4000000000后面跟着10000，3000000000在4000000000的前面。

时间戳增量是相对于前一个时间戳的无符号整型数。时间戳增量可以是24位或者32位。

RTMP块流

本讲述了实时消息传递协议块流(RTMP块流)。它作为一款高级多媒体流协议提供了流的多路复用和打包服务。RTMP块流被设计用来传输实时消息协议(第6章)，它可以使用任何协议来发送消息流。每个消息都包含时间戳和有效类型标识。RTMP块流和RTMP适用于各种视听传播的应用程序，包括一对一的，和一对多的视频直播、点播服务、互动会议应用程序。

当使用一个可靠的传输协议如TCP[RFC0793]时，RTMP块流提供了一种可以在多个流中，基于时间戳的端到端交付所有消息的方法。RTMP块流不提供任何优先级或类似形式的控制，但可以使用更高级别的协议来提供这样的优先级。例如，一个视频服务器可以根据发送的时间或确认每个消息的时间，来决定为一个网络差的用户丢弃视频信息，以确保音频信息的及时接收。

RTMP块流不仅包含了自己的协议控制信息，同时也提供了一个更高级别的协议机制，用来嵌入用户自定义控制信息。

消息格式

消息格式可以被分割成多个块，用来在更高的协议中支持多路复用。在创建块消息格式时，应该包含以下字段:

时间戳

消息的时间戳。这个字段占用4字节。

长度

消息的有效长度。如果消息头不能被忽略，它应该包括长度。这个字段在块头中占用3字节。

类型ID

各种类型的协议控制消息的ID。这些消息使用RTMP块流协议和更高级别的协议来传输信息。所有其他类型的ID可以用在高级协议，这对于RTMP块流来说，是不透明的。事实上，RTMP块流中没有要求使用这些值作为类型；所有(无协议的)消息可能是相同的类型，或者应用程序使用这个字段来区分多个连接，而不是类型。这个字段在块头中占用1字节。

消息流ID

消息流ID可以是任意值。当同一个块流被复用到不同的消息流中时，可以通过消息流ID来区分它们。另外，对于RTMP块流而言，这是一个不透明值。该字段占用4字节，使用小端序。

握手

RTMP连接从握手开始。它包含三个固定大小的块，不像其他的协议，是由头部大小可变的块组成的。

客户端（初始化连接的一端）和服务端发送同样的三个块。为了方便描述，客户端发送的三个块命名为C0，C1，C2；服务端发送的三个块命名为S0，S1，S2。

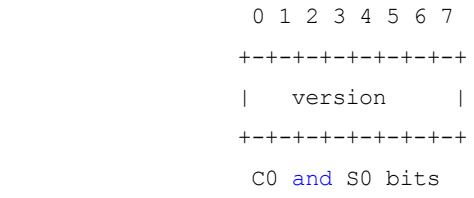
握手序列

客户端通过发送C0和C1消息来启动握手过程。客户端必须接收到S1消息，然后发送C2消息。客户端必须接收到S2消息，然后发送其他数据。

服务端必须接收到C0或者C1消息，然后发送S0和S1消息。服务端必须接收到C1消息，然后发送S2消息。服务端必须接收到C2消息，然后发送其他数据。

C0和S0格式

C0和S0包由一个字节组成，下面是C0/S0包内的字段：



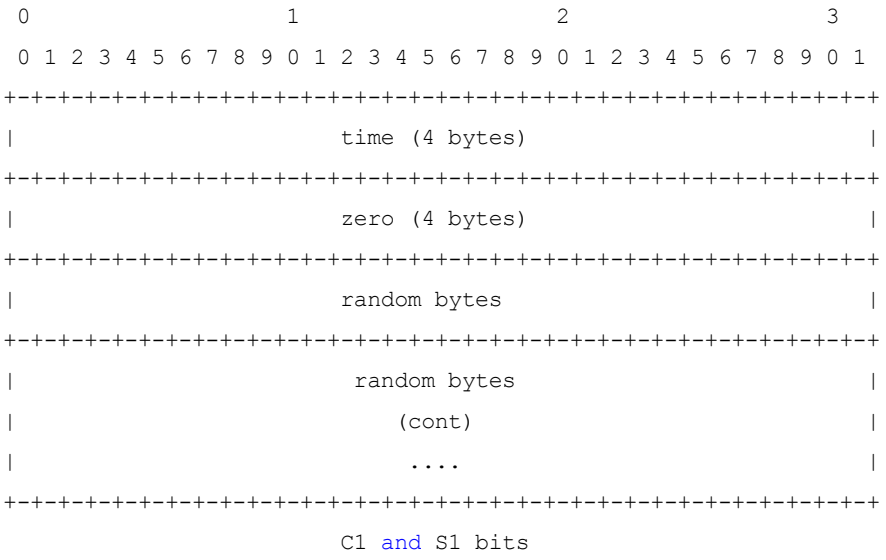
1-4B 版本

版本(8比特)

在C0包内，这个字段代表客户端请求的RTMP版本号。在S0包内，这个字段代表服务端选择的RTMP版本号。此文档使用的版本是3。版本0-2用在早期的产品中，现在已经被弃用；版本4-31被预留用于后续产品；版本32-255（为了区分RTMP协议和文本协议，文本协议通常以可打印字符开始）不允许使用。如果服务器无法识别客户端的版本号，应该回复版本3。客户端可以选择降低到版本3，或者中止握手过程。

C1和S1格式

C1和S1包长度为1536字节，包含以下字段：



1-4B 时间

5-8B 零

9-1536B 其他数据

时间(4字节)

本字段包含一个时间戳，客户端应该使用此字段来标识所有流块的时刻。时间戳取值可以为零或其他任意值。为了同步多个块流，客户端可能希望多个块流使用相同的时间戳。

零(4字节)

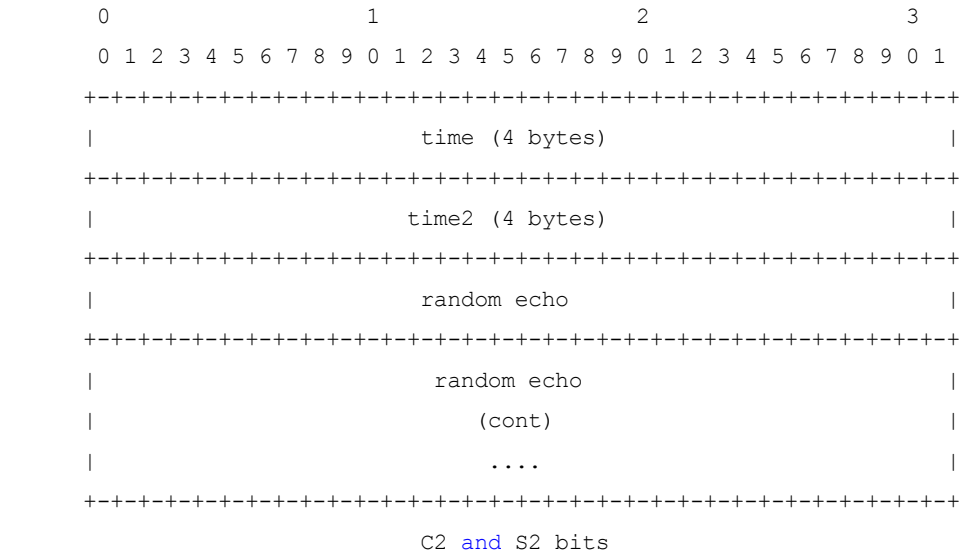
本字段必须为零。

随机数据(1528字节)

本字段可以包含任意数据。由于握手的双方需要区分另一端，此字段填充的数据必须足够随机(以防止与其他握手端混淆)。不过没必要为此使用加密数据或动态数据。

C2和S2格式

C2和S2包长度为1536字节，作为C1和S1的回应，包含以下字段:



1-4B 时间

5-8B 时间

9-1536B 其他数据

时间(4字节)

本字段必须包含对端发送的时间戳。

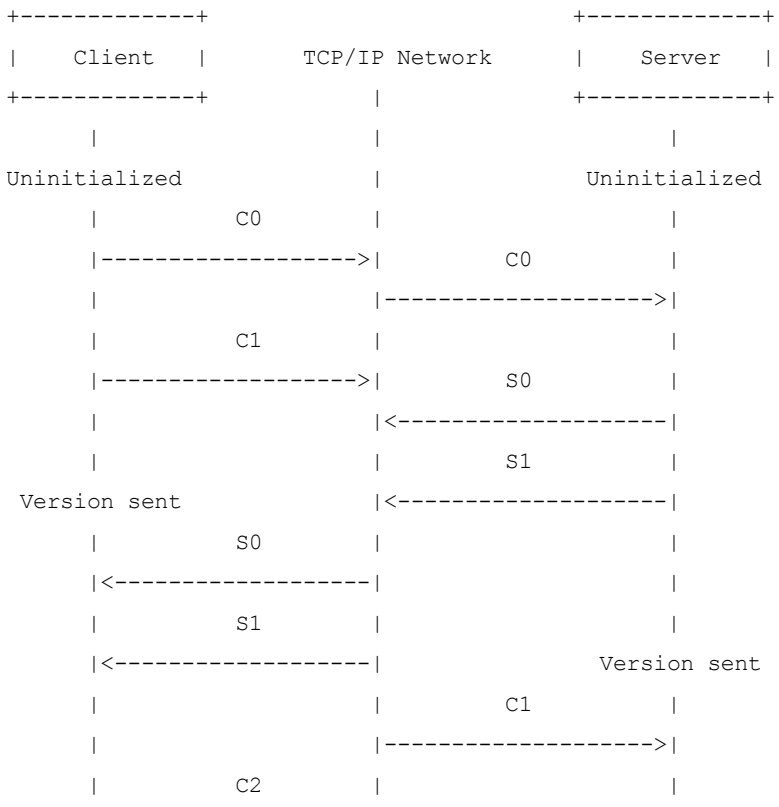
时间(4字节)

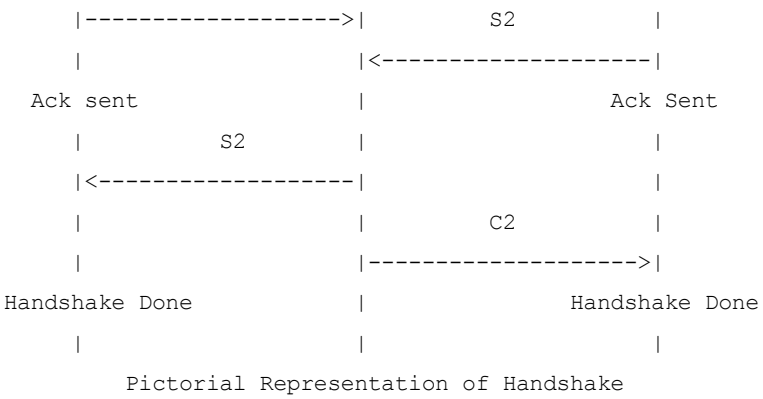
本字段必须包含时间戳，取值为接收对端发送过来的握手包的时刻。

随机数据(1528字节)

本字段必须包含对端发送过来的随机数据。握手的双方可以使用时间1和时间2字段来估算网络连接的带宽和/或延迟，但是不一定有用。

握手过程示意图





Pictorial Representation of Handshake

[译]握手示意图

下面是握手示意图中提到的状态:

未初始化

协议版本号在此阶段发送。客户端和服务器均处于未初始化状态。客户端发送携带协议版本号的C0包。如果服务器支持此版本，回复S0和S1包。如果服务器不支持此版本，使用适当的动作回复。在RTMP协议中，此动作是中止连接。

注: 在"C0和S0格式"章节中提及,如果服务器不支持客户端的版本号,可以选择降到版本3或中止。

发送版本

客户端和服务器双方在未初始化状态后，会进入发送版本状态。之后，客户端等待S1包，服务器等待C1包。待接收到数据包，客户端发送C2包，服务器发送S2包。然后，双方都进入答复状态。客户端等待C2的答复，服务器等待S2的答复。

握手完成

客户端和服务器交换消息。

组块

握手完成之后，此连接可以复用于一到多个块流。每个块流携带一个消息流的某种类型的消息。块通过网络进行传输。传输过程中，每个块必须被完整的发送后，才能发送下一个块。接收端接收完成之后，根据块流ID把块组装成完整的消息。

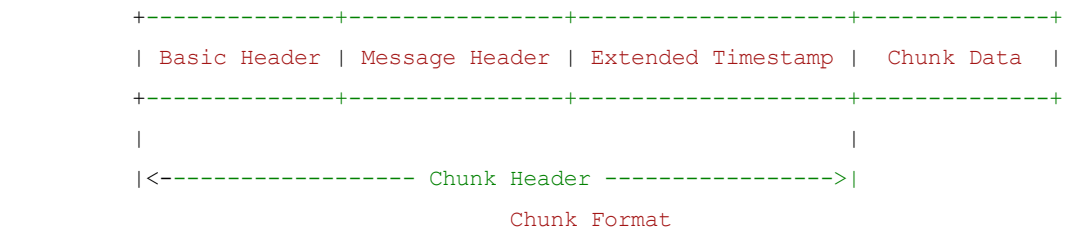
块允许把高级协议的大消息分割成更小的消息分片，例如为了防止低优先级的大消息(如视频消息)阻碍高优先级的小消息(如音频和控制消息)。

块允许以更少的开销来发送小消息，减少开销的方法是压缩必须携带的块头的数据。

块的大小是可配置的。配置的方法详见5.4.1章节的设置块大小控制消息。大的块消息可以减少CPU使用率，但是在带宽比较小时会导致其他内容的延迟。小的块消息不利于高码率流的传输。流的每一个方向都可以配置独立的块大小。

块格式

每一个块包含了头和数据。块头包含了三个部分:



Chunk Format

基本头(1-3字节)

这个字段包含块流ID和块类型。块类型决定了编码过的消息头的格式。这个字段是一个变长字段，长度取决于块流ID。

消息头(0,3,7,11字节)

这个字段包含被发送的消息信息(无论是全部,还是部分)。字段长度由块头中的块类型来决定。

扩展时间戳(0,4字节)

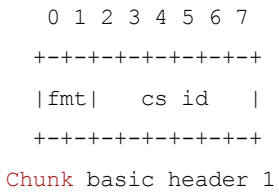
这个字段是否存在取决于块消息头中编码的时间戳。更多详情参考5.3.1.3章节。

块数据(可变大小)

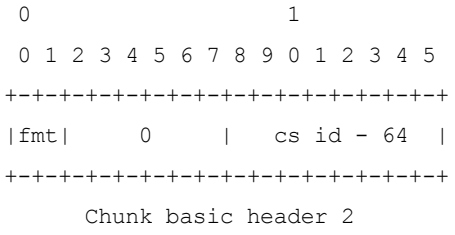
当前块的有效数据，上限为配置的最大块大小。

块的基本头

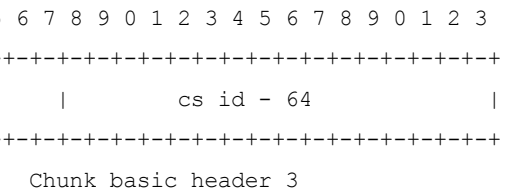
块的基本头包含块流ID和块类型(下面的fmt字段)。块类型代表了编码过的消息头的格式。此字段根据块流ID的不同，长度可能为1,2或3字节。
在实现协议时，此字段应该使用可以容纳ID的最小长度。
此协议支持最多65597个流，ID从3到65599。0,1,2这三个为保留ID。当块的基本头长度为2字节时，第3-8比特取值为0。当长度为3字节时，第3-8比特取值为1。块流ID为2时保留作为低级协议的控制消息和命令消息。
1字节长度的基本头包含了2到63的块流ID。



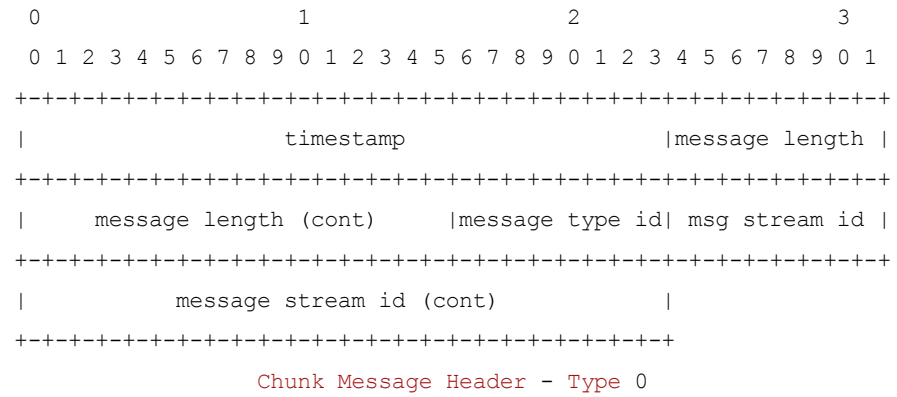
2字节长度的基本头包含了64到319的块流ID。块流ID的计算方法为，第2个字节加上64。



3字节长度的基本头包含了64到65599的块流ID。块流ID的计算方法为，第3个字节乘以256，加上第2个字节，再加上64。



块流ID(6比特)
本字段包含了从2到63的块流ID。当块的基本头长度为2字节时，此字段取值为0。当块的基本头长度为3字节时，此字段取值为1。
类型(2比特)
本字段标识了块消息头的类型。块消息头的4种类型，会在下一小节中说明。
块流ID(8或16比特)
本字段的取值为块流ID减去64。例如，块流ID为365的存储方式为，第3-8比特为1，第9-24比特为301。
64-319之间的块流ID，既可以使用2字节头长度，也可以使用3头长度。不过前面曾经提到过"在实现协议时，此字段应该使用可以容纳ID的最小长度"，因此应该选择使用2字节头长度。
块的消息头
根据块的基本头中类型字段的取值不同，块的消息头有4种类型。
在实现协议时，此字段应该使用可以容纳消息头的最小长度。
类型0
类型为0的消息头长度为11字节。块流必须以这种类型的消息开始，当块流的时间戳回环时，也必须以这种类型的消息开始。

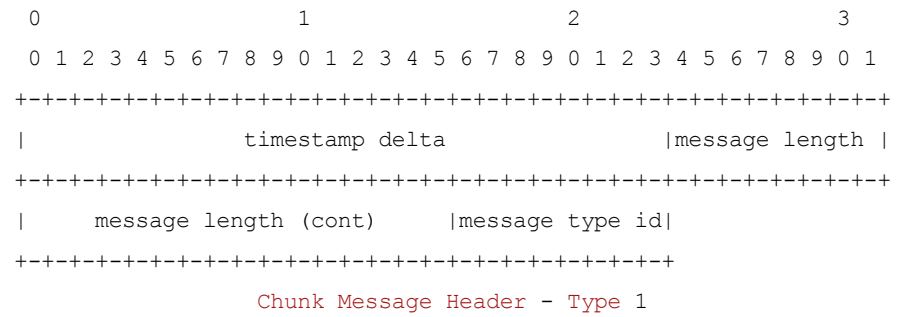


时间戳(3字节)

类型为0的块，此字段必须为绝对时间戳。如果时间戳大于等于16777215(0xFFFFFF)，此字段的取值必须为16777215，并且与扩展时间戳一起组成32比特的完整时间戳。如果时间戳小于16777215，那么此字段代表了完整的时间戳。

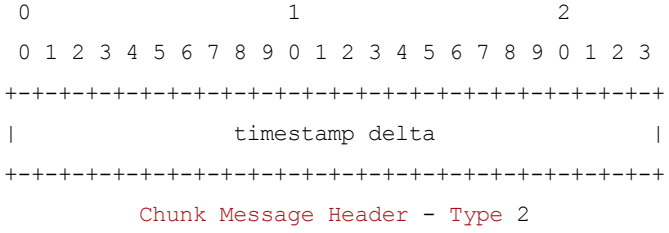
类型1

类型为1的消息头长度为7字节。不包含消息流ID；当前块和上一个块使用相同的流ID。拥有可变大小消息(视频数据)的流应该首个消息之后，在每个消息的第1个块使用这种类型。



类型2

类型为2的消息头长度为3字节。不包含消息流ID或消息长度；当前块和上一个块使用相同的流ID和流长度。拥有固定大小消息(音频或其他数据)的流应该在首个消息之后，在每个消息的第1个块使用这种类型。



类型3

类型为3的块没有消息头。消息头内的相关字段都不存在。当前块和上一个块使用相同的流ID，长度和时间间隔。
当一个独立的消息被分割成块时，从第二个块开始使用这种类型。第5.3.2.2章节的示例2展示了这种情况。
如果流由完全相同大小的消息，流ID，间隔时间组成时，应该在类型为2的块之后使用这种类型。第5.3.2.1章节的示例1展示了这种情况。
如果第一个和第二个消息之间的时间间隔和第一个消息的时间戳相同，那么类型为0的块后面跟随类型为3的块，没有必要再使用类型为2的块来标识时间间隔。
如果类型为0的块后面跟随着类型为3的块，那么类型为3的块的时间间隔与类型为1的块的时间戳相同。

常见的头字段

块消息头里面的每个字段的描述:

时间间隔(3字节)

类型为1或2的块里，本字段代表当前块和上一个块的时间戳之差。如果时间间隔大于等于16777215(0xFFFFFF)，此字段的取值必须为16777215，并且与扩展时间戳一起组成32比特的完整时间戳。如果时间戳小于16777215，那么此字段代表了完整的时间戳。

消息长度(3字节)

类型为0或1的块里，本字段代表消息的长度。这个值通常情况下跟块的有效数据长度是不相等的。块的有效数据长度是块大小的最大值。

消息类型ID(1字节)

类型为0或1的块里，本字段代表消息的类型。

消息流ID(4字节)

类型为0的块里，本字段代表消息流的ID。并使用小端方式存储。通常情况下，相同块流里面的消息来自于相同的消息流。由于消息头的压缩，可能导致独立的消息流被混合到相同的块流。无论如何，如果关闭了一个消息流，又打开了另一个，那么已经存在的块流不能被用于发送新的类型为0的块。

扩展的时间戳

扩展的时间戳字段用来表示时间戳或时间间隔取值大于16777215(0xFFFFFF)的。也就是说，类型为0或1或2的块中，时间戳或时间间隔的取值超出了24比特。本字段与之前的时间字段一起构成完整的32比特时间戳或时间间隔。
完整时间值的前一部分(24比特)位于类型为0的块的时间戳字段，或类型为1或2的时间间隔字段，取值为

16777215(0xFFFFFF)。类型为3的块中，扩展时间戳字段与上一个块中(可能为类型0或1或2)的扩展时间戳相等。

示例

示例1

本示例展示了一个音频消息流。流中包含有冗余信息。

	Message Stream ID	Message Type ID	Time	Length
Msg # 1	12345	8	1000	32
Msg # 2	12345	8	1020	32
Msg # 3	12345	8	1040	32
Msg # 4	12345	8	1060	32

Sample audio messages to be made into chunks

下面的表格展示了由此音频流产生的块信息。从第3条消息开始，数据传输达到最大优化。每条消息的头部只增加了1字节长度。

	Chunk Stream ID	Chunk Type	Header Data	No.of Bytes After Header	Total No.of Bytes in the Chunk
Chunk#1	3	0	delta: 1000 length: 32, type: 8, stream ID: 12345 (11 bytes)	32	44
Chunk#2	3	2	20 (3 bytes)	32	36
Chunk#3	3	3	none (0 bytes)	32	33
Chunk#4	3	3	none (0 bytes)	32	33

Format of each of the chunks of audio messages

示例2

本示例展示了一条长消息，由于消息的长度超过了块的最大长度(128字节)，此消息在传输时将被分割成若干个块。

	Message Stream ID	Message Type ID	Time	Length
Msg # 1	12346	9 (video)	1000	307

Sample Message to be broken to chunks

下面是消息分割后产生的块:

	Chunk Stream ID	Chunk Type	Header Data	No. of Bytes after Header	Total No. of bytes in the chunk
Chunk#1	4	0	delta: 1000 length: 307 type: 9, stream ID: 12346 (11 bytes)	128	140
Chunk#2	4	3	none (0 bytes)	128	129
Chunk#3	4	3	none (0 bytes)	51	52

Format of each of the chunks

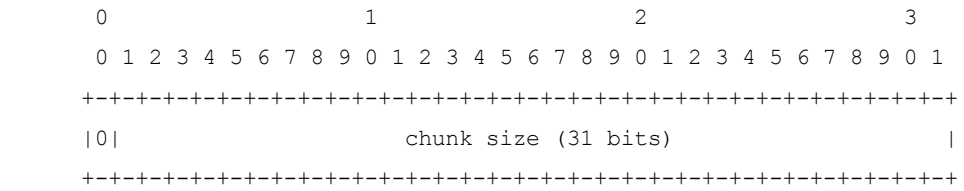
第一个块的头数据显示了消息的长度为307字节。
在这两个示例中，类型为3的块有两种使用方式。第一种是说明消息的继续。第二种是说明新消息的头信息可以由前面已经存在的消息推导出来。

协议控制消息

RTMP块流使用消息类型ID 1、2、3、5、6作为控制消息。这些消息包含了必要的RTMP块流协议信息。
这些协议控制消息必须使用0作为消息流ID(作为已知的控制流ID)，同时使用2作为块流ID。协议控制消息接收立即生效；解析时，时间戳字段被忽略。

设置块大小 (1)

协议控制消息(1)，设置块大小，被用来通知对方新的最大的块大小。
默认最大的块大小为128字节，客户端和服务器可以使用此消息来修改默认的块大小。例如，假设客户端想要发送的音频数据大小为131字节，而块大小为128字节。在这种情况下，客户端可以通知服务器新的块大小为131字节，然后就可以使用一个块来发送完整的音频数据了。
最大的块大小至少为128字节，块至少携带1个字节的内容。通信的每一个方向(例如从客户端到服务器)拥有独立的块大小设置。



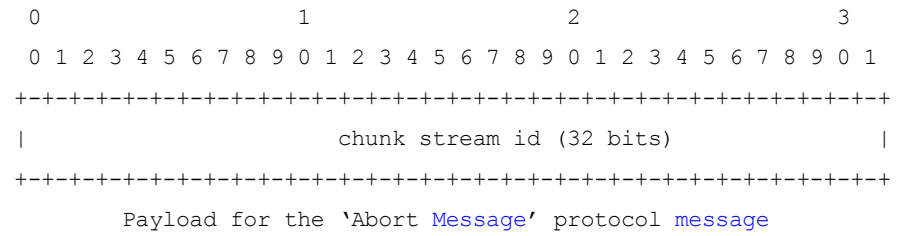
Payload for the 'Set Chunk Size' protocol message

0
当前比特位必须为零。
chunk size (31 bits): This field holds the new maximum chunk size, in bytes, which will be used for all of the sender's subsequent chunks until further notice. Valid sizes are 1 to 2147483647 (0x7FFFFFFF) inclusive; however, all sizes greater than 16777215 (0xFFFFFF) are equivalent since no chunk is larger than one message, and no message is larger than 16777215 bytes.
块大小(31比特)

本字段标识了新的最大块大小，以字节为单位，发送端之后将使用此值作为最大的块大小。本字段的有效值为1-2147483647(0x7FFFFFFF)，由于消息的最大长度为16777215(0xFFFFFF)，而一个块最多只能携带一条消息，因此本字段的实际有效值为1-16777215(0xFFFFFF)。

中断消息 (2)

协议控制消息(2)，中断消息，用来通知通信的对方，如果正在等待一条消息的部分块(已经接收了一部分)，那么可以丢弃之前已经接收到的块。通信的一方将接收到块流ID作为当前协议消息的有效数据。应用程序可以发送此消息来通知对方，当前正在传输的消息没有必要再处理了。

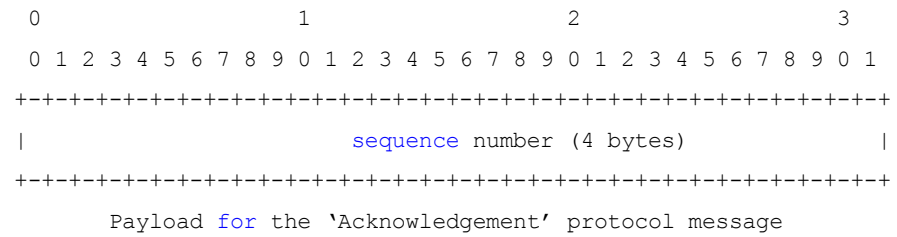


块流ID(32比特)

本字段包含了块流ID，用来标识哪个块流ID的消息将被丢弃。

应答 (3)

客户端和服务器在接收到与接收窗口大小相等的数据后，必须发送答应消息给对方。窗口大小的定义为发送方在接收到接收方的任何应答前，可以发送的最大数据量。本消息包含了序列号，序列号为截止目前接收到的数据总和，以字节为单位。

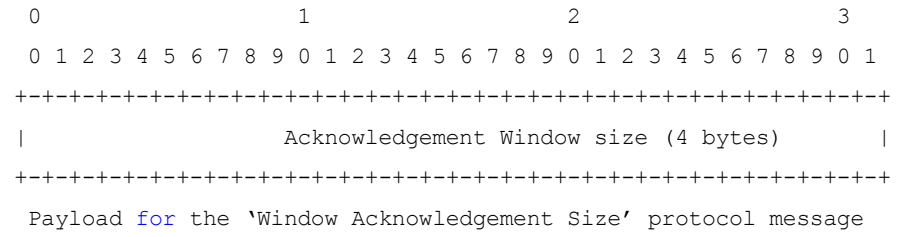


序列号(32比特)

本字段包含了截止目前接收到的数据总和，以字节为单位。

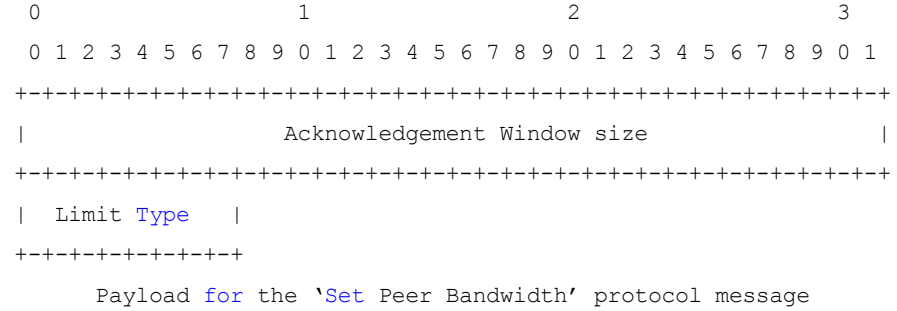
应答窗口大小 (5)

客户端和服务器发送这个消息来通知对方答应窗口的大小。发送方在发送了等于窗口大小的数据之后，等待接收对方的应答消息(在接收到答应之前停止发送数据)。接收方必须发送应答消息，在会话开始时，或从上一次发送应答之后接收到了等于窗口大小的数据。



设置流带宽 (6)

客户端和服务器发送此消息来说明对方的出口带宽限制。接收方以此来限制自己的出口带宽，即限制未被答应的消息数据大小。接收到此消息的一方，如果窗口大小与上次发送的不一致，应该回复答应窗口大小的消息。



限制类型的取值为下面之一:

硬限制(0)
应该限制出口带宽为指明的窗口大小。
软限制(1)
应该限制出口带宽为指明的窗口大小，或已经生效的小一点的窗口大小。
动态限制(2)
如果上一次为硬限制，此消息被视为硬限制，否则忽略此消息。

RTMP消息格式

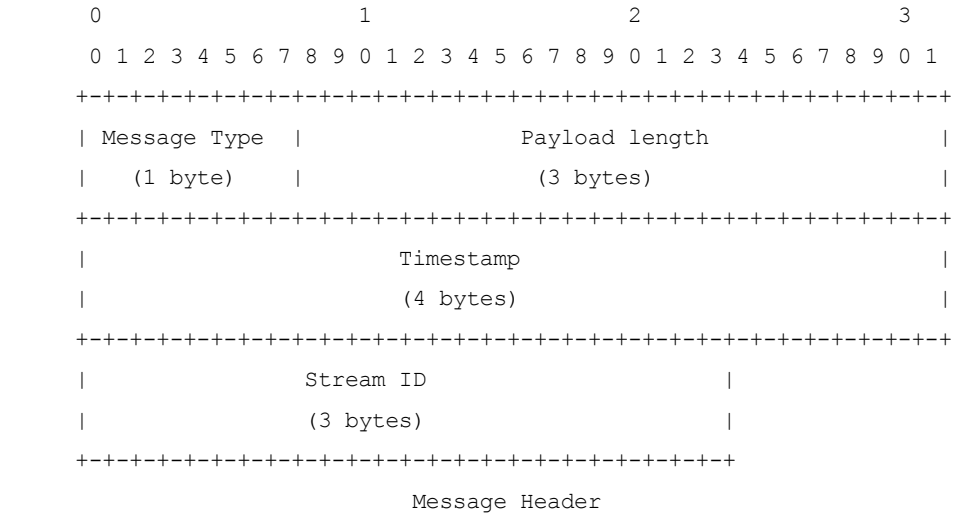
本章描述了RTMP消息的格式，使用网络传输层在两个实体端之间做数据传输，例如RTMP块流。由于RTMP协议基于RTMP块流，因此可以适配任何传输协议。RTMP块流和RTMP相结合，可以广泛的使用于音视频应用，点对点，点对多的广播应用，点播服务，以及交互式会议应用。

RTMP消息格式

服务器和客户端通过网络发送RTMP消息来和对方交流。消息可以包含音频，视频，或其他消息。
RTMP消息有两部分组成，头部和有效数据。

消息头

消息头部包含以下字段:
消息类型
消息类型，1字节长度。从1到6的ID被保留用于协议控制消息。
长度
消息的有效数据大小，3字节长度。使用大端方式。
时间戳
消息的时间戳，4字节长度。使用大端方式。
消息流ID
消息流标识，3字节长度。使用大端方式。

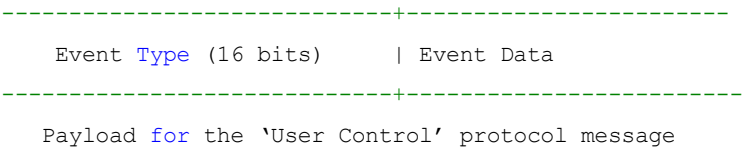


消息的有效数据

RTMP消息的另一部分是有效数据，存放的是真正的消息数据内容。例如，一些音频采样，或压缩过的视频数据。有关数据的格式和解释不在本章节说明。

用户控制消息 (4)

用户控制消息的类型为4。此消息位于RTMP协议流层。RTMP块流使用消息类型ID 1、2、3、5、6作为控制消息(见第5.4章)。
用户控制消息应该使用0作为消息流ID，当通过RTMP块流发送此消息时，块流ID为2。RTMP流中的用户控制消息在接收时立即生效，消息中的时间戳被忽略。
客户端或服务器发送此消息用来通知对方用户控制事件。此消息包含事件类型和事件数据。



用户控制消息的前2个字节数据用来标识事件类型。事件类型后面是事件数据。事件数据字段是可变的。由于此消息是通过RTMP块流层发送的，块大小的最大值(第5.4.1章节)应该满足在一个块里包含此消息。
有关事件类型和事件数据格式的说明见第7.1.7章节。

RTMP命令消息

这部分描述了服务器和客户端之间交互使用到的不同类型的消息和命令。
服务器和客户端之间交互的消息类型包括：用于发送音频数据的音频消息，用于发送视频数据的视频消息，用于发送用户自定义数据、共享对象和命令的数据消息。共享对象提供了一种通用的方式来管理多个客户端和服务器之间的分布式消息。命令消息使用AMF编码的命令在客户端和服务器之间交互。客户端或服务器可以通请求远程过程调用(RPC)，使用命令消息和对方交流。

消息类型

服务器和客户端通过网络进行消息交互。消息可以是音频消息、视频消息、命令消息、共享对象消息、数据消息和用户自定义消息。

命令消息(20, 17)

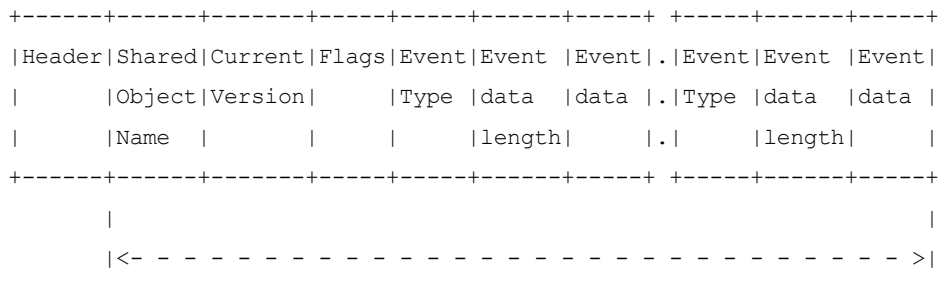
服务器和客户端之间使用AMF编码的命令消息交互。命令消息在AMF0编码中，类型为20；在AMF3编码中，类型为17。一些命令消息被用来发送操作指令，比如connect, createStream, publish, play, pause。另外一些命令消息被用来通知发送方请求命令的状态，比如onstatus, result等。一条命令消息包括命令名称、交互ID、包含相关参数的命令对象。服务器和客户端通过在创建的流中远程调用的方式，使用命令消息来进行交互。

数据消息 (18, 15)

客户端或服务器使用此消息来发送元数据或其他用户数据。元数据包含了(音视频)数据的细节信息，像流的创建时间，时间点，主题等等。数据消息在AMF0编码中，类型为18；在AMF3编码中，类型为15。

共享对象消息 (19, 16)

共享对象是Flash对象，可以通过多客户端，实例同步传输。在AMF0编码中，类型为19；在AMF3编码中，类型为16。每个消息可以包含多个事件。



The shared object message format

下面是共享消息支持的事件类型:

Event	Description
Use(=1)	The client sends this event to inform the server about the creation of a named shared object.
Release(=2)	The client sends this event to the server when the shared object is deleted on the client side.
Request Change (=3)	The client sends this event to request that the change the value associated with a named parameter of the shared object.
Change (=4)	The server sends this event to notify all clients, except the client originating the request, of a change in the value of a named

	parameter.
Success (=5)	The server sends this event to the requesting client in response to RequestChange event if the request is accepted.
SendMessage (=6)	The client sends this event to the server to broadcast a message. On receiving this event, the server broadcasts a message to all the clients, including the sender.
Status (=7)	The server sends this event to notify clients about error conditions.
Clear (=8)	The server sends this event to the client to clear a shared object. The server also sends this event in response to Use event that the client sends on connect.
Remove (=9)	The server sends this event to have the client delete a slot.
Request Remove (=10)	The client sends this event to have the client delete a slot.
Use Success (=11)	The server sends this event to the client on a successful connection.

音频消息 (8)

客户端或服务器使用此消息来发送音频消息。此消息的类型为8。

视频消息 (9)

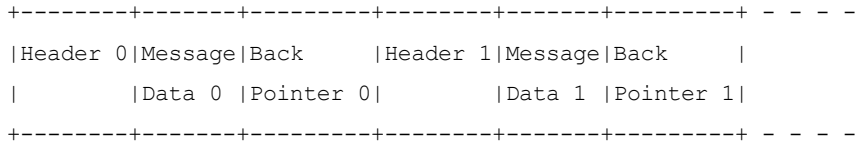
客户端或服务器使用此消息来发送视频消息。此消息的类型为9。

集合消息 (22)

集合消息是一个独立消息，包含了一系列的RTMP消息，格式描述见6.1章。此消息的类型为22。
集合消息由消息头和消息内容组成。
消息内容由子消息组成，子消息由消息头,消息数据,回放指针组成。

Header	Aggregate Message body

The Aggregate Message format



The Aggregate Message body format

集合消息的消息流ID覆盖此消息内的子消息流的ID。
集合消息和第一个子消息的时间戳之间的偏移量，用来将子消息的时间戳处理为流的时间刻度。每个子消息的时间戳可以通过添加偏移量来处理为正常的流时间。第一个子消息的时间戳应该和集合消息的时间戳相同，因此偏移量应该

为零。
反向指针包含了以前的消息(包含头信息)的大小。集合消息包含此字段，一是为了适配FLV文件格式，二是为了回放定位。
使用集合消息有如下几个优势:
块流在一个块内至多可以携带一条完整的消息。使用集合消息之后，不仅可以增加块大小，同时还减少了发送的块数量。
集合消息的子消息可以连续的存储在内存中。当系统调用网络发送数据时更高效。

用户控制消息事件

客户端或服务器通过发送此消息来通知对方用户控制事件。此消息的格式信息详见第6.2章节。
用户控制事件支持如下类型:

流开始事件(0)

Event	Description
Stream Begin (=0)	The server sends this event to notify the client that a stream has become functional and can be used for communication. By default, this event is sent on ID 0 after the application connect command is successfully received from the client. The event data is 4-byte and represents the stream ID of the stream that became functional.
Stream EOF (=1)	The server sends this event to notify the client that the playback of data is over as requested on this stream. No more data is sent without issuing additional commands. The client discards the messages received for the stream. The 4 bytes of event data represent the ID of the stream on which playback has ended.
StreamDry (=2)	The server sends this event to notify the client that there is no more data on the stream. If the server does not detect any message for a time period, it can notify the subscribed clients that the stream is dry. The 4 bytes of event data represent the stream ID of the dry stream.
SetBuffer Length (=3)	The client sends this event to inform the server of the buffer size (in milliseconds) that is used to buffer any data coming over a stream. This event is sent before the server starts processing the stream. The first 4 bytes of the event data represent the stream ID and the next 4 bytes represent the buffer length, in milliseconds.
StreamIs Recorded (=4)	The server sends this event to notify the client that the stream is a recorded stream. The

		4 bytes event data represent the stream ID of the recorded stream.
PingRequest (=6)		The server sends this event to test whether the client is reachable. Event data is a 4-byte timestamp, representing the local server time when the server dispatched the command. The client responds with PingResponse on receiving MsgPingRequest.
PingResponse (=7)		The client sends this event to the server in response to the ping request. The event data is a 4-byte timestamp, which was received with the PingRequest request.

命令类型

客户端和服务器通过AMF编码的数据交换命令。发送者发送包含命令名称，事务ID，包含相关参数的命令对象的消息。例如，通过连接命令中包含的APP参数来告诉服务器连接的对方是哪个客户端。接收方处理命令消息，并使用相同的事务ID应答。应答字符串为_result或_error或方法名，例如verifyClient或contactExternalServer。事务ID标明了应答指向的命令。事务ID相当于IMAP协议或其他协议中的标签。命令字符串中的方法名，表明了发送端想要在接收端执行的方法。

下面的类对象被用来发送各种命令:

NetConnection

服务器和客户端之间进行网络连接的一种高级表示形式。

NetStream

代表了发送音频流，视频流，或其他相关数据的频道。当然还有一些像播放，暂停之类的命令，用来控制数据流。

网络连接命令

网络连接管理着客户端和服务器之初是的双向连接。另外，它也支持异步远程命令调用。

网络连接允许使用以下的命令:

连接 connect

调用 call

停止 close

创建流 createStream

连接

客户端发送连接命令给服务器，来获取一个和服务器通信的实例。客户端发送给服务器的命令结构如下:

Field Name	Type	Description
Command Name	String	Name of the command. Set to "connect".
Transaction ID	Number	Always set to 1.
Command Object	Object	Command information object which has the name-value pairs.
Optional User Arguments	Object	Any optional information

下面是连接命令的命令对象里包含的键值对的说明:

Property	Type	Description	Example Value
app	String	The Server application name the client is connected to.	testapp
flashver	String	Flash Player version. It is the same string as returned by the ApplicationScript getversion () function.	FMSc/1.0
swfUrl	String	URL of the source SWF file making the connection.	file://C:/FlvPlayer.swf
tcUrl	String	URL of the Server. It has the following format. protocol://servername:port/appName/appInstance	rtmp://localhost:1935/test/instance1
fpad	Boolean	True if proxy is being used.	true or false
audioCodecs	Number	Indicates what audio codecs the client supports.	SUPPORT_SND_MP3
videoCodecs	Number	Indicates what video codecs are supported.	SUPPORT_VID_SORENSON
videoFunction	Number	Indicates what special video functions are supported.	SUPPORT_VID_CLIENT_SEEK
pageUrl	String	URL of the web page from where the SWF file was loaded.	http://somehost/sample.html
objectEncoding	Number	AMF encoding method.	AMF3

音频编码属性的可选值:
原始PCM，ADPCM，MP3，NellyMoser(5,8,11,16,22,44kHz)，AAC，Speex。

Codec Flag	Usage	Value
SUPPORT_SND_NONE	Raw sound, no compression	0x0001
SUPPORT_SND_ADPCM	ADPCM compression	0x0002
SUPPORT_SND_MP3	mp3 compression	0x0004
SUPPORT_SND_INTEL	Not used	0x0008

SUPPORT_SND_UNUSED	Not used	0x0010
SUPPORT_SND_NELLY8	NellyMoser at 8-kHz compression	0x0020
SUPPORT_SND_NELLY	NellyMoser compression (5, 11, 22, and 44 kHz)	0x0040
SUPPORT_SND_G711A	G711A sound compression (Flash Media Server only)	0x0080
SUPPORT_SND_G711U	G711U sound compression (Flash Media Server only)	0x0100
SUPPORT_SND_NELLY16	NellyMouser at 16-kHz compression	0x0200
SUPPORT_SND_AAC	Advanced audio coding (AAC) codec	0x0400
SUPPORT_SND_SPEEX	Speex Audio	0x0800
SUPPORT_SND_ALL	All RTMP-supported audio codecs	0x0FFF

视频编码属性的可选值:
Sorenson , V1 , On2 , V2 , H264。

Codec Flag	Usage	Value
SUPPORT_VID_UNUSED	Obsolete value	0x0001
SUPPORT_VID_JPEG	Obsolete value	0x0002
SUPPORT_VID_SORENSON	Sorenson Flash video	0x0004
SUPPORT_VID_HOMEBREW	V1 screen sharing	0x0008
SUPPORT_VID_VP6 (On2)	On2 video (Flash 8+)	0x0010
SUPPORT_VID_VP6ALPHA (On2 with alpha channel)	On2 video with alpha channel	0x0020
SUPPORT_VID_HOMEBREWV (screensharing v2)	Screen sharing version 2 (Flash 8+)	0x0040
SUPPORT_VID_H264	H264 video	0x0080

SUPPORT_VID_ALL	All RTMP-supported video codecs	0x00FF

视频函数属性的可选值:

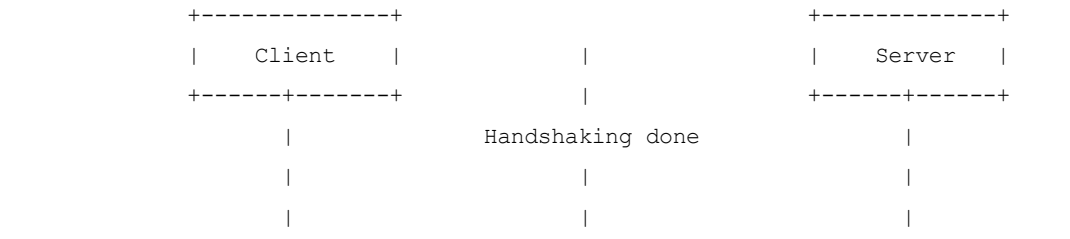
Function Flag	Usage	Value
SUPPORT_VID_CLIENT_SEEK	Indicates that the client can perform frame-accurate seeks.	1

对象编码属性的可选值:

Encoding Type	Usage	Value
AMF0	AMF0 object encoding supported by Flash 6 and later	0
AMF3	AMF3 encoding from Flash 9 (AS3)	3

服务器发送给客户端的命令结构如下:

Field Name	Type	Description
Command Name	String	_result or _error; indicates whether the response is result or error.
Transaction ID	Number	Transaction ID is 1 for connect responses
Properties	Object	Name-value pairs that describe the properties(fmsver etc.) of the connection.
Information	Object	Name-value pairs that describe the response from|the server. 'code', 'level', 'description' are names of few| among such information.



Message flow in the connect command

命令执行过程中的消息流如下:
客户端发送连接命令给服务器，获得与服务器连接的实例。
服务器在接收到连接命令后，发送应答窗口大小的消息给客户端。同时与连接命令中接到的应用建立连接。
服务器发送设置流带宽消息给客户端。
客户端在接收并处理了设置流带宽的消息后，发送应答窗口大小的消息给服务器。
服务器接着发送开始流的用户控制消息给客户端。
服务器发送result命令消息给客户端，通知连接状态是成功或失败。命令消息中包含了事务ID。消息中还包含了像FMS版本之类的属性，以及级别，编码，描述，对象编码等信息。

调用

网络连接对象中包含的call方法，会在接收端执行远程过程调用(RPC)。被调用的RPC方法名作为call方法的参数传输。
从发送端到接收端的命令结构如下:

Field Name	Type	Description
Procedure Name	String	Name of the remote procedure that is called.
Transaction ID	Number	If a response is expected we give a transaction Id. Else we pass a value of 0
Command Object	Object	If there exists any command info this is set, else this is set to null type.
Optional Arguments	Object	Any optional arguments to be provided

应答的命令结构如下:

Field Name	Type	Description
Command Name	String	Name of the command.

Transaction ID	Number	ID of the command, to which the response belongs.
Command Object	Object	If there exists any command info this is set, else this is set to null type.
Response	Object	Response from the method that was called.

创建流

客户端通过发送此消息给服务器来创建一个用于消息交互的逻辑通道。音频，视频，和元数据都是通过createStream命令创建的流通道发布出去的。

NetConnection是默认的交互通道，流ID为0。协议和一部分命令消息，包含createStream，都是使用默认的交互通道发布的。

从客户端发送给服务器的命令结构如下:

Field Name	Type	Description
Command Name	String	Name of the command. Set to "createStream".
Transaction ID	Number	Transaction ID of the command.
Command Object	Object	If there exists any command info this is set, else this is set to null type.

从服务器发送给客户端的命令结构:

Field Name	Type	Description
Command Name	String	_result or _error; indicates whether the response is result or error.
Transaction ID	Number	ID of the command that response belongs to.
Command Object	Object	If there exists any command info this is set, else this is set to null type.
Stream ID	Number	The return value is either a stream ID or an error information object.

网络流命令

网络流定义了通过网络连接把音频，视频和数据消息流在客户端和服务器之间进行交换的通道。一个网络连接对象可以多个网络流，进而支持多个数据流。

客户端可以通过网络流发送到服务器的命令如下:

播放 play
播放2 play2
删除流 deleteStream
关闭流 closeStream
接收音频 receiveAudio
接收视频 receiveVideo
发布 publish
定位 seek
暂停 pause
服务器通过发送onStatus命令给客户端来通知网络流状态的更新。

Field Name	Type	Description
Command Name	String	The command name "onStatus".
Transaction ID	Number	Transaction ID set to 0.
Command Object	Null	There is no command object for onStatus messages.
Info Object	Object	An AMF object having at least the following three properties: "level" (String): the level for this message, one of "warning", "status", or "error"; "code" (String): the message code, for example "NetStream.Play.Start"; and "description" (String): a human-readable description of the message. The Info object MAY contain other properties as appropriate to the code.

Format of NetStream status message commands.

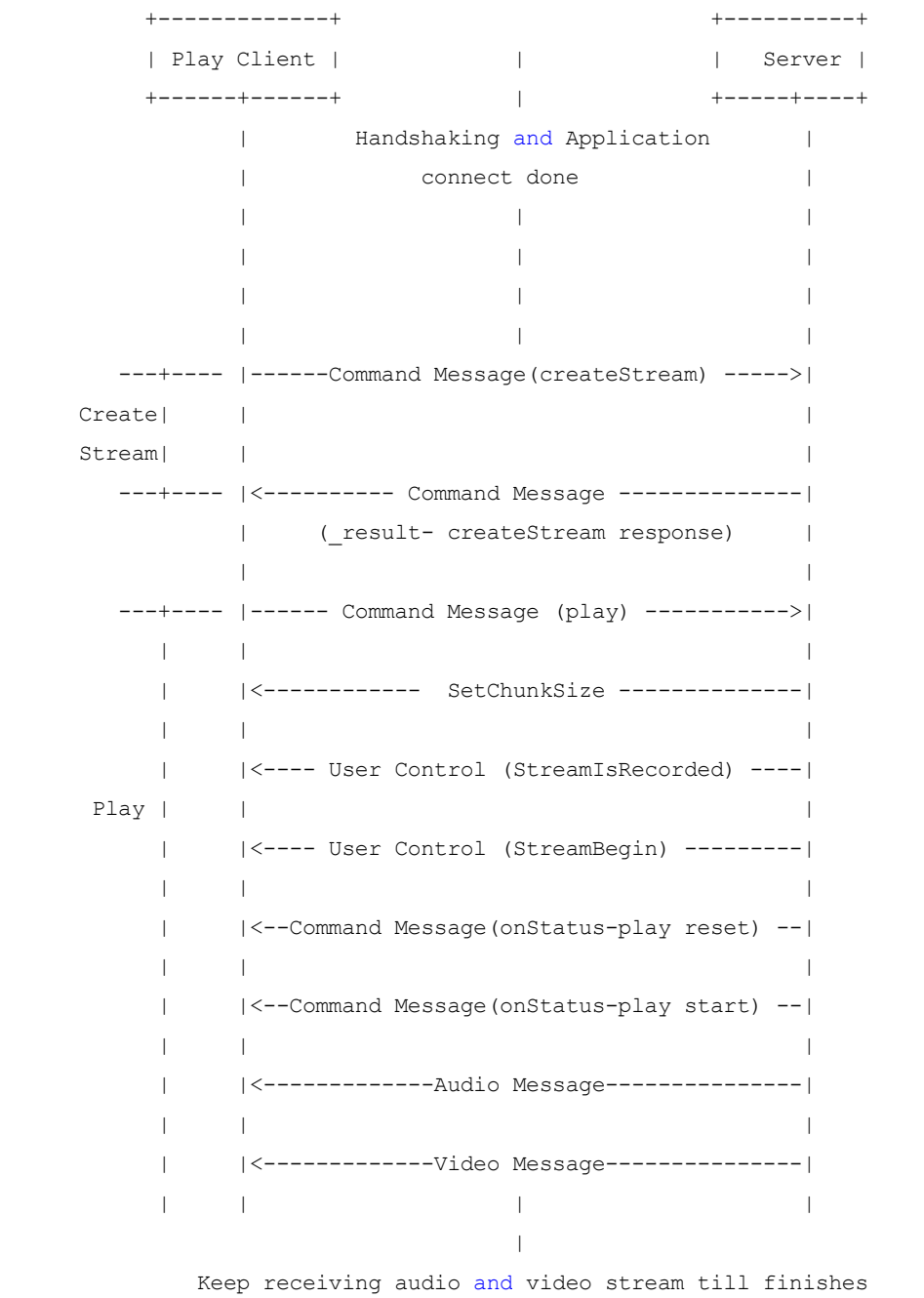
播放

客户端发送此命令来通知服务器开始播放流。多次使用此命令可以创建一个播放列表。如果想要创建一个动态播放列表来在不同的直播或点播流之间切换，可以通过多次调用播放命令，同时将Reset字段设置为false。相反，如果想要立即播放指定的流，先清理掉之前的播放队列，再调用播放命令，同时将Reset字段设置为true。
从客户端发送给服务器的命令结构如下:

Field Name	Type	Description
Command Name	String	Name of the command. Set to "play".
Transaction ID	Number	Transaction ID set to 0.
Command Object	Null	Command information does not exist. Set to null type.
Stream Name	String	Name of the stream to play.

		To play video (FLV) files, specify the
		name of the stream without a file
		extension (for example, "sample"). To
		play back MP3 or ID3 tags, you must
		precede the stream name with mp3:
		(for example, "mp3:sample". To play
		H.264/AAC files, you must precede the
		stream name with mp4: and specify the
		file extension. For example, to play the
		file sample.m4v,specify "mp4:sample.m4v"
Start	Number	An optional parameter that specifies
		the start time in seconds. The default
		value is -2, which means the subscriber
		first tries to play the live stream
		specified in the Stream Name field. If a
		live stream of that name is not found,it
		plays the recorded stream of the same
		name. If there is no recorded stream
		with that name, the subscriber waits for
		a new live stream with that name and
		plays it when available. If you pass -1
		in the Start field, only the live stream
		specified in the Stream Name field is
		played. If you pass 0 or a positive
		number in the Start field, a recorded
		stream specified in the Stream Name
		field is played beginning from the time
		specified in the Start field. If no
		recorded stream is found, the next item
		in the playlist is played.
Duration	Number	An optional parameter that specifies the
		duration of playback in seconds. The
		default value is -1. The -1 value means
		a live stream is played until it is no
		longer available or a recorded stream is
		played until it ends. If you pass 0, it
		plays the single frame since the time
		specified in the Start field from the
		beginning of a recorded stream. It is
		assumed that the value specified in
		the Start field is equal to or greater
		than 0. If you pass a positive number,
		it plays a live stream for
		the time period specified in the
		Duration field. After that it becomes

		available or plays a recorded stream	
		for the time specified in the Duration	
		field. (If a stream ends before the	
		time specified in the Duration field,	
		playback ends when the stream ends.)	
		If you pass a negative number other	
		than -1 in the Duration field, it	
		interprets the value as if it were -1.	
Reset	Boolean	An optional Boolean value or number	
		that specifies whether to flush any	
		previous playlist.	



Message flow in the play command

命令执行过程中的消息流如下：

当客户端接收到服务器返回的createStream成功的消息时，开始发送播放命令。

服务器接收到播放命令后，发送设置块大小的消息。
服务器发送一条用户控制消息，消息内包含了StreamIsRecorded事件和流ID。事件类型位于消息的前2个字节，流ID位于消息的最后4个字节。
服务器发送一条用户控制消息，消息内包含了StreamBegin事件，用于通知客户端开始播放流。
如果客户端已经成功发送了播放命令，那么服务器发送两条onStatus命令给客户端，命令的内容为NetStream.Play.Start 和 NetStream.Play.Reset。服务器只有在客户端发送了设置有重置标签的播放命令后，才能发送NetStream.Play.Reset命令。如果服务器找不到客户端请求播放的流，那么发送NetStream.Play.StreamNotFound命令给客户端。
之后，服务器发送音频和视频数据给客户端。

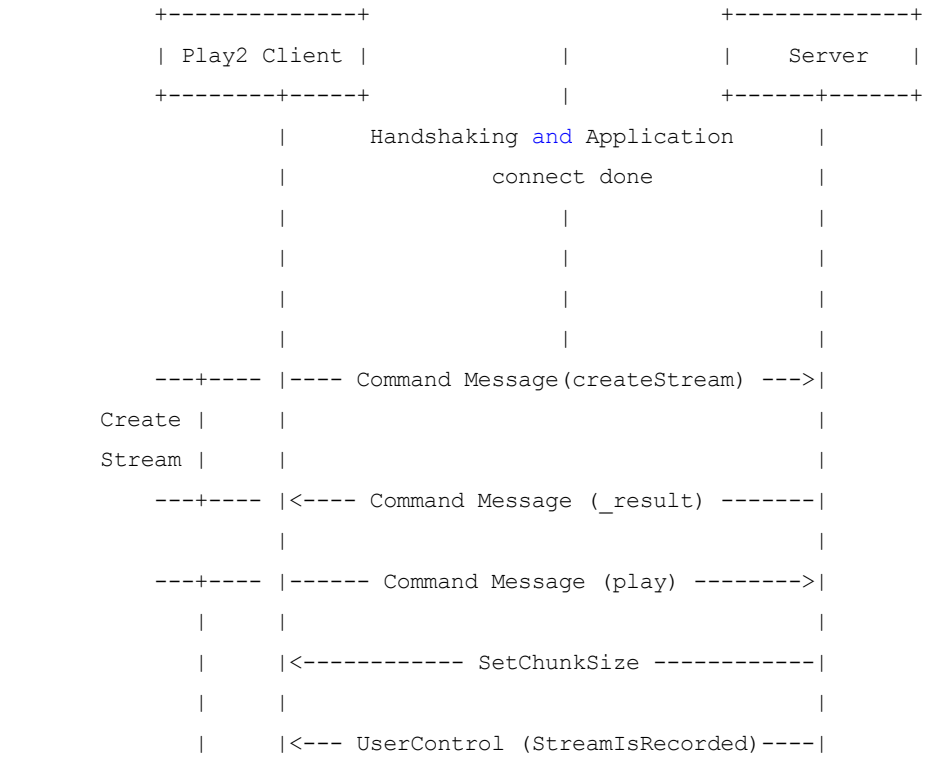
播放2

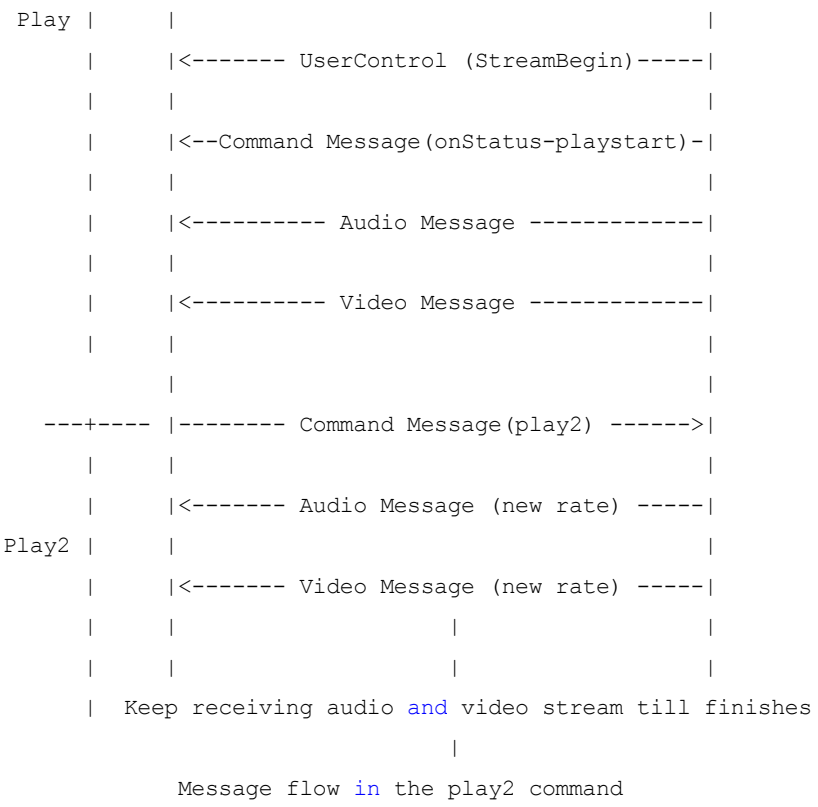
与播放命令的不同之处在于，播放2命令可以在不修改播放内容时间线的前提下切换到一个不同码率的流。服务器包含了多个不同码率的流文件用于支持客户端的播放2请求。

The command structure from the client to the server is as follows:

Field Name	Type	Description
Command Name	String	Name of the command, set to "play2".
Transaction ID	Number	Transaction ID set to 0.
Command Object	Null	Command information does not exist. Set to null type.
Parameters	Object	An AMF encoded object whose properties are the public properties described for the flash.net.NetStreamPlayOptions ActionScript object.

有关NetStreamPlayOptions对象的公开属性的说明详见AS3语言的文档。
此命令的消息流如下图所示:





Message flow in the play2 command

删除流

如果需要销毁网络流对象，可以通过网络流发送删除流消息给服务器。

客户端发送给服务器的命令结构如下：

The command structure from the client to the server is as follows:

Field Name	Type	Description
Command Name	String	Name of the command, set to "deleteStream".
Transaction ID	Number	Transaction ID set to 0.
Command Object	Null	Command information object does not exist. Set to null type.
Stream ID	Number	The ID of the stream that is destroyed on the server.

服务器接收到此消息后，不做任何回复。

接收音频

网络流发送此消息通知服务器，是否要发送音频数据给客户端。

The command structure from the client to the server is as follows:

Field Name	Type	Description
Command Name	String	Name of the command, set to "receiveAudio".

Transaction ID	Number	Transaction ID set to 0.
Command Object	Null	Command information object does not exist. Set to null type.
Bool Flag	Boolean	true or false to indicate whether to receive audio or not.

如果服务器接收到带有flase标签的消息后，不做任何回复。如果接收到带有true标签的消息，服务器回复带有NetStream.Seek.Notify和NetStream.Play.Start的消息给客户端。

接收视频

网络流发送此消息通知服务器，是否要发送视频数据给客户端。

客户端发送给服务器的命令结构如下：

Field Name	Type	Description
Command Name	String	Name of the command, set to "receiveVideo".
Transaction ID	Number	Transaction ID set to 0.
Command Object	Null	Command information object does not exist. Set to null type.
Bool Flag	Boolean	true or false to indicate whether to receive video or not.

如果服务器接收到带有flase标签的消息后，不做任何回复。如果接收到带有true标签的消息，服务器回复带有NetStream.Seek.Notify和NetStream.Play.Start的消息给客户端。

发布

客户端发送此消息，用来发布一个有名字的流到服务器。其他客户端可以使用此流名来播放流，接收发布的音频，视频，以及其他数据消息。

客户端发送给服务器的命令结构如下：

Field Name	Type	Description
Command Name	String	Name of the command, set to "publish".
Transaction ID	Number	Transaction ID set to 0.
Command Object	Null	Command information object does not exist. Set to null type.
Publishing Name	String	Name with which the stream is published.

Publishing Type	String	Type of publishing. Set to "live", "record", or "append". record: The stream is published and the data is recorded to a new file.The file is stored on the server in a subdirectory within the directory that contains the server application. If the file already exists, it is overwritten. append: The stream is published and the data is appended to a file. If no file is found, it is created. live: Live data is published without recording it in a file.

服务器接收到此消息后，回复onStatus命令来标记发布的开始。

定位

客户端发送此消息来定位多媒体文件或播放列表的偏移(以毫秒为单位)。

客户端发送给服务器的命令结构如下：

Field Name	Type	Description
Command Name	String	Name of the command, set to "seek".
Transaction ID	Number	Transaction ID set to 0.
Command Object	Null	There is no command information object for this command. Set to null type.
milliSeconds	Number	Number of milliseconds to seek into the playlist.

当定位完成后，服务器回复NetStream.Seek.Notify状态消息给客户端。如果定位失败，将回复_error消息。

暂停

客户端发送此消息来通知服务器暂停或开始播放。

客户端发送给服务器的命令结构如下：

Field Name	Type	Description
Command Name	String	Name of the command, set to "pause".
Transaction ID	Number	There is no transaction ID for this command. Set to 0.
Command Object	Null	Command information object does not exist. Set to null type.
Pause/Unpause Flag	Boolean	true or false, to indicate pausing or resuming play

milliSeconds	Number	Number of milliseconds at which the the stream is paused or play resumed. This is the current stream time at the Client when stream was paused. When the playback is resumed, the server will only send messages with timestamps greater than this value.

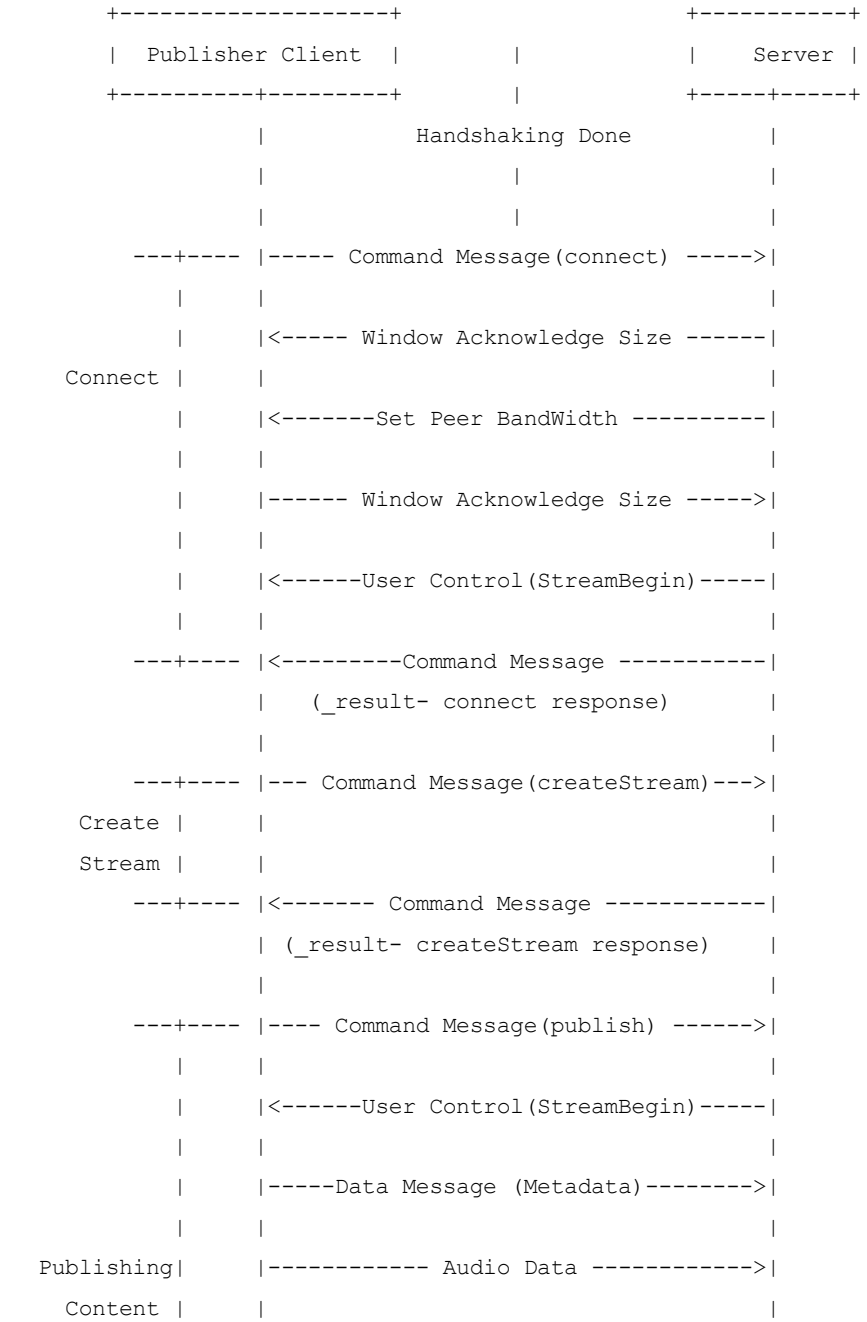
当流暂停成功，服务器发送NetStream.Pause.Notify状态消息经给客户端，如果流未暂停，服务器发送NetStream.Unpause.Notify状态消息给客户端。如果暂停失败，则发送_error消息。

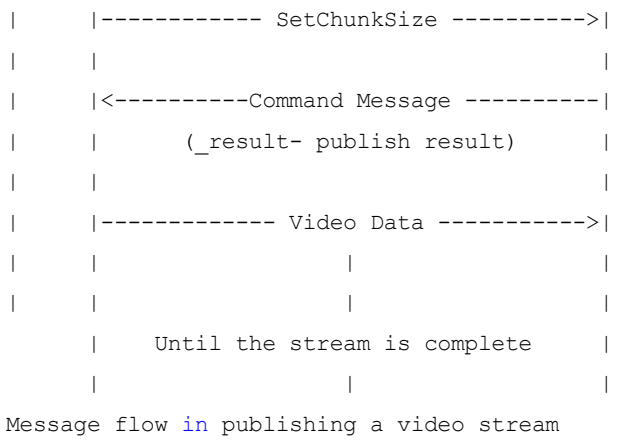
消息交互示例

下面是一些使用RTMP协议交互消息的示例。

发布录制的视频

此示例阐述了发布者如何发布视频流到服务器。其他客户端可以订阅并播放此视频流。

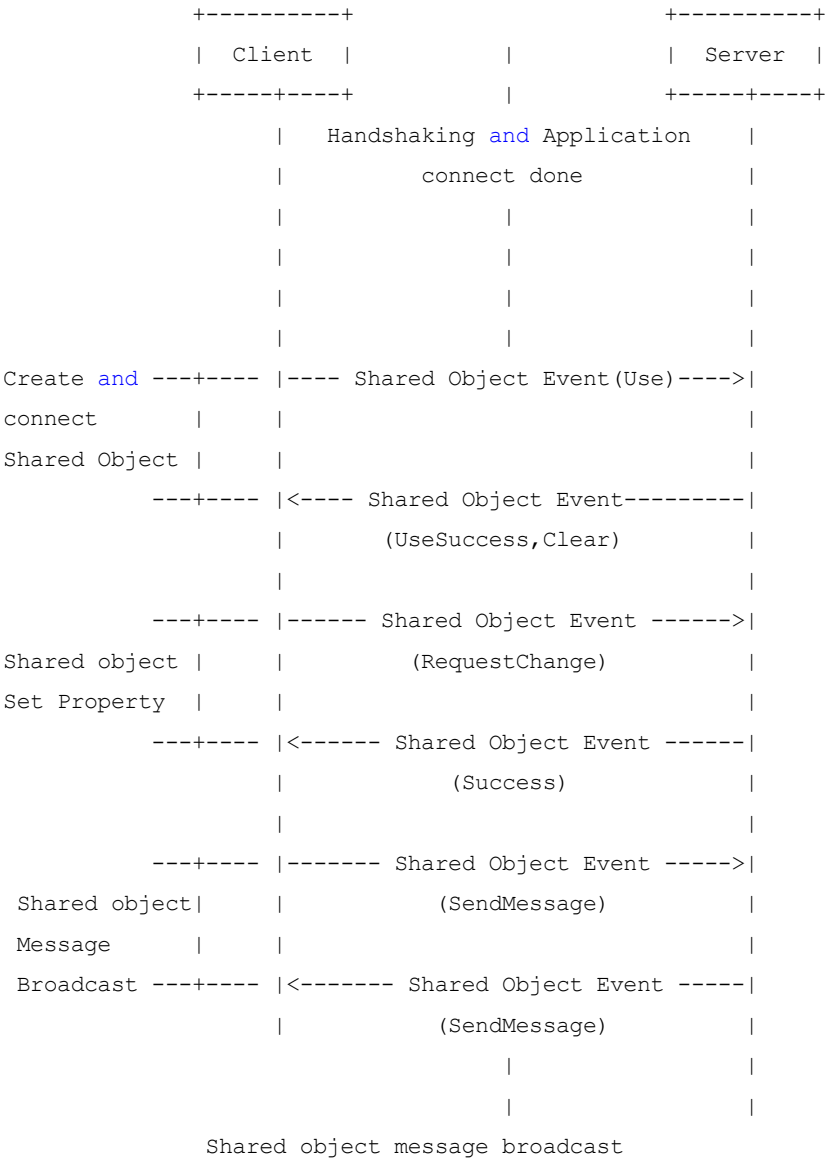




Message flow in publishing a video stream

广播共享对象消息

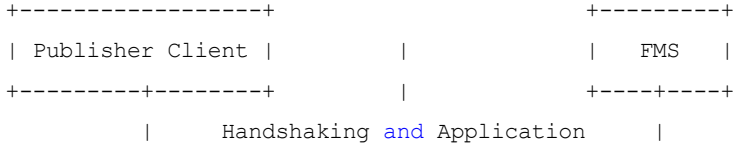
此示例阐述了流创建过程中的消息交换，和共享对象的变换过程。同时还展示了共享对象消息广播的处理过程。

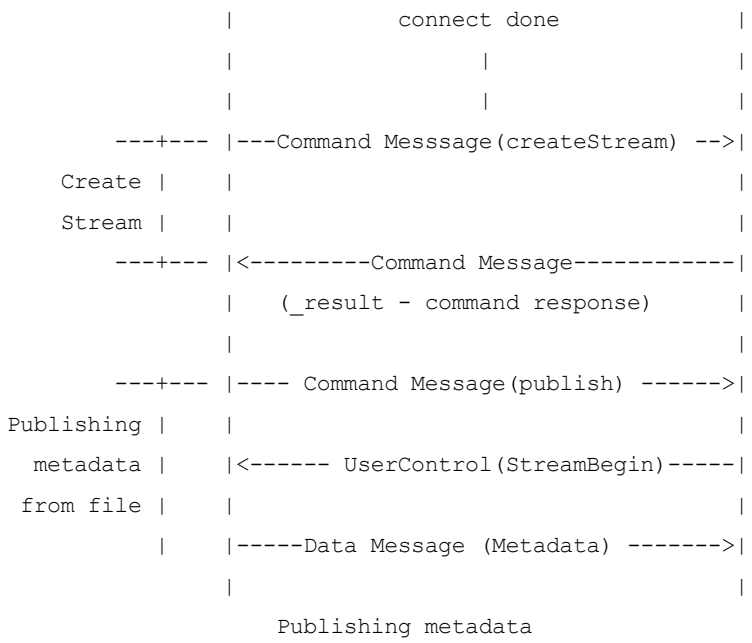


Shared object message broadcast

从录制的流发布元数据

本示例展示了发布元数据的消息交换过程。





引用

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.
[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.
[RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982, August 1996.
[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
[AS3] Adobe Systems, Inc., "ActionScript 3.0 Reference for the Adobe Flash Platform", 2011, http://www.adobe.com/devnet/actionscript/documentation.html.
[AMF0] Adobe Systems, Inc., "Action Message Format -- AMF 0", December 2007, http://opensource.adobe.com/wiki/download/attachments/1114283/amf0_spec_121207.pdf.
[AMF3] Adobe Systems, Inc., "Action Message Format -- AMF 3", May 2008, http://opensource.adobe.com/wiki/download/attachments/1114283/amf3_spec_05_05_08.pdf.

作者的联系方式

Hardeep Singh Parmar (editor)
Adobe Systems Incorporated
345 Park Ave
San Jose, CA 95110-2704
US
Phone: +1 408 536 6000
Email: hparmar@adobe.com
URI: http://www.adobe.com/
Michael C. Thornburgh (editor)
Adobe Systems Incorporated
345 Park Ave
San Jose, CA 95110-2704
US
Phone: +1 408 536 6000
Email: mthornbu@adobe.com
URI: http://www.adobe.com/